Fast Algorithms for the Minimum Volume Estimator

Selin Damla Ahipaşaoğlu
Singapore University of Technology and Design

August 14, 2014

Abstract

The MVE estimator is an important tool in robust regression and outlier detection in statistics. We develop fast and efficient algorithms for the MVE estimator problem and discuss how they can be implemented efficiently. The novelty of our approach stems from the recent developments in the first-order algorithms for solving the related Minimum Volume Enclosing Ellipsoid problem. Comparative computational results are provided which demonstrate the strength of the algorithms.

1 INTRODUCTION

The problem of enclosing a given set of points with a simple body has many applications in statistics, physical sciences, and optimization. For example, the Minimum Enclosing Ball (MEB) problem finds the ball of smallest radius that covers a given set of points. When the set to be covered is symmetric around the origin, this is a trivial problem; while for the general case, it is a convex optimization problem that can be solved efficiently. In particular, an ϵ-approximate solution for a set of m points in $\mathbb{R}^n$ can be found in $O(mn/\epsilon)$ operations as discussed in [40] and [3].

Similarly, the Minimum Volume Enclosing Ellipsoid (MVEE) problem finds an ellipsoid that encloses the given data set which is centered around the origin and has the smallest volume. It is shown in [20] that when the data set is not symmetric with respect to the origin, it can be lifted to a higher dimensional space where the (lifted) data set is centered. Therefore, the assumption that the MVEE ellipsoid is centered at the origin is not a restrictive one. The MVEE problem can be formulated as a convex

optimization problem like the MEB problem. This problem and its dual, the so-called D-optimal Design problem, have vast applications in various fields, especially in statistics, and so they have received significant attention from the statictics and optimization communities. It is known that an ϵ-approximate solution for the MVEE problem can be found in $O(mn^2/\epsilon)$ arithmetic operations for m data points in $\mathbb{R}^n$. (See [32] and [2] for the latest developments in the MVEE problem.) When the data set is given as a system of linear equations, finding the minimum volume enclosing ellipsoid is NP-complete. In principle, this ellipsoid can be calculated by solving an MVEE problem on the set of the vertices of the polytope but the vertex set may be very large. There are methods that can provide good solutions for small dimensional instances as in [12], nevertheless devising an algorithm that can handle larger problems remains a significant challenge for the global optimization community.

Using ellipsoids (or balls which are special ellipsoids) to approximate a data set has been popular due to three reasons. First, the problem of enclosing a data set with an ellipsoid is relatively easy for a family of well-defined objective functions (matrix means of the shape matrix of the ellipsoid). Second, ellipsoids are simple but flexible enough to be used in estimating many properties of the data set such as the volume of its convex hull. Third, finding the minimizer of a linear or a quadratic function over an ellipsoid is a straightforward task.

In some applications, enclosing with other bodies is of interest, nevertheless most of these problems are much harder than covering with ellipsoids. For example, enclosing a given data set with a simplex, i.e., finding $(n + 1)$ points whose convex hull encloses the data set and has the smallest possible volume, is a global optimization problem where multiple local minima may exist. This problem is referred to as the Minimum Volume Simplicial Enclosure problem (MVSEP) and is used extensively in linear mixing models, see for example [17]. Another related problem with many local minima is that of finding an enclosing minimum volume hyperrectangle. This problem is investigated in great detail in [18].

In this paper, we study the problem of enclosing a subset of a given set of points with an ellipsoid so that the ellipsoid has the smallest volume and the cardinality of the chosen subset is not smaller than a given number. This is a combinatorial problem which can be solved by an enumeration method: 1) list all possible subsets of the required cardinality; 2) find the MVEE of every subset in this list and record its volume; 3) output the subset with the smallest volume. When the data set has m points and the goal is

to enclose at least h of them, this would correspond to solving $\binom{m}{h}$ MVEE problems with h data points in each problem. This is a challenging task, if not impossible, especially when there are many points in the data set.

The paper is organized as follows. In Section 2, we introduce the Minimum Volume Ellipsoid Estimator and provide a short literature review. Section 3 provides technical background on the MVEE problem and a state-of-the-art algorithm to calculate the MVEE of a set. Section 4 provides new bounds that relate the volume of the MVEEs of two sets that differ from each other by at most one point, i.e., either one set has an extra point or two sets differ only by one point. These bounds are used in developing a 2-exchange heuristic in Section 5. Computational results follow in Section 6, before we conclude with a summary of our results in Section 7.

2 THE MVE ESTIMATOR

Given an integer h, $1 \leq h \leq m$, and an arbitrary data set $\mathcal{X} = \{x_1, \ldots, x_m\} \subset \mathbb{R}^n$, which has m distinct points, the Minimum-Volume Ellipsoid (MVE) estimator is defined to be the minimum-volume ellipsoid that encloses at least h points in the data set. Letting $\mathcal{I} = \{1, \ldots, m\}$, the problem of finding the MVE estimator of $\mathcal{X}$ can be formulated as

$$\begin{aligned} \min_{H>0, c\in\mathbb{R}^n} \quad & f(H) := -\ln\det H \\ & |\{i \in \mathcal{I} : (x_i - c)^T H (x_i - c) \leq n\}| \geq h. \end{aligned}$$

The MVE estimator is an important tool in robust regression and outlier detection in statistics. It is known that if $h = \lceil (m + n + 1)/2 \rceil$, then the MVE estimator has the maximum breakpoint, i.e., an outlier detection method based on the MVE estimator can detect outliers in data sets with many contaminated points [27]. The problem of finding the exact MVE estimator is a hard combinatorial problem with two interactive components: finding the optimum subset of at least h points and computing the minimum-volume ellipsoid that encloses this subset. Given a set of points, the minimum-volume enclosing ellipsoid (MVEE) can be computed very quickly using a coordinate-descent algorithm as discussed in Section 3 (see also [32] and [2]). Unfortunately, choosing the best subset is not an easy task as the number of subsets to be considered is large even for small instances of the problem.

There are various algorithms in the literature that address the problem of finding the MVE estimator. These algorithms can be divided into three

categories: Exact methods, heuristics, and metaheuristics. A direct enumeration algorithm in which all h-point subsets are investigated was proposed in [8]. This method is very slow and can only be useful for very small instances. Later, [1] proposed a branch and bound (B-and-B) algorithm which avoids investigating a large proportion of the subsets, nevertheless, still not able to solve large instances. The following random search is proposed by [27]: 1) Randomly generate a large number of subsets with $n + 1$ points; 2) calculate the MVEE of these subsets; 3) Deflate or inflate each of these ellipsoids to enclose at least h points and record their volumes. The ellipsoid with the smallest volume is chosen as the MVE estimator. This is a rough estimate and a large number of subsets need to be investigated for a good approximation. The Feasible Solution Algorithm (FSA), proposed in [15], starts with a random h-point subset and searches the neighborhood by swaping two points each time. It is also discussed that the algorithm should make approximately 5000 random starts to find a satisfactory estimate . Modified versions of the FSA algorithm are also discussed which use different methods to choose the swaps. Also, a lower bound is used to eliminate some subsets before calculating the minimum-volume enclosing ellipsoid. These algorithms are improved in [16] by using an easy-to-check condition during the initialization to eliminate some of the covered points if the initial sample covers more than h points. A much simpler algorithm was proposed by [23] based on EID values. The EID value of a point is an estimate of the contribution of the point to the determinant of the Fischer information matrix. EID values are known to be closely related to the volume of the minimum-volume enclosing ellipsoid as discussed in [24]. This algorithm is very fast but can fail to find a near-optimal solution. Hawkins' FSA algorithms and Poston's EID algorithm are combined by [13] to obtain better approximations. Various metaheuristics such as genetic algorithms, tabu search, and simulated annealing were applied in [36].

In this paper, we develop a 2-exchange heuristic and provide ideas that can be used in building a branch and bound algorithm for the MVE estimator. In this context, a 2-exchange is a local search method which produces a solution that is at least as good as every other solution that can be obtained by dropping one point from the current subset and adding another one. Hawkin's FSA algorithm is also a 2-exchange heuristic; nevertheless our algorithm uses new bounds that have not been considered before. We demonstrate that on a large set of problem instances the new algorithm (1) is very fast, (2) finds optimal or near optimal solutions and (3) improves the solutions obtained from the EID heuristic significantly. We build on the recent developments in the study of the Minimum Volume Enclosing Ellip-

soid problem and provide valuable insight for the MVE estimator problem. The new exchange algorithm proposed here should not be taken as a competitor to the existing algorithms. It is rather complementary in the sense that the new bounds developed in this paper can be incorporated into the existing methods to improve their performance without much computational burden.

3 CALCULATING MINIMUM VOLUME ENCLOSING ELLIPSOIDS

We start by describing in detail how to efficiently calculate the MVEE of a data set $\mathcal{X}$, using a first-order algorithm. We assume that the data set $\mathcal{X}$ and all subsets of $\mathcal{X}$ are centered around the origin. The MVEE problem for the set $\mathcal{X} = \{x_1, \ldots, x_m\}$ is written as

$$(\mathcal{P}) \qquad \begin{array}{lrcl} \min_{H>0} & f(H) & := & -\ln\det H \\ & x_i^T H x_i & \leq & n,\ i \in \mathcal{X}, \end{array}$$

and its dual as

$$(\mathcal{D}) \qquad \begin{array}{lrcl} \max_u & g(u) & := & \ln\det XUX^T \\ & e^T u & = & 1, \\ & u & \geq & 0, \end{array}$$

where $X := [x_1, \ldots, x_m]$, $U := \mathrm{Diag}(u)$, and e is a vector of ones in $\mathbb{R}^m$. The assumption on the centrality of the data set is without loss of generality. Given an arbitrary set $\hat{X} = \{\hat{x}_1, \ldots, \hat{x}_m\} \subset \mathbb{R}^n$, we can obtain a new set

$$X = \{[\hat{x}_1; 1], [-\hat{x}_1; -1], \ldots, [\hat{x}_m; 1], [-\hat{x}_m; -1]\} \subset \mathbb{R}^{n+1},$$

which has twice many points and centered around the origin. It is proved in [20] that MVEE of the original set is the intersection of the MVEE of the centralized set with the hyperplane $x_{n+1} = 1$.

Problem $(\mathcal{D})$ is the well-known statistical problem of finding a D-optimal design measure on the set $\mathcal{X}$ (See [5], [6], [9], [25], [28], [37], and [38]. A detailed proof of the duality of the two problems above can be found in [2]. The following optimality conditions are derived from this relationship and will be needed to develop algorithms below.

Lemma 3.1. *Let H^* and u^* be optimal solutions with respect to $(\mathcal{P})$ and $(\mathcal{D})$. Then $f(H^*) = g(u^*)$. Furthermore, the following conditions are necessary and sufficient:*

(a) $u_i^* > 0$ only if $x_i^T H^* x_i = n$; and

(b) $H^* = (XU^*X^T)^{-1}$.

The optimality conditions can be understood better from a geometric point of view. A feasible solution to problem $(\mathcal{D})$ is a vector u in the unit simplex in $\mathbb{R}^m$, which is also a probability distribution over the set of points in $\mathcal{X}$. There is an ellipsoid centered at the origin and defined by the shape matrix $H(u) := (XUX^T)^{-1}$ corresponding to u, which assigns an *ellipsoidal distance* (from the origin) to a point $x_k \in \Re^n$. This distance, defined as $\xi_k := x_k^T(XUX^T)^{-1}x_k$, is sometimes also referred to as the Mahalabolis distance of x_k and plays an important role in detecting outliers: The MVE estimator is actually a set of points (a subset of the given data set) and a probability distribution over them, which together define a distance function, which in turns identifies the outliers. If for a feasible solution u, the points in $\mathcal{X}$ with positive weight also happen to lie on the surface of the ellipsoid, i.e., the ellipsoidal distances of these points are all exactly equal to n, then the problems $(\mathcal{P})$ and $(\mathcal{D})$ are both solved to optimality. In practice, it is enough to find approximate solutions to both problems, which are defined as follows:

Definition 3.1. *Given a positive ϵ, a dual feasible point u satisfies the* ϵ-approximate optimality conditions *or it is an* ϵ-approximate optimal solution *if*

(a) $\xi_i(u) \leq (1+\epsilon)n$ *for all* i, *and*

(b) $\xi_i(u) \geq (1-\epsilon)n$ *whenever* $u_i > 0$.

It is straightforward to show that the objective function value of an ϵ-approximate optimal solution is at most $\ln(1+\epsilon)$ away from the optimal objective function value of $(\mathcal{D})$. Such a solution can be found efficiently by a customized Frank-Wolfe type algorithm that maximizes a linearization of the objective function at each iteration (see [10] for the Frank-Wolfe method). Various such algorithms were proposed and analysed by statisticians since the 1960s. The first algorithms appeared in [9] and [38], which only considered iterations that increase the weight of a chosen coordinate. These methods were very slow, and soon were improved by [6] by allowing the weight of the chosen coordinate to also decrease, referred often as the *away steps*. Recently, these algorithms were analysed rigorously by the optimization community, starting with [20] that provided the time complexity analysis of the algoritm without the away steps. Later, [21] proposed to start this algorithm at a carefully chosen initial solution so that the final solution has significantly smaller number of nonzero weights than that of [20]. This variant is much faster than the one without the initialization

scheme both in theory and practice. They also introduced the concept of *core sets*, and were able to provide upper bounds on the number of nonzero weights of the solution obtained by the algorithm. The analysis was extended by [32] to also include away steps. Hence, rigorous complexity results for an algorithm equivalent to that of [6] were obtained. During this period, [14] proved a simple condition that characterizes the points that have zero weight in the optimal solution. Such points do not contribute to the optimal solution and therefore can be eliminated from $\mathcal{X}$ immediately. Incorporating this condition to any Frank-Wolfe type algorithm is straightforward and decreases the computational time dramatically (see Chapter 2 in [4]). In addition, [2] proved that Frank-Wolfe type algorithms with away steps and exact line search have favorable local convergence properties and therefore can be used to obtain very acurate solutions. This state-of-the art algorithm is provided below as Algorithm 1. Just to give a quick idea to the interested reader, this variant can find a 10^{-7}-approximate solution for a randomly generated problem instance with 500.000 points in $\mathbb{R}^{50}$ in a few minutes on an ordinary laptop (same one used in our experiments in Section 6). In order to achieve this, the algorithm should be implemented carefully, especially when updating the ξ values in Step 4 (see [4] for implementation details).

In contrast to Frank-Wolfe type algorithms, multiplicative algorithms update all weights simultaneously. Several early versions were developed, including [31] and [29], and more recent and faster versions exist, such as [41] and [39]. The convergence properties of the original algorithm are analyzed in [22]. A relatively recent survey on multiplicative algorithms together with a new multiplicative approach can also be found in [33]. Although there does not exist a thorough comparative study between Frank-Wolfe type algorithms and the variants of the multiplicative algorithm, the first-order method provided in this paper is more promising for large scale problem instances. In addition, for the main goal of the paper, which is to develop efficient exhange heuristics for the MVE estimator problem, Algorithm 1 is very suitable: The 2-way exchanges performed by the heuristics below mimic the add and away steps of this algorithm, therefore, it is easy to incorporate this algorithm as a subroutine to the exchange algorithms developed below.

Another interesting and modern approach is using semidefinite programming reformulations as discussed in [35]. This approach fails to solve large problems due to the lack of efficient solvers.

Algorithm 1.
Input: *$\mathcal{X} \in \mathbb{R}^{n\times m}$, $\epsilon > 0$, and $N_E > 0$.*
Step 0. *Use the algorithm of Kumar and Yıldırım [21] to obtain an initial feasible u. Set $iter = 0$ and compute $\xi(u)$.*
Step 1. *Find $j := \arg\max_l\{\xi_l(u) - n\}$, $i = \arg\min_l\{\xi_l(u) - n : u_l > 0\}$.*
If *$\xi_j(u) - n \le \epsilon n$ and $\xi_i(u) - n \ge -\epsilon n$,*
STOP: *u satisfies the ϵ-approximate optimality conditions.*
Else, *$iter = iter + 1$,*
if *$mod\{iter, N_E\} = 0$,* ***then*** *find $\mathcal{X}_E := \left\{x_i : \xi_i(u) < n(1 + \frac{\epsilon}{2} - \frac{\sqrt{\epsilon(4+\epsilon-4/\epsilon)}}{2})\right\}$,*
update $\mathcal{X} = \mathcal{X} - \mathcal{X}_E$, and erase the corresponding coordinates of u and $\xi(u)$.
if *$\xi_j(u) - n > n - \xi_i(u)$,* ***go to*** *Step 2;* ***else, go to*** *Step 3.*
Step 2. *Update $u := (1-\tau)u + \tau e_j$, where $\tau = \frac{\xi_j(u)/n-1}{\xi_j(u)-1} > 0$.* ***Go to*** *Step 4.*
Step 3. *Update $u := (1-\tau)u + \tau e_i$, where now $\tau = \max\{\frac{\xi_i(u)/n-1}{\xi_i(u)-1}, -\frac{u_j}{1-u_j}\} < 0$.*
Step 4. *Update $\xi(u)$ and* ***go to*** *Step 1.*
Output: *An ϵ-approximate MVEE for $\mathcal{X}$.*

4 NEW BOUNDS FOR THE MVE ESTIMATOR PROBLEM

We assume that the data set $\mathcal{X}$ and all subsets of $\mathcal{X}$ are centered around the origin as before. In addition, let $\mathcal{A} \subseteq \mathcal{I}$ be a subset of indices. The MVEE problem for the set $\mathcal{X}_\mathcal{A} := \{x_i \in \mathcal{X} : i \in \mathcal{A}\} \subseteq \mathcal{X}$ can be written as

$$(\mathcal{P}_\mathcal{A}) \qquad \begin{array}{lrcl} \min_{H>0} & f_\mathcal{A}(H) & := & -\ln\det H \\ & x_i^T H x_i & \le & n,\ i \in \mathcal{A}, \end{array}$$

and its dual as

$$(\mathcal{D}_\mathcal{A}) \qquad \begin{array}{lrcl} \max_u & g_\mathcal{A}(u) & := & \ln\det XUX^T \\ & e^T u & = & 1, \\ & u & \ge & 0, \\ & u_i & = & 0,\ i \notin \mathcal{A}, \end{array}$$

where $X := [x_1, \ldots, x_m]$, $U := \mathrm{Diag}(u)$, and e is a vector of ones in $\mathbb{R}^m$.

Definition 4.1. *A feasible solution for the MVE estimator problem is any subset $\mathcal{A} \subset \mathcal{I}$ such that $|\mathcal{A}| \ge h$. Its value is equal to $f_\mathcal{A}(H^*)$, where H^* is the optimal solution for $(\mathcal{P}_\mathcal{A})$.*

We need to solve $(\mathcal{P}_{\mathcal{A}})$ and $(\mathcal{D}_{\mathcal{A}})$ in order to evaluate the value of each subset. These values can be calculated by the algorithms mentioned in Section 3. Nevertheless, we should avoid solving this problem from scratch as much as we can. The following observations will be useful in determining which subsets to investigate or eliminate. When two subsets lie in the neighborhood of each other in the search space, (i.e., one subset can be obtained by adding or removing a single point, or exchanging two points), solving the MVEE problem for one of them gives information about the value of the other one as discussed next. For simplicity we will assume that the data points are indexed so that $\mathcal{A} = \{1, 2, \ldots, |\mathcal{A}|\}$. In addition, the points to be added, removed, or exchanged are chosen among $x_{|\mathcal{A}|}$ and $x_{|\mathcal{A}|+1}$.

Lemma 4.1. *Given $\mathcal{A} = \{1, \ldots, k-1\}$ and $\hat{\mathcal{A}} = \mathcal{A} \cup \{k\}$, let H, $\hat{H}$, u, and $\hat{u}$ be optimal for $(\mathcal{P}_{\mathcal{A}})$, $(\mathcal{P}_{\hat{\mathcal{A}}})$, $(\mathcal{D}_{\mathcal{A}})$, and $(\mathcal{D}_{\hat{\mathcal{A}}})$, respectively. Also let $\xi_k := x_k^T (XUX^T)^{-1} x_k$. If $\xi_k \leq n$, then*

$$f_{\hat{\mathcal{A}}}(\hat{H}) = f_{\mathcal{A}}(H).$$

Otherwise,

$$f_{\hat{\mathcal{A}}}(\hat{H}) - f_{\mathcal{A}}(H) \geq n \ln\left(\frac{\xi_k}{n}\right) - (n-1) \ln\left(\frac{\xi_k - 1}{n-1}\right).$$

Proof. When $\xi_k \leq n$, x_k is already enclosed by MVEE($\mathcal{X}_{\mathcal{A}}$); hence MVEE($\mathcal{X}_{\mathcal{A}}$) is also MVEE($\mathcal{X}_{\hat{\mathcal{A}}}$).

Assume $\xi_k > n$. For any $0 \leq \tau < 1$, $u_+ := (1-\tau)u + \tau e_k$ is feasible w.r.t $(\mathcal{D}_{\hat{\mathcal{A}}})$, where e_k is the k^{th} unit vector.

The Sherman–Morrison-Woodbury formula gives

$$(XU_+X^T)^{-1} = \frac{1}{1-\tau}\left[(XUX^T)^{-1} - \frac{\tau (XUX^T)^{-1} x_k x_k^T (XUX^T)^{-1}}{1 - \tau + \tau \xi_k(u)}\right]$$

and hence

$$\det XU_+X^T = (1-\tau)^{n-1}[1 - \tau + \tau\xi_k(u)] \det XUX^T. \tag{1}$$

Using this, we have

$$\begin{aligned} f_{\hat{\mathcal{A}}}(\hat{H}) &= g_{\hat{\mathcal{A}}}(\hat{u}) \geq g_{\hat{\mathcal{A}}}(u_+) \\ &= (n-1)\ln(1-\tau) + \ln(1 - \tau + \tau\xi_k(u)) + g_{\mathcal{A}}(u) \\ &= (n-1)\ln(1-\tau) + \ln(1 - \tau + \tau\xi_k(u)) + f_{\mathcal{A}}(H). \end{aligned}$$

The rest of the proof follows from the fact that $\tau = \frac{\xi_k/n-1}{\xi_k - 1}$ maximizes the right-hand side of this inequality. □

Lemma 4.2. *Given $\mathcal{A} = \{1, \ldots, k\}$ and $\hat{\mathcal{A}} = \mathcal{A} - \{k\}$, let H, $\hat{H}$, u, and $\hat{u}$ be optimal for $(\mathcal{P}_{\mathcal{A}})$, $(\mathcal{P}_{\hat{\mathcal{A}}})$, $(\mathcal{D}_{\mathcal{A}})$, and $(\mathcal{D}_{\hat{\mathcal{A}}})$, respectively. If $u_k = 0$, then*

$$f_{\hat{\mathcal{A}}}(\hat{H}) = f_{\mathcal{A}}(H).$$

Otherwise,

$$f_{\hat{\mathcal{A}}}(\hat{H}) - f_{\mathcal{A}}(H) \geq -n \ln(1 - u_k) + \ln(1 - nu_k).$$

Proof. When $u_k = 0$, u is feasible and also optimal for $(\mathcal{D}_{\hat{\mathcal{A}}})$; hence MVEE$(\mathcal{X}_{\mathcal{A}})$ is also MVEE$(\mathcal{X}_{\hat{\mathcal{A}}})$.

Suppose now that $u_k > 0$. Since u is optimal w.r.t $(\mathcal{D}_{\mathcal{A}})$, $u_k > 0$ implies that $\xi_k = n$. Consider the update $u_+ := (1 - \tau)u + \tau e_k$ where $\tau = \frac{-u_k}{1-u_k}$. Since $(u_+)_k = 0$, u_+ is feasible w.r.t $(\mathcal{D}_{\hat{\mathcal{A}}})$. Using (1), we have

$$\begin{aligned} f_{\hat{\mathcal{A}}}(\hat{H}) &= g_{\hat{\mathcal{A}}}(\hat{u}) \geq g_{\hat{\mathcal{A}}}(u_+) \\ &= (n-1)\ln(1/(1-u_k)) \\ &\quad + \ln((1 - \xi_k u_k)/(1 - u_k) + g_{\mathcal{A}}(u) \\ &= -n \ln(1 - u_k) + \ln(1 - nu_k) + f_{\mathcal{A}}(H). \end{aligned}$$

□

Assuming that we have found the MVEE of a particular subset $\mathcal{X}_{\mathcal{A}}$, Lemmas 4.1 and 4.2 provide lower bounds on the value of the subsets that can be obtained by adding an element to $\mathcal{A}$ or removing one from $\mathcal{A}$. These lower bounds can be useful in a branch-and-bound scheme to eliminate some subsets even without solving the MVEE problem for them. The following lemma takes a further step ahead and derives a lower bound on the value of the subsets that can be obtained by adding a point and removing another one from the current subset $\mathcal{A}$ simultaneously. This will be useful in developing the 2-exchange heuristic in the following section.

Lemma 4.3. *Given two sets $\mathcal{A} = \{1, \ldots, k-1, k\}$ and $\hat{\mathcal{A}} = \{1, \ldots, k-1, k+1\}$, let H, $\hat{H}$, u, and $\hat{u}$ be optimal for $(\mathcal{P}_{\mathcal{A}})$, $(\mathcal{P}_{\hat{\mathcal{A}}})$, $(\mathcal{D}_{\mathcal{A}})$, and $(\mathcal{D}_{\hat{\mathcal{A}}})$, respectively. Then we have*

$$\begin{aligned} f_{\hat{\mathcal{A}}}(\hat{H}) - f_{\mathcal{A}}(H) &\geq LB_{k,k+1} \\ &:= (n-1)\ln(1-\tau_2) - n\ln(1-u_k) + \ln(1 - u_k\xi_k - \tau_2 + \tau_2\delta), \end{aligned}$$

where $\xi_{k(k+1)} := x_k(XUX^T)^{-1}x_{k+1}$,

$$\delta = u_k\xi_k + (1-u_k)\xi_{k+1} - u_k(1-u_k)\left(\xi_k\xi_{k+1} - \xi_{k,k+1}^2\right),$$

and $\tau_2 = \max\{0, \frac{\delta - n + nu_k\xi_k + 1 - u_k\xi_k}{n\delta}\}$.

Proof. Consider

$$u_+ := (1 - \tau_1 - \tau_2)u + \tau_1 e_k + \tau_2 e_{k+1}. \tag{2}$$

If we choose

$$\tau_1 = \frac{u_k(\tau_2 - 1)}{1 - u_k}, \tag{3}$$

then $(u_+)_k = 0$. Then u_+ is a feasible solution for $(\mathcal{D}_{\hat{\mathcal{A}}})$. We can calculate the dual objective function value at this point easily as follows. We have

$$XU_+X^T = (1 - \tau_1 - \tau_2)(XUX^T) + \tau_1 x_k x_k^T + \tau_2 x_{k+1} x_{k+1}^T,$$

which leads to

$$\det(XU_+X^T) = \det(XUX^T)(1 - \tau_1 - \tau_2)^n(1 + \tfrac{\tau_1}{1-\tau_1-\tau_2}\xi_k + \tfrac{\tau_2}{1-\tau_1-\tau_2}\xi_{k+1} + \tfrac{\tau_1\tau_2}{(1-\tau_1-\tau_2)^2}(\xi_k\xi_{k+1} - \xi_{k,k+1}^2)).$$

Plugging in (3) and taking logarithms, we get

$$\ln\det(X\hat{U}_+X^T) = \ln\det(XUX^T) + (n-1)\ln(1-\tau_2) - n\ln(1-u_k) + \ln(1 - u_k\xi_k + \tau_2\delta).$$

The rest of the proof follows since $\tau_2 = \max\{0, \frac{\delta - n + nu_k\xi_k + 1 - u_k\xi_k}{n\delta}\}$ maximizes the right-hand side of this inequality while keeping u_+ feasible. □

Note that Lemma 4.3 deduces to Lemma 4.1 for $u_k = 0$ and to Lemma 4.2 for $\tau_2 = 0$ as expected.

5 A 2-EXCHANGE HEURISTIC

In this section, we develop a 2-exchange heuristic which is capable of finding good solutions quickly even for large instances of the problem. The heuristic can roughly be summarized in three steps:

1. Find an initial feasible solution and evaluate its value.
2. Select an exchange that will improve the objective function.
3. Perform the exchange.

The last two steps are repeated until there does not exist an exchange that improves the best solution found so far. It is unlikely that a single run of the algorithm would return a local optimum. It should be repeated many times with many different initial solutions in order to guarantee a good quality solution. As we will discuss below, the quality (by quality

we mean the speed of the method as well as the quality of the solutions produced) of the heuristic is highly dependent on the initial solutions, the number of times the algorithm is called, and the selection of the points to be swapped at each iteration. Lower bounds on the value of the candidate subsets developed in the previous section are useful in decreasing the number of exchanges performed by the heuristic. Running time can further be reduced by careful implementation, especially when performing the exchange in Step 3. We will elaborate on these details in the following subsections.

5.1 Finding a Feasible Initial Solution

Here we introduce several methods to find an initial feasible solution. While some of the methods are elegant but computationally expensive, others are crude but cheap as we will illustrate in detail in the next section.

a. Ellipsoidal peeling

Ellipsoidal peeling starts with the MVEE of the whole data set X and 'peels' this ellipsoid by discarding one of the points on its boundary at each iteration until only h points remain. Lemma 4.2 provides an estimate (lower bound) on the volume of the new MVEE when one of the points is discarded from the current subset. Since the lower bound is a decreasing function of the weight of the point being discarded, a point on the boundary with maximum weight is left outside. This method starts with covering all points and generates ellipsoids which cover $m-1, m-2, \ldots,$ and h points as it proceeds. It can be summarized as follows:

Algorithm 2 (Ellipsoidal Peeling)**.**
Input: $X \in \mathbb{R}^{n\times m}$, $h \in \{1, \ldots, m\}$.
Step 1. *Set* $\mathcal{A} = \mathcal{I}$.
Step 2. If $|\mathcal{A}| = h$, ***STOP:*** *Output* $\mathcal{A}$.
Else, *find an optimal solution* u *for* $(\mathcal{D}_{\mathcal{A}})$.
Step 3. *Find* $j = \arg\max_i u_i$. *Set* $\mathcal{A} = \mathcal{A} - \{j\}$.
Go to *Step 2.*

Although ellipsoidal peeling is an elegant idea, it has three major disadvantages. First, it is an expensive algorithm because we need to solve $m-h$ instances of the MVEE problem. Second, it is a deterministic algorithm which is not useful in designing a local search heuristic. Third, it does not produce good quality solutions. Nevertheless, when combined with other

methods such as Poston's EID method, it becomes very fast and produces remarkably good quality solutions as we will discuss in the next section. b. Ellipsoidal ordering

The following algorithm calculates the MVEE of the whole data set, orders points with respect to their ellipsoidal distance (defined using the MVEE), and picks the points with the h smallest distances as the initial solution. It is much faster than ellipsoidal peeling but it is still a deterministic algorithm.

Algorithm 3 (Ellipsoidal Ordering)**.**
Input: $X \in \mathbb{R}^{n\times m}$, $h \in \{1,\dots,m\}$.
Step 1. *Find an optimal solution u for $(\mathcal{D}_{\mathcal{A}})$.*
Step 2. *Sort $\xi_{i_1}(u) \le \xi_{i_2}(u) \le \cdots \le \xi_{i_m}(u)$.*
STOP: *Output $\mathcal{A} = \{i_1,\dots,i_h\}$.*

c. Random ellipsoidal peeling/building

The following algorithm randomizes the ellipsoidal peeling algorithm. It is useful in local search and usually very fast compared to its deterministic version.

Algorithm 4 (Random E. Peeling/Building)**.**
Input: $X \in \mathbb{R}^{n\times m}$, $h \in \{1,\dots,m\}$.
Step 1. *Choose a subset $\mathcal{A}$ randomly s.t.*
$\mathcal{A} \subset \mathcal{I}$, $|\mathcal{A}| = n+1$, *and* $X_{\mathcal{A}}$ *spans* $\mathbb{R}^n$.
Step 2. *Find an optimal solution u for $(\mathcal{D}_{\mathcal{A}})$.*
Step 3. *Let $\mathcal{A} = \{i : \xi_i(u) \le n\}$.*
Step 4. If $|\mathcal{A}| = h$, ***STOP:*** *Output $\mathcal{A}$.*
elseif $|\mathcal{A}| < h$, *set $\mathcal{A} = \mathcal{A} \cup \{j\}$ where,*
$j = \arg\min\{\xi_i(u) : \xi_i(u) > n\}$;
else *set $\mathcal{A} = \mathcal{A} - \{j\}$, where $j = \arg\max_i u_i$.*
Step 5. Go to *Step 2.*

d. Random ellipsoidal ordering

Similarly, we can also randomize the ellipsoidal ordering algorithm as follows.

Algorithm 5 (Random E. Ordering)**.**
Input: $X \in \mathbb{R}^{n\times m}$, $h \in \{1, \ldots, m\}$.
Step 1. *Choose a subset* $\mathcal{A}$ *randomly s.t.*
$\mathcal{A} \subset \mathcal{I}$, $|\mathcal{A}| = (n+1)$, *and* $\mathcal{X}_{\mathcal{A}}$ *spans* $\mathbb{R}^n$.
Step 2.*Find an optimal solution* u *for* $(\mathcal{D}_{\mathcal{A}})$.
Step 3. *Sort* $\xi_{i_1}(u) \leq \xi_{i_2}(u) \leq \cdots \leq \xi_{i_m}(u)$.
Output $\mathcal{A} = \{i_1, \ldots, i_h\}$.

5.2 Selecting Points for the Exchange

At each iteration of the heuristic, we move from the current solution (a subset $\mathcal{A}$ with h points) to a new one in its neighborhood by making an exchange between a covered point $i \in \mathcal{A}$ and an uncovered point $j \notin \mathcal{A}$. Although there are $h \times (m-h)$ possible exchanges, considering exchanges between points on the boundary of the MVEE of the current subset and the $(m-h)$ uncovered ones is enough. This is due to the fact that exchanging a point in the interior of the current ellipsoid will not lead to a subset whose MVEE has smaller volume. Given a set of points in $\mathbb{R}^n$, it is shown in [19] that there exists a subset of at most $n(n+1)/2$ points, which can be referred to as the *core set*, or the *0-core set* to be precise, such that the MVEE of the smaller subset is equal to the MVEE of the whole set. In addition, when the MVEE is calculated approximately as in Definition 3.1, then [21] shows that there exists an *ϵ-core set* of cardinality at most $O(n\log(n)/\epsilon))$, which is the subset of points whose approximate MVEE has the same volume with the approximate MVEE of the whole set. Obviously, neither of these bounds give an upper bound on the number of points on the boundary of the MVEE as all of the h covered points may be on the boundary. Nevertheless, this is a highly unlikely situation. According to the author's experience, the number of points on the boundary of an MVEE for a typical dataset is usually limited by $O(n)$. Therefore, the number of possible exchanges is usually around $O(n) \times (m-h)$ instead of $h \times (m-h)$.

For each candidate pair of points to be exchanged, the lower bound (on the volume of the candidate subset) from Lemma 4.3 is used in order to eliminate nonpromising exchanges. Once we obtain a candidate exchange x_i and x_j whose lower bound is lower than the current objective function value, we solve the corresponding MVEE problem for the candidate set $\hat{\mathcal{A}} = \mathcal{A} - \{i\} \cup \{j\}$. If the volume of MVEE($\hat{\mathcal{A}}$) is smaller than that of MVEE($\mathcal{A}$), we record this value. Three different versions of the algorithm can be used:

i. BEST exchange version: evaluates all possible exchanges and uses the best one.

ii. FIRST exchange version: picks the first exchange that provides an MVEE with a smaller value.

iii. BEST BOUND version: calculates a lower bound (from Lemma 4.3) for all possible exchanges, picks the best, and calculates its value. This exchange is used if the value of the candidate solution is better than the value of the current solution, otherwise the exchange with the second (or third, etc.) smallest lower bound is considered.

All versions of the heuristic are faster than the previously reported exchange heuristics in the literature because of the decreased number of exchanges considered at every iteration and the improvement on the lower bound. In addition, we are capable of evaluating each candidate exchange faster because we don't solve the corresponding MVEE problem from scratch but start at a feasible solution obtained from the current optimal solution for MVEE($\mathcal{A}$) using the best step size as discussed in Lemma 4.3. In addition, we stop evaluating a subset when the objective function value exceeds the best solution found so far.

We can summarize the heuristic as follows:

Algorithm 6 (2-Exchange Heuristic: FIRST)**.**
Input: $X \in \mathbb{R}^{n\times m}$, $h \in \{1,\dots,m\}$.
Step 1. *Obtain an $\mathcal{A}$ from Algorithms 2-5.*
Let u be an optimal solution for $(\mathcal{D}_{\mathcal{A}})$, and H be an optimal solution for $(\mathcal{P}_{\mathcal{A}})$.
Set $Bvol = f_{\mathcal{A}}(H)$ and $opt = 1$.
Step 2. *For all $k : u_k > 0$ and all $j : j \notin \mathcal{A}$,*
calculate LB_{kj} as in Lemma 4.3 with j replacing $k+1$, and let $\hat{\mathcal{A}} = \mathcal{A} - \{k\} \cup j$.
If *$LB_{kj} < Bvol$,*
find optimal $\hat{H}$ and $\hat{u}$ for $(\mathcal{P}_{\hat{\mathcal{A}}})$ and $(\mathcal{D}_{\hat{\mathcal{A}}})$.
If *$f_{\hat{\mathcal{A}}}(\hat{H}) < Bvol$, set $Bvol = f_{\hat{\mathcal{A}}}(\hat{H})$,*
$c\mathcal{A} = \hat{\mathcal{A}}$, $cH = \hat{H}$, $cu = \hat{u}$ and $opt = 0$.
Step 3. If *$opt = 0$, then set $\mathcal{A} = c\mathcal{A}$, $H = cH$,*
$u = cu$ and $opt = 1$. ***Go to*** *Step 2.*
Else*, output $Bvol$ and $\mathcal{A}$.*

6 COMPUTATIONAL RESULTS

In this section, we investigate the four main questions below: 1) How much does the quality of the output depend on the initial feasible solution. 2) How effective is the local search, i.e., how much does the initial solution improve with the exchange heuristic. 3) How many random starts are needed by the heuristic to obtain reasonable solutions. 4) How do the outliers affect the computational time and quality of the proposed heuristics. For this analysis we use two types of synthetic data. The first type of synthetic datasets are used in Sections 6.1, 6.2, and 6.3. These are generated from 4 or more independent random multivariate normal distributions as in [2] and [30]. These datasets have four or more clusters of data points, which can be encountered in practice. Existence of several clusters make these datasets more challenging than the typical data sets used in literature such as [34]. These datasets are generated using an independent multivariate normal distribution, which is contaminated with outliers. In particular, the datasets used in [34] have 5% or 30% contamination. We use same method to obtain our second set of synthetic datasets and discuss the effect of contamination in Section 6.4 below. The algorithms used for the tests are developed in C++[1] using the GSL library. The experiments are conducted on a Macbook Air with a 1.7 GHz Intel Core i7 processor and 8 GB memory. In all experiments we have set $\epsilon = 10^{-2}$ and $N = 1$ unless stated otherwise.

6.1 Effect of the Initial Feasible Solution

We have first tested the exchange heuristic with various initialization methods. We have used small sized problems (from $(n, m) = (2, 20)$ to $(n, m) = 5, 30)$) that we can solve exactly using a branch and bound technique in order to be able to calculate the optimality gap for each problem. The average results (over a sample of thousand instances) are presented in Table 1. The first three columns demonstrate the dimensions of the problem. The remaining columns correspond to the relative percentage errors in the objective function value of the heuristic solution, where the initial feasible solution is obtained by the ellipsoidal peeling method (Column 4: Algorithm 2); a hybrid of the ellipsoidal ordering and peeling methods (Column 5: Algorithm 3 followed by Steps 2 and 3 of Algorithm 2); a hybrid of the EID

[1]The main code for Algorithm 1 is provided in `https://code.google.com/p/minimum-volume-enclosing-ellipsoid/`. The exchange heuristic used in tests together with a branch-and-bound algorithm that finds the optimal solution are also available on request together with the source code for generating datasets.

n	m	h	EP	EO-EP	EID-EP	REP
2	20	12	22	7	0	7
2	30	17	30	13	1	10
2	50	27	35	17	1	10
3	20	12	24	6	5	9
3	30	17	31	14	5	11
3	50	27	37	15	4	11
5	20	13	13	6	3	5
5	30	18	13	4	3	9

Table 1: Average relative error of the heuristic solution with various initialization methods (in percents).

and the ellipsoidal peeling methods (Column 6: EID algorithm followed by Steps 2 and 3 of Algorithm 2); and the random peeling method (Column 7: Algorithm 4). We more a more detailed version of this table with minimum and maximum values as well as the standard deviations in Table 2.

It is remarkable that the quality of the solution depends highly on the initial solution. It seems that the EID heuristic followed by an ellipsoidal peeling is a reliable strategy to use, especially when the computational resources are limited and only few starting points can be tried out. This is generally the case for large problem instances. We have also observed that when started with this particular strategy, the heuristic stops in much shorter time than the rest of initialization strategies, usually less than half time of its closest competitor.

6.2 Effectiveness of the neighborhood search

In this section we compare the quality of the solutions obtained by the EID method and those obtained by the 2-exchange algorithm (Algorithm 6), which uses the initial feasible solution from the EID method. The first set of results presented in Table 3, correspond to small problem instances. The first three columns demonstrate the dimensions. The next two columns give the corresponding average objective function values from the EID heuristic and the 2-Exchange algorithm which uses the solution obtained by the EID heuristic as a starting point. The next column provides the average optimal value obtained by the enumeration algorithm. The last two columns provide the time required by the exchange heuristic and the enumeration algorithm. We provide more statistics about these experiments such as the

	n	m	h	EP	EO-EP	EID-EP	REP
mean	2	20	12	22	7	0	7
min				0	0	0	0
max				615	165	32	168
std				38	17	2	19
mean	2	30	17	30	13	1	10
min				0	0	0	0
max				514	445	88	441
std				41	30	4	26
mean	2	50	27	35	17	1	10
min				0	0	0	0
max				824	816	55	213
std				44	37	3	22
mean	3	20	12	24	6	5	9
min				0	0	0	0
max				682	148	148	271
std				44	49	14	21
mean	3	30	17	31	14	5	11
min				0	0	0	0
max				752	759	98	222
std				49	41	13	22
mean	3	50	27	37	15	4	11
min				0	0	0	0
max				338	215	209	213
std				36	26	15	20
mean	5	20	13	13	6	3	5
min				0	0	0	0
max				140	112	87	112
std				17	12	9	11
mean	5	30	18	13	4	3	9
min				0	0	0	0
max				148	73	43	169
std				19	14	7	16

Table 2: Average, minimum, and maximum relative errors obtained by different initializations

n	m	h	EID	2 EX	OPT	Time2X (ms)	TimeBB (ms)
2	20	12	15.13	13.73	12.32	0.99	99.02
2	30	17	14.98	13.73	12.05	1.55	4206.85
3	20	12	20.82	17.86	15.95	1.91	132.54
3	30	17	21.13	18.30	15.68	3.28	6449.11
5	20	13	31.53	26.86	24.37	2.95	135.71
5	30	18	31.76	26.37	23.28	5.47	6483.03

Table 3: Average objective function value for small datasets

minimum and the maximum values as well as the standard deviations in Table 4.

We observe that the exchange heuristic improves the objective function value significantly over the EID heuristic. It is remarkable how fast we are able to find good solutions for these instances. We also observe that the exchange heuristic scales much better than the branch and bound algorithm. The worse case performance of the branch and bound algorithm can be extremely bad as given in Table 4. We have also tested larger instances with parameters varying from $(n, m) = (5, 100)$ to $(n, m) = (20, 500)$, which are reported in Table 5 and Table 6 in more detail. For these instances, it is not possible to find the optimal objective function value with the current state of the art methods. We still observe that the exchange heuristic improves the EID solution significantly. More importantly, the exchange heuristic is able to provide good solutions for such large instances in short amount of time even for cases when the initial solution is quite poor as often is the case with the EID heuristic. Note that the exchange heuristic used in this experiment is deterministic, it starts with the EID heuristic, so it cannot be started at many random points. We demonstrate here how much it can improve a single starting point such as EID start. If more accurate solutions are needed, then the exchange heuristic should be applied to many random starts for example using the random ellipsoidal peeling method discussed above. We investigate this further next.

6.3 Effect of the number of random starts

For a set of problem sizes, we have run the heuristic $N = 1, 10, 100, 1000$ times, where in each run an initial feasible solution is generated using the random ellipsoidal peeling method presented in Table 7 provides details of the experiment. In particular, the first three columns demonstrate the

	n	m	h	EID	2EX	OPT	Time2EX(ms)	TimeBB(ms)
mean	2	20	12	15.13	13.73	12.32	0.99	99.02
std				2.78	3.31	3.12	0.62	133.48
min				1.36	-0.82	-0.82	0.19	0.58
max				21.11	20.39	19.81	4.45	1483.05
mean	2	30	17.00	14.98	13.73	12.05	1.55	4206.85
std				2.84	3.36	3.14	1.01	6724.75
min				2.15	2.00	0.34	0.27	1.69
max				20.57	20.56	19.76	10.13	44356.00
mean	3	20	12.00	20.82	17.86	15.95	1.91	132.54
std				4.48	4.87	4.21	1.10	130.98
min				3.79	1.51	-1.53	0.30	3.31
max				28.52	28.48	25.75	10.50	945.47
mean	3	30	17.00	21.13	18.30	15.68	3.28	6449.11
std				4.36	4.90	4.06	2.07	10334.72
min				4.38	3.44	1.99	0.51	12.49
max				29.90	27.93	25.21	16.81	87477.60
mean	5	20	13.00	31.53	26.86	24.37	2.95	135.71
std				7.00	7.01	6.38	1.45	87.39
min				0.52	-1.03	-1.70	0.53	14.83
max				44.88	40.29	38.35	12.40	668.95
mean	5	30	18.00	31.76	26.37	23.28	5.47	6483.03
std				7.33	7.20	6.36	2.46	8459.12
min				-5.12	-7.71	-8.25	0.95	68.87
max				45.66	44.17	38.18	16.86	71808.40

Table 4: Average, minimum, and maximum objective function values for small datasets with standard deviations

dimensions as usual. The next four columns provide the number of problem instances (out of the 1000) that is solved by the heuristic with 1,10,100, and 1000 random starts, respectively. It is reported by [15] that the FSA algorithm requires around 5000 random starts for a relatively small data set ($n = 3, m = 28, h = 16$) to guarantee a good solution. Our tests suggest the heuristics developed in this paper require significantly less number of random starts to find the optimal solution. In particular, for the largest problem size we have tried so far, 100 starts were enough to find the optimal solution for 70% of the instances, which is very encouraging.

n	m	h	EID	2 EX	Time (ms)
5	100	53	32.68	27.97	37.89
5	200	103	32.91	29.35	119.11
5	500	253	32.68	30.15	606.32
10	100	56	60.54	51.23	82.49
10	200	106	59.59	51.54	303.07
10	500	256	58.18	52.10	1663.19
20	100	61	115.58	102.65	132.63
20	200	111	116.88	105.92	500.42
20	500	261	112.98	104.00	3252.19

Table 5: Average objective function value for large datasets

6.4 Effect of outliers

The MVE estimator is a tool to identify outliers, therefore, it is natural to test the algorithms with data sets that have a certain ratio of outliers. We generate random datasets similar to those in [34]. We first generate m data points in $\mathbb{R}^n$ from the multivariate standard normal distribution for $n = 3$ and $m = 30$. Then we shift the mean of 5% or 30% of the observations up to 7 units. We provide the average, minimum, and maximum objective function values of three algorithms, the Branch and Bound Algorithm, the 2 Exchange Heuristic, and the EID Heuristic in Figure 1. The results are averaged over 1000 instances. The standard deviation for all three algorithms is around 0.6. We observe that although both heuristics can perform quite bad compared to the optimal solution on some rare instances, especially with highly contaminated data sets, on average the heuristics provide a reasonable alternative. We also observe that the exchange heuristic provides close to optimal solutions on the average and improves the EID heuristic.

The effect of contamination on the running time of the branch and bound algorithm and the exchange heuristic is quite dramatic as observed in Figure 2, where we report the average CPU times required by the two algorithms over the same 1000 instances as above. As the dataset gets more contaminated the problem becomes much easier to solve. This is expected since the bounds on the volume used by the heuristic would be more useful in the contaminated cases, where the exchange heuristic can eliminate the exchanges including the outlier points directly without evaluating the corresponding subsets.

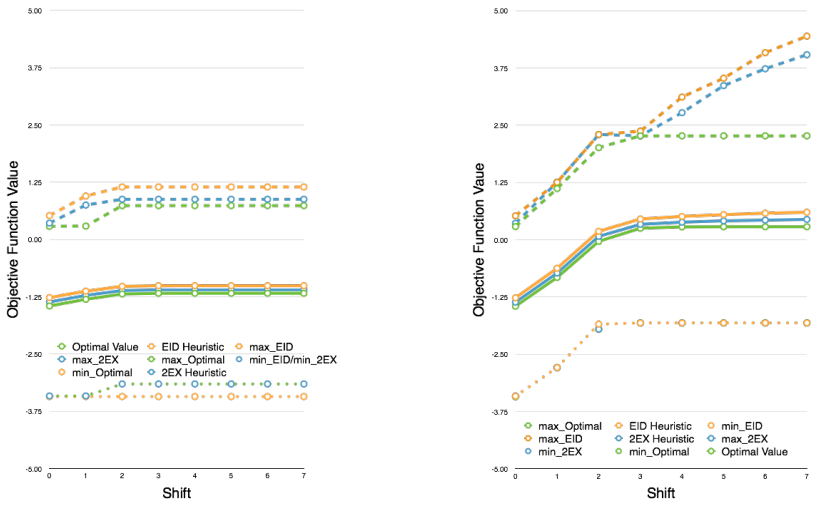

Figure 1: Average, minimum, and maximum objective function values obtained by the Branch and Bound algorithm and the two heuristics, the EID heuristic and the 2-exchange heuristic. (Left: 5% contamination, RIGHT: 30% contamination)

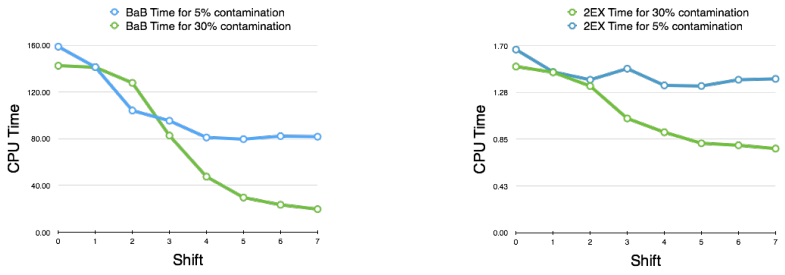

Figure 2: Average CPU times for the Branch and Bound algorithm (LEFT) and the 2-exchange heuristic (RIGHT).

7 CONCLUSIONS AND SUMMARY

Detecting outliers in a data set is an important question in statistics. There are various methods that can be used for this purpose such as the Minimum Covariance Determinant (MCD) Estimator of [26] or distance-based methods as in [7]. The Minimum Volume Ellipsoid Estimator (MVE) is one of the most reliable methods in outlier detection. The MVE problem finds an ellipsoid with the smallest possible volume which covers a large portion of the data set (usually close to half). Naturally, the problem is related to the well-known Minimum Volume Enclosing Ellipsoid problem. The MVEE problem can be formulated as a convex optimization problem and can be solved efficiently even for very large instances using a first-order method.

This paper develops a new heuristic for calculating the MVE estimator based on the new advances in solving the MVEE problem. In particular, the suggested method improves the existing methods in three ways. First, new initialization techniques are presented which can be incorporated into other methods as well. Second, given a feasible solution to the problem, lower bounds are provided on the feasible solutions in the neighborhood. These bounds decrease the number of candidate subsets to be investigated which increases the efficiency of the algorithm significantly. Third, it is noted that each subset can be evaluated much faster by taking advantage of better initial solutions and fathoming unpromising candidates much earlier. Although, the proposed heuristic makes heavy use of the first-order method that is used to solve the MVEE problems for each subset, the ideas presented in this paper can be insightful in developing similar search methods based on other MVEE algorithms. In addition, similar techniques can be used to develop a richer class of exchange heuristics where more than two points are exchanged at each iteration. We also provide enough information for developing an efficient branch-and-bound algorithm to obtain exact solutions for the problem. Further computational studies can be carried out to compare the new method and the other available methods such as metaheuristics. We believe that the method outlined in this paper has significant advantages because it exploits the structure of the particular problem instead of applying a generic approach.

References

[1] J. Agullo, Exact Iterative Computation of the Multivariate Minimum Volume Ellipsoid Estimator with a Branch and Bound Algorithm, Pro-

ceedings in Computational Statistics editted by A.Prat, 175–180, Heidelberg: Physica-Verlag (1996)

[2] S. D. Ahipaşaoğlu and P. Sun and M. J. Todd, Linear Convergence of a Modified Frank-Wolfe Algorithm for Computing Minimum-Volume Enclosing Ellipsoids, Optimization Methods and Software, 23, 5–19 (2008)

[3] S. D. Ahipaşaoğlu, S. D. and E. A. Yıldırım, Identification and Elimination of Interior Points for the Minimum Enclosing Ball Problem, SIAM Journal on Optimization, 19, 1392–1396 (2008)

[4] S. D. Ahipaşaoğlu, Solving Ellipsoidal Inclusion and Optimal Experimental Design Problems: Theory and Algorithms, Ph.D. thesis, Cornell University, August (2009)

[5] C. L. Atwood, Optimal and Efficient Designs of Experiments, The Annals of Mathematical Statistics, 40, 1570–1602 (1969)

[6] C. L. Atwood, Sequences converging to D-optimal designs of experiments, The Annals of Statistics, 1, 342–352 (1973)

[7] S. D. Bay and M. Schwabacher, Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule, In Proceedings of Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, D.C. USA, 29–38 (2003)

[8] R. D. Cook, D. M. Hawkins and S. Weisberg, Exact Iterative Computation of the Robust Multivariate Minimum Volume Ellipsoid Estimator, Statistics and Probability Letters,16, 213–218 (1993)

[9] V. V. Fedorov, , Theory of Optimal Experiments, Academic Press, New York (1972)

[10] M. Frank and P. Wolfe, An Algorithm for Quadratic Programming, Naval Res. Logis. Quart., 3, 95–110 (1956)

[11] S. Galelli and A. Castelletti, Tree-based iterative input variable selection for hydrological modeling, Water Resouces Research, 49, 4295–4310 (2013)

[12] J. Gotoh and H. Konno, Minimal Ellipsoid Circumscribing a Polytope Defined by a System of Linear Inequalities, Journal of Global Optimization, 34, 1–14 (2006)

[13] S. C. Grambow and A. J. Stromberg, Combining the EID and FSA for Computing Minimum Volume Ellipsoid, Dept. of Stats., University of Kentucky (1998)

[14] R. Harman and L. Pronzato, Improvements on removing non-optimal support points in D-optimum design algorithms, Statistics and Probability Letters, 77, 90–94 (2007)

[15] D. M. Hawkins, A Feasible Solution for the Minimum Volume Ellipsoid estimator in Multivariate Data, Computational Statistics, 8, 95–107 (1993)

[16] D. M. Hawkins and D. J. Olive, Improved Feasible Solution Algorithms for High Breakdown Estimation, Computational Statistics and Data Analysis, 30, 1–11 (1999)

[17] E. M. T. Hendrix, I. Garca, J. Plaza, and A. Plaza, On the Minimum Volume Simplex Enclosure Problem for Estimating a Linear Mixing Model, Journal of Global Optimization, doi: 10.1007/s10898-012-9876-5 (2012)

[18] E. M. T. Hendrix and B. G. Toth, Introduction to Nonlinear and Global Optimization, Springer, Cambridge (2010)

[19] F. John, Extremum Problems with Inequalities as Subsidiary Conditions, in: Studies and Essays, presented to R. Courant on his 60th birthday January 8, 187–204, Interscience, New York (1948) reprinted in: J. Moser, (ed.), Fritz John, Collected Papers, vol. 2, Birkhuser, Boston, 543–560 (1985)

[20] L. G. Khachiyan, Rounding of Polytopes in the Real Number Model of Computation, Mathematics of Operations Research, 21, 307–320 (1996)

[21] P. Kumar and E. A. Yıldırım, Minimum volume enclosing ellipsoids and core sets, Journal of Optimization Theory and Applications, 126 (1), 1–21 (2005)

[22] A Pazman. Foundations of Optimum Experimental Design. Reidel, Dordrecht (1986)

[23] W. L. Poston, A Deterministic Method for Robust Estimation of Multivariate Location and Shape, Journal of Computational and Graphical Statistics, 6300–313 (1997)

[24] W. L. Poston R. H.Tolson, Maximizing the Determinant of the Information Matrix with the Effective Independence Distribution Method, AIAA Journal of Guidance, Control and Dynamics, 15, 1513–1514 (1992)

[25] F. Pukelsheim, Optimal Design of Experiments, John Wiley and Sons, Inc., New York (1993)

[26] P. J. Rousseeuw and K. V. Driessen, A Fast Algorithm for the Minimum Covariance Deteriminant Estimator, Technometrics, 41, 212-223 (1999)

[27] P. J. Rousseeuw and A. M. Leroy, Robust Regression and Outlier Detection, John Wiley and Sons, New York (1987)

[28] S. D. Silvey, Optimal Design: An Introduction to the Theory for Parameter Estimation, Chapman and Hall, New York (1980)

[29] S. D. Silvey, D. H. Titterington and B. Torsney, An algorithm for optimal designs on a design space, Communications in Statistics: Theory and Methods, Comm. Statist. Theory Methods, 7(14), 1379–1389 (1978)

[30] P. Sun and R. M. Freund, Computation of Minimum Volume Covering Ellipsoids, Operations Research, 52(5), 690–706 (2002)

[31] D. M. Titterington, Algorithms for computing D-optimal designs on a finite design space, In: Conference on Information Sciences and Systems, Department of Electrical Engineering, Johns Hopkins University of Baltimore, pp.213–216 (1976)

[32] M. J. Todd and E. A. Yıldırım, On Khachiyan's Algorithm for the Computation of Minimum Volume Enclosing Ellipsoids, Discrete and Applied Mathematics, 155, 1731–1744 (2007)

[33] B. Torsney and R. R. Martin-Martin, Multiplicative algorithms for computing optimum designs, Journal of Statistical Planning and Inference, 139, 3947–3961 (2009)

[34] F. Torti and D. Perrotta and C. Atkinson and M Riani, Benchmark Testing of Algorithms for Very Robust Regression: FS, LMS and LTS, Computational Statistics and Data Analysis, 56, 2501-2512 (2012)

[35] L. Vandenberghe and S. Boyd, Applications of Semidefinite Programming, Applied Numer. Math., 29, 283–299 (1998)

[36] D. L. Woodruff and D. R. Rocke, Heuristic Search Algorithms for the Minimum Volume Ellipsoid, Computational and Graphical Statistics, 2, 69–95 (1993)

[37] H. P. Wynn, The sequential generation of D-optimum experimental design, Annals of Mathematical Statistics, 41, 1655–1664 (1970)

[38] H. P. Wynn, Results in the Theory and Construction of D-optimum Experimental Designs, Journal of the Royal Statistical Society, Series B (Methodological), 34,133–147 (1972)

[39] M. Yang and S. Biedermann and E. Tang, E, On optimal designs for nonlinear models: a general and efficient algorithm, Journal of the American Statistical Association, In press. DOI: 10.1080/01621459.2013.806268.

[40] E. A. Yıldırım, Two algorithms for the minimum enclosing ball problem, SIAM Journal on Optimization, 19,1368–1391 (2008)

[41] Y. Yu, D-optimal designs via a cocktail algorithm, Statistics and Computing, 21, 475–481 (2011)

	n	m	h	EID	2EX	Time2EX(ms)
mean	5	100	53	32.68	27.97	37.89
std				6.81	7.42	26.10
min				2.57	-3.89	7.31
max				45.50	44.89	162.08
mean	5	200	103	32.91	29.35	119.11
std				6.42	7.41	126.05
min				7.66	0.24	20.90
max				46.39	45.76	990.47
mean	5	500	253	32.68	30.15	606.32
std				6.56	7.30	654.13
min				6.09	2.92	210.46
max				45.06	44.80	7434.33
mean	10	100	56	60.54	51.23	82.49
std				12.16	13.13	42.98
min				5.67	0.23	14.72
max				83.75	77.43	278.59
mean	10	200	106	59.59	51.54	303.07
std				12.71	13.65	207.19
min				-2.20	-14.36	42.60
max				83.96	78.60	1316.48
mean	10	500	256	58.18	52.10	1663.19
std				12.44	13.55	1851.06
min				1.90	0.55	247.67
max				86.03	84.64	15425.50
mean	20	100	61	115.58	102.65	132.63
std				29.22	29.01	46.55
min				-16.87	-22.12	18.67
max				172.50	164.50	399.52
mean	20	200	111	116.88	105.92	500.42
std				26.32	27.00	246.59
min				-38.70	-46.06	77.63
max				172.86	169.45	1779.15
mean	20	500	261	112.98	104.00	3252.19
std				26.87	28.25	2329.57
min				10.65	1.37	364.63
max				176.30	174.27	15740.60

Table 6: Average, minimum, and maximum objective function values for large datasets with standard deviations

n	m	h	N =1	N =10	N =100	N=1000
2	20	12	429	837	912	976
2	30	18	312	735	870	953
2	50	27	221	653	815	949
3	20	12	351	826	857	910
3	30	18	129	633	788	908
3	50	27	89	469	738	890
5	20	12	340	637	758	870
5	30	18	293	598	733	810

Table 7: Number of problems (out of 1000 random instances) solved to optimality with $N = 1, 10, 100$, and 1000 random starts.