An Evolutionary Approach to Constructing the Minimum Volume Ellipsoid Containing a Set of Points and the Maximum Volume Ellipsoid Embedded in a Set of Points

Rewayda Abo-Alsabeh¹ & Abdellah Salhi²
¹Department of Mathematical Sciences, University of Kufa, Iraq
²Department of Mathematical Sciences, University of Essex, UK

Email: ruwaida.mohsin@uokufa.edu.iq, as@essex.ac.uk

Abstract. Given a set of points \( C = \{x_1, x_2, ..., x_m\} \subseteq \mathbb{R}^n \), what is the minimum volume ellipsoid that encloses it? Equally interestingly, one may ask: What is the maximum volume ellipsoid that can be embedded in the set of points without containing any? These problems have a number of applications beside being interesting in their own right. In this paper we review the important results concerning these and suggest an evolutionary-type approach for their solution. We will also highlight computational results. Keywords: Löwner-John ellipsoid, Maximum volume ellipsoid, Approximation, Optimisation, Genetic algorithm

1. Introduction
Consider first the problem of estimating the Löwner-John ellipsoid. It can be described as follows. Given a set \( C \) of points in \( \mathbb{R}^n \), what is the minimum volume ellipsoid that contains all the points of \( C \)? This is referred to as the Minimum Volume Enclosing Ellipsoid or MVEE problem. The second problem, also related to MVEE, that we consider here is that of finding the Largest Volume Enclosed Ellipsoid or LVEE, that can be embedded within the set of points \( C \) in \( \mathbb{R}^n \), without containing any of the points. This can be described as the problem of finding the biggest hole within a set of points in some dimension. Equivalency, it is the problem of finding the largest convex hull that is embedded within the set of discrete points. In the following we treat these problems in turn and show how they can be formulated, and solved in particular using an evolutionary approach. Illustrations will be provided as well as computational results.

1.1. The ellipsoid: Preliminaries
An ellipsoid in \( \mathbb{R}^n \) is the image of the unit ball \( B^n \) of \( \mathbb{R}^n \) under a nonsingular affine map of the form.

\[
\Psi(x) := Ax + b
\]

where \( \Psi : \mathbb{R}^n \to \mathbb{R}^n \) is linear and \( b \in \mathbb{R}^n \) is a constant vector. The ellipsoid has many representations, such as

\[
E = \{ x \in \mathbb{R}^n \mid (x - b)^T A^{-1} (x - b) \leq 1 \}
\]

for \( b \in \mathbb{R}^n \) and \( A = A^T > 0 \).
The MVEE or Löwner-John (L-J) ellipsoid $E_{ij}$ of a set $C$ is the minimum volume ellipsoid that contains the interior point approach were introduced by the same authors [70]. Without loss of generality, we can assume that the volume of $E$ is proportional to $\det A^{-1}$. For $A \in S_+^{n}$, the problem of finding the MVEE of $C$ can be written as

$$\minimize \log \det A^{-1}$$

subject to

$$\sup_{b \in C} \| Av + b \|_2 \leq 1, \quad i = 1, 2, \ldots, m$$

for an implicit constraint $A > 0$. The variables are $A \in S^n$, and $b \in R^n$. This problem is convex as both the objective and the constraint functions are convex in $A$ and $b$.

1.2. The Minimum Volume Enclosing Ellipsoid Problem

MVEE arises in a number of practical situations such as when placing relay antennae, packing and packaging to name a few. It is commonly solved using convex programming and boils down to finding the so-called Löwner-John ellipsoid [10]. Computing the MVEE of a finite set of points $x_1, \ldots, x_m \in R^n$ is equivalent to computing the MVEE of a polytope that is defined by the convex hull of those points. By applying (4) this problem can be written as

$$\minimize \log \det A^{-1}$$

subject to

$$\| Av + b \|_2 \leq 1, \quad i = 1, 2, \ldots, m$$

for the implicit constraint $A > 0$, and the variables $A \in S^n$, and $b \in R^n$, [5]. The problem of approximating $E$ has many applications in statistics and optimal design, [20, 22], integer programming, [14, 9], computational geometry [19], and randomized algorithms for polytope volume computation, [6, 15]. Leonid G. Khachiyan [11] considered the problem of $(1 + \epsilon)n$-rounding of a set $C$ of $m$ points in $R^n$, which boils down to finding an ellipsoid $E \in R^n$ where

$$[(1 + \epsilon)n]^{-1} \subseteq E \subseteq \text{conv}(C) \subseteq E$$

He suggested an $O(mn^2(\epsilon^{-1} + \ln n + \ln \ln m))$ algorithm, which implies that approximating the minimum volume ellipsoid of $C$ can be solved in $O(m^{3.5} \ln(me^{-1}))$ operations, where $\epsilon$ is a relative accuracy.

Kumar and Yildirim [13] addressed the problem of calculating a $(1 + \epsilon)$-approximation of the MVEE of a set $C$ of $m$ points by modifying Khachiyan's algorithm. Their algorithm has the better complexity bound of $O(mn^3/\epsilon)$ operations for $\epsilon \in (0, 1)$. It returns a core set $Y \subset C$ such that the MVEE of $Y$ gives a good approximation of $C$. Moreover, the size of $Y$ does not rely on the number of points $m$, but only on the dimension $n$ and relative accuracy $\epsilon$. Their results give $|Y| = O(n^2/\epsilon)$ for $\epsilon \in (0, 1)$.

Todd and Yildirim [23] studied two related problems; finding an approximate rounding ellipsoid of the convex hull of a finite set of points and finding the MVEE of a finite set of points. They found a relationship between the polar of the deepest cut ellipsoid method and Khachiyan's barycentric method coordinate descent (BCD) which calculates an approximate rounding of the convex hull of a set of vectors. They implemented Khachiyan's BCD method in an efficient way and also provided a modification.

$S_n$ and $S^{n+}$ denote $n \times n$ symmetric matrices and symmetric positive definite matrices, respectively. We use $\succ$ to denote strict matrix inequality between symmetric matrices.

Cation of the algorithm of Kumar and Yildirim, [13]. They found that their modification does not raise the complexity of the latter algorithm. Their algorithm calculates a good approximate solution and calculates a smaller core set than that computed by the algorithm of Kumar and Yildirim, [13].

The minimal ellipsoid circumscribing a polytope that is defined by a set of finitely many points can be computed by many algorithms. Algorithms that use the interior point approach were introduced by Zhang [28], Zhang and Gao [29], and Khachiyan and Todd [12]. The one provided by Barnes [3] relies on quadratic programming. There are other algorithms of the stochastic type, introduced by Gärtner and Schönherr [7]. Other studies of the problem are by Yildirim [27], Vaidya [24], and Anstreicher [1].
Sun and Freund [21] introduced the Dual Reduced Newton (DRN) algorithm, which solves large instances that arise in data mining contexts. It is based on combining active-set [21] and interior-point methods. The DRN is regarded as an interior point algorithm. It was applied to 10 randomly generated large problem instances, \(10 \leq n \leq 30\) and \(1000 \leq m \leq 30000\).

Welzl [25] developed a simple randomized algorithm that finds the smallest enclosing disks (balls and ellipsoids) in the plane for a set of points in linear time. This algorithm depends on Seidel’s Linear Programming algorithm [18], which works mainly in low dimensions. Welzl generalized it to higher dimensions and provided a heuristic that leads to an improved procedure, which finds the smallest enclosing ball for 5000 points in 10-dimensions.

1.3 Computing MVEE in low dimensions

We illustrate briefly here how to compute MVEE using existing software such as the CVX package of Matlab, [8], which implements the Path-Following Infeasible Interior-Point Algorithm. Instances of the model described by Equation (5) can be submitted to this package in a straightforward manner. However, in dimensions > 3, the problems become computationally demanding. We consider here the case of 2d.

Consider again the model of the L-J ellipsoid problem of Equation (5). Recall that in this model the unknowns are \(A\) and \(b\). The data is the set of points \(x_i \in \mathbb{R}^n, i = 1, \ldots, m, \ m = 50\), randomly generated within interval \([20, 30]\) in 2d to ensure that the ellipsoids would be entirely in the positive quadrant. The resulting instance has been successfully solved with CVX. The output is represented in Figure 1 showing the minimum ellipsoid enclosing perfectly the convex hull of the given set of points in 2d.

The aim of this paper is to solve the problems of MVEE and LVEE using an evolutionary approach. This is different from the methods that are mentioned above. The novelty is in the adequate implementation of GA. It requires a suitable representation of the solutions, a fitness function that distinguishes between the ellipsoids as well as suitable genetic operators and stopping criteria. The rest of the paper is organized as follows. Section 2 is the proposed implementation of GA algorithm for solving the MVEE. The problem of LVEE is studied in section 3. Section 4 is the conclusion.

2. Computing MVEE using GA

As mentioned earlier, exact approaches to MVEE are not viable in high dimensions and large cardinality sets. Approximate approaches are therefore called upon. Here, we consider an evolutionary approach namely the Genetic Algorithm or GA. Moreover, we intend to implement and apply GA to the MVEE problem in an innovative way which is different from applying existing implementations of GA as can be found in the Matlab Optimisation Toolbox, for instance.

2.1. GA Implementation and application to MVEE

There are a number of ways to implement GA, [16]. They differ mainly in the way individuals are represented. Some representations are more suitable than others in terms of ease of coding and computing time.
Figure 1. Minimum ellipsoid generated by the infeasible path-following approach implemented in CVX, [8]

2.1.1. Implementation 1: handling the MVEE model (5)
Consider again model (5). In $2d$, $A$ is a $2 \times 2$ matrix corresponding to four variables and $b$ is a $2 \times 1$ column corresponding to two variables. Given the points to enclose by an ellipsoid of minimum volume, a solution to model (5) can be represented as a 1-dimensional array with $n(n + 1)/2$ entries of matrix $A$ and $n$ entries of vector $b$, i.e. $(n^2 + 3n)/2$, or $5$ in $2d$. In other words, chromosomes of length $5$ will have to be generated to form a population of individuals on which the GA will work. The fitness function is no other than the objective function of model (5). Solutions must satisfy the constraints of this model too. The solver minimizes the objective function subject to the constraints until the minimum volume ellipsoid enclosing the points is found.

In the candidate solution, the first four components belong to $A$ and the remainder belong to $b$ which is the centre of the ellipsoid. From this solution, the parameters of the MVEE are extracted as follows. Suppose all the eigenvalues of $A$ are positive real numbers. The length of the major axis is $\sqrt{\Theta}$ where $\Theta$ is the largest eigenvalue of $A$, while the minor axis is $\sqrt{\theta}$ with $\theta$ being the smallest eigenvalue of $A$. The solution found by this implementation of GA is not quite as good as one would like it to be since it is only an approximation. The instance solved with the exact approach has also been solved with this implementation of GA.

2.1.2. Implementation 2: an alternative solution representation
This implementation represents a solution as the ellipsoid itself rather than the entries of matrix $A$ and vector $b$ of model (5).

3. Solution representation
A solution is an ellipsoid which is represented as a chromosome depicted in Table 1. It comprises the following genes:

- The coordinates $(c_1, c_2)$ of a point representing the centre of the ellipsoid;
- The length $a$ of the major axis of the ellipsoid;
- The length $b$ of the minor axis of the ellipsoid;
- The angle $\theta$ of tilt of the major axis with respect to the x-axis.

Table 1. Chromosome representation of an ellipsoid

| Major axis | Minor axis | centre | angle of tilt |
|------------|------------|--------|---------------|
| $a$        | $b$        | $c_1$  | $c_2$         | $\theta$ |

Chromosome for $d > 2$

| axes | centre | angle of tilt |
|------|--------|---------------|
| $a_n$ | $c_n$ | $\theta$ |

For $d > 2$, the chromosome's length can be generalized to $2n + 1$. The first $n$ genes are the axes, the second $n$ genes are the centre components, and the last gene is for the tilt angle.

4. Genetic operators
Starting from an initial randomly generated population of ellipsoids (solutions), subsequent populations are produced using genetic operators of crossover, mutation and reproduction.

Crossover operator There are a number of possible implementations of this operator. The constant theme running through them all is that they operate on two parents to produce two children. Here, the parents are ellipsoids as represented in Table 1. The crossover produces two new children with
potentially different axes, centres and angles of tilt. The Intermediate operator is used here. The gene values of the child are selected somewhere between and around the gene values of the parents. The children are created according to the formula:

$$Child1 = Parent1 + \beta (Parent2 - Parent1)$$

in which $\beta$ is a scaling factor selected uniformly randomly in $[-0.25; 1.25]$. Each gene in the new child is the output of combining the genes of the parents according to Equation (7) with the new $\beta$ selected for each gene [17]. Figure 2 illustrates that; it represents the two children with Child2 containing all the points. Note that crossover helps cover the search space in the process of optimisation. Table 3 shows the experiments that lead to choose this crossover.

**Mutation operator** The mutation operator enables GA to avoid becoming trapped in local optima. This is implemented by changing one or more genes. Gaussian mutation is the type of mutation implemented here, where the experiments in Table 4 show that it is the suitable one for this problem. It has been introduced in Evolution Strategies by Rechenberg in 1973, [4], where a Gaussian element is added to the individual or parent to generate the new offspring. This number is taken from a Gaussian distribution with mean value 0. Gaussian mutation can control the variance of the population fitness of each generation according to [26].

**Reproduction** This operator copies some fit individuals into the next generation without any modification.

### 5. Fitness function

The fitness function is the measure by which individuals/solutions are ranked. Here, it is the difference between the volume of the ellipsoid and the number of points it encloses as given below. Minimizing this expression provides the desired minimal ellipsoid containing the set of points. The fitness is written as follows. Let $\phi_E$ be the fitness of ellipsoid $E$, $C$ the set of points to enclose, $C_E$ the set of points that ellipsoid $E$ contains, $V_E$ the volume of ellipsoid $E$, and $\alpha$ a constant, such that $\alpha \geq |C|^2$, here for the example that is illustrated down $\alpha = 100$.

$$\phi_E = \begin{cases} 
|C_E| - |C| + V_E + \alpha & \text{if } (|C_E| < |C|) \\
V_E - |C_E| & \text{else}
\end{cases}$$

Figure 2. (a) An illustration of a chromosome; (b) Breeding new children from parents

Note that randomly generated ellipsoids may contain all the points while others may contain only some or none. The ellipsoids that contain all the points may have different volumes. According to the GA strategy, individuals with better fitness, i.e. a relatively small volume and a large proportion of points enclosed, are likely to survive into the next generation. This means ellipsoids containing no points
of \( C \) are unlikely to survive. Those with all points and large volumes will not have a good fitness value in contrast to those with small volumes and all points in. The latter are likely to survive.

To illustrate how the fitness works, suppose we have five ellipsoids with different volumes and aim to enclose a set of points \(|C| = 10\). Five different cases may arise, see Table 2. It is clear that the best ellipsoids that contain \( C \) are \( E_3 \) and \( E_5 \), but if we sort the fitness values say in ascending order, the values become \([-1, 1, 4, 6, 10]\) for \( E_1, E_2, E_3, E_4, E_5 \), respectively. This produces \( E_1 \) and \( E_2 \) as the best individuals that may be selected by the selection operator to produce new children and gives \( E_4 \) better than \( E_5 \). This inconsistency between the fitness values leads to an ineffective selection which will often miss the best individuals. Therefore, any ellipsoid has \(|C_E| < |C|\), we compute its fitness value using this function \( \phi_E = (|C| - |C_E| + V_E + \alpha)^2 \) to make its fitness very large in comparison to those ellipsoids that contain all the points. This will give \( E_1, E_2 \) and \( E_4 \) much worse fitness values than those of \( E_3 \) and \( E_5 \), resulting into the following new ranking \([4, 10, 11881, 12321, 13456]\) for \( E_3, E_5, E_1, E_2, E_4 \), respectively. This gives \( E_3, E_5 \), the chance to survive into the next generation.

### Table 2. An example of different cases of ellipsoids

| \( E \) | \( V_E \) | \(|C_E|\) | \( V_E - |C_E| \) |
|---|---|---|---|
| \( E_1 \) | 1 | 2 | -1 |
| \( E_2 \) | 6 | 5 | 1 |
| \( E_3 \) | 14 | 10 | 4 |
| \( E_4 \) | 6 | 0 | 6 |
| \( E_5 \) | 20 | 10 | 10 |

6. Stopping criteria

Two stopping criteria are used here; one is the maximum number of generations, the other is the number of still generations, i.e. the number of generations with no progress in the objective function, or fitness function value.

7. Selection procedure

It selects parents for the next generation based on their fitness values. The Stochastic uniform selection used here, is also known as the Stochastic Universal Sampling (SUS) which is an improvement on the Roulette Wheel selection, [2]. This type of selection is chosen depending on the results of the experiments in Table 5.

7.1 GA for MVEE: Experimental investigation

To choose the most suitable crossover and mutation operators we proceeded to some experimentation as explained below.

- We applied five types of crossover operators (Single point, Two point, Intermediate, Scattered, and Heuristic) to different sets of points in \( 2d \) and \( 3d \) for the same set of points and initial population. These results show that the Intermediate operator produces the smallest MVEE for most of the cases, see Table 3. We applied all the mutations and selections that are used in Table 4 and 5 with the five operators and we found that Gaussian mutation and Stochastic uniform selection produce best result with the Intermediate operator.

- We applied three types of mutation (Uniform, Gaussian, and Adaptive feasible). We applied them to different sets of points and selected the best.

Table 4 shows that Gaussian mutation produces the smallest MVEE for most of the cases. Here we used also the crossovers and selections by turn with these mutations then we chose the Intermediate operator and Stochastic uniform selection to get these results.
In the same way, five types of selections (Stochastic uniform, Remainder, Uniform, Roulette, and Tournament) are tested. Table 5 shows that the Stochastic uniform selection, Remainder, and Roulette wheel are the best. Their performances are very similar, and so we settle for the Stochastic uniform selection. Moreover, in our experimental investigation Intermediate and Gaussian operators are used.

Table 3. Crossover operators

| Dim | Points | Single point | Two point | Intermediate | Scattered | Heuristic |
|-----|--------|--------------|-----------|--------------|-----------|-----------|
| 2   | 10     | 76.24        | 70.14     | 52.89        | 80.25     | 52.83     |
| .   | 20     | 153.12       | 145.77    | 112.50       | 137.77    | 112.31    |
| .   | 50     | 142.90       | 119.11    | 93.81        | 122.75    | 122.75    |
| .   | 60     | 180.81       | 167.16    | 137.51       | 164.99    | 133.727   |
| 3   | 10     | 175.78       | 146.09    | 55.31        | 204.82    | 123.798   |
| .   | 20     | 144.23       | 228.85    | 95.142       | 184.66    | 113.42    |
| .   | 30     | 295.89       | 230.70    | 141.15       | 222.71    | 141.82    |
| .   | 50     | 607.35       | 361.63    | 212.11       | 374.80    | 276.93    |

Based on these results we applied Algorithm 1 with the Intermediate crossover and the Gaussian mutation. SUS was adopted for selection procedure. Note that in the implementation, when crossover is applied, it is followed by mutation on the best individual generated to improve the result.

Table 4. Mutation operators

| Dim | Points | Uniform | Gaussian | Adaptive feasible |
|-----|--------|---------|----------|-------------------|
| 2   | 10     | 201.32  | 79.61    | 26.73             |
| .   | 20     | 145.39  | 87.74    | 138.63            |
| .   | 50     | 160.35  | 112.60   | 112.83            |
| 3   | 20     | 466.79  | 166.16   | 304.66            |
| .   | 30     | 400.57  | 138.09   | 213.05            |
| .   | 50     | 680.65  | 211.00   | 299.47            |

Table 5. Selection functions

| Dim | Points | Stochastic uniform | Reminder | Uniform | Roulette | Tournament |
|-----|--------|---------------------|----------|---------|----------|------------|
| 2   | 10     | 92.01               | 93.24    | 144.09  | 93.06    | 284.46     |
| .   | 20     | 61.11               | 59.96    | 91.06   | 61.06    | 174.87     |
| .   | 50     | 105.42              | 105.74   | 159.78  | 106.29   | 321.65     |
| 3   | 10     | 102.84              | 102.05   | 677.50  | 101.272  | 758.56     |
| .   | 20     | 135.15              | 133.64   | 721.88  | 137.29   | 888.97     |
| .   | 50     | 161.09              | 159.678  | 787.21  | 169.16   | 682.35     |

7.2. In 2 and 3 dimensions
We considered the same 50 points in 2d that are used in the exact approach of Section 2.1.1. To find the L-J ellipsoid containing them with GA, we first randomly generated a population of 100 ellipsoids, 200 randomly generated points in the plane; every pair of points representing an ellipsoid. The rates of crossover and mutation are set to 0.8, and 0.1, respectively. This means that the rate of reproduction is 0.1. After 51 generations, an approximate L-J ellipsoid is obtained. Figure 3 represents three different stages of the GA search process. Figure (a) shows the initial population of ellipsoids. Figure (b) shows, overlapping, the best ellipsoids found in generations 10, 20, 30 and 40 generations. Figure (c) shows the
final solution after 51 generations. The same idea is applied to a set of points in 3d, with GA using a population of 70 ellipsoids. The problem is solved after 120 generations. Figure 4 illustrates this.

**Algorithm 1 GA for MV EE**

1: Input $C$ the set of random points to be enclosed;
2: Input $\rho$ the rate of crossover and $\mu$ the rate of mutation;
3: Generate a random population of ellipsoids $E$;
4: Evaluate $\phi$ the fitness of all ellipsoids in $E$ using Equation (8);
5: Rank individuals according to their fitness;
6: Select parents from the population according to (SUS) selection;
7: Generate a new population by applying the following operators: crossover (Intermediate), mutation (Gaussian), and reproduction with their particular rates;
8: Compute the fitness of the individuals of the new population;
9: Until (The stopping criteria are satisfied) Repeat from 5.
10: Return Best solution.

7.3. *Alleviating the computational burden*

Checking the inclusion of large numbers of points in a given ellipsoid is time consuming even in low dimensions. To make the GA approach viable in practice and higher dimensions, it is necessary to reduce this computational burden. It is clear, for instance that, if the convex hull of the given set of points is known, then only its vertices need enclosing. Unfortunately computing the convex hull is harder than finding the L-J ellipsoid or MV EE. However, this suggests that only the peripheral points of the set need to be known, and at most $(n^2 + 3n)/2$ points, [10]; these are the ones that need enclosing; those inside would belong necessarily, if the peripheral ones belong. This may reduce the computational load. Here, we outline three different ways to reduce the number of points to consider for inclusion without compromising the inclusion of the rest of the points of set $C$. They are:

1. The convex hull approach which considers only the vertices of the convex hull of $C$;
2. The boundary approach which considers only the boundary points of set $C$;
3. The border approach which considers the points in a band of arbitrary width $\epsilon$ that is between two concentric hypercubes the largest being the smallest containing $C$.

**Figure 3.** (a) 100 Ellipsoids in 2d; (b) Ellipsoids of intermediary generations; (c) LÖwner -John ellipsoid found in generation 51
As said earlier, the convex hull approach is expensive; so is the boundary approach which identifies all points on the boundary of C. The border approach which we introduce here, on the other hand, is a rough procedure to find the peripheral or border points of the set that need enclosing. It is computationally cheap. This procedure is as follows:

1. Find points with minimum and maximum coordinates over all axis;
2. Choose width $\epsilon$ for the border that we want to impose. Such a border could be defined by inequalities
   \[
   \min_i x_i + \epsilon \leq x_i \leq \max_i x_i - \epsilon, \quad i = 1, \ldots, m
   \]  

The number of points in the border depends on its width $\epsilon$. For example 1000 points with $\epsilon = 0.1$ leads to about 40 points to check for inclusion. A comparison between these three approaches is provided in Table 6. The convex hull procedure (Columns 3 and 4) comes from Matlab and handles points up to 9d. The boundary points are found using a Matlab Tool box function which only works in spaces for points up to 3d, (columns 5 and 6). The border points approach we introduce here performs best, (Columns 7, 8, and 9). Over all, the total time for finding the MVEE depends on the number of points, the number of generations, and the population size. Reducing any of these reduces computing time.

Table 6. Comparison of Convex Hull, Boundary, and the Border points approaches in 2 to 9d

| Dim | Points | ConvHull | Time(s) | Boundary | Time(s) | Border | Time(s) | $\epsilon$ |
|-----|--------|----------|---------|----------|---------|--------|---------|-----------|
| 2   | 50     | 12       | 0.057   | 15       | 0.080   | 16     | 0.059   | 0.6       |
| 2   | 100    | 9        | 0.050   | 13       | 0.076   | 14     | 0.061   | 0.3       |
| 2   | 1000   | 17       | 0.062   | 43       | 0.094   | 40     | 0.098   | 0.1       |
| 2   | 10000  | 28       | 0.082   | 165      | 0.255   | 91     | 0.081   | 0.02      |
| 3   | 1000   | 61       | 0.059   | 220      | 0.120   | 86     | 0.068   | 0.6       |
| 4   | 1000   | 165      | 0.740   | *        | *       | 159    | 0.044   | 0.1       |
| 5   | 1000   | 298      | 0.115   | *        | *       | 116    | 0.062   | 0.1       |
| 6   | 1000   | 461      | 0.542   | *        | *       | 510    | 0.045   | 0.28      |
| 7   | 1000   | 633      | 5.494   | *        | *       | 637    | 0.047   | 0.35      |
| 8   | 1000   | 778      | 58.792  | *        | *       | 768    | 0.062   | 0.4       |
| 9   | 1000   | *        | *       | *        | *       | 877    | 0.062   | 0.48      |

(*) solution not found
7.4. Computing MVEE in high dimensions
Computing MVEE in high dimensions ($n \geq 4$) is computationally challenging. The approach is the same as in lower dimensions. Here the initial population of the ellipsoids should be varied enough to include large ellipsoids that enclose all the points. Experimental results recorded in Table 7 show the number of points and the total computing time in seconds taken to solve different instances in different dimensions. The population size is restricted to be between 70 and 100. We run the algorithm between 1-3 times with different $\epsilon$ and select the best. We compare these results for the same data sets with those returned by an exact approach namely the path-following infeasible algorithm. The first two columns represent the dimensions and the sets of points. Columns 3 to 7 show the results from the GA algorithm, the number of points on the border of the set that are used, the number of generations, the CPU time, $\epsilon$ the width of the border, and the volume of the approximate minimum ellipsoid found. The columns 8 and 9 of the table show the results of the exact algorithm, volume and CPU time, respectively. The last column is the error as a percentage of the volume of the exact solution, or minimum ellipsoid returned by the exact approach. Note that the exact algorithm cannot find the solution for large data sets and in high dimensions. Having said that, although the GA approach works on all problems, the quality of the solutions is rather poor, according to what is recorded in the error column. Note, however, that in high dimensions' balls and ellipsoids behave like empty shells will all the volume concentrated near the surface. It is, therefore, understandable that although the ellipsoids returned by the exact method and GA may be very similar, a slight tilt will result in a huge difference in volume between the two ellipsoids.

The relation between $\epsilon$ and the dimensions and between $\epsilon$ and the number of points are opposite. This factor must be small when increasing the dimension and number points. Although, the ellipsoid has enclosed the border points, some points are missed and lie outside the ellipsoid. So we need to add these points to the border points and enclose them again. Note that both algorithms have been run five times and the best solutions (volume and time) are recorded.

![Figure 5. A set with outlier point](image)

7.5. Special case
Applications of MVEE often consider near perfect cases where the data set is convex and has no outliers or clusters. In the presence of outliers, the formulation of the MVEE problem can take that into consideration. With respect to our algorithm when the set has some outliers, the interval that is used to generate the ellipsoids should be large enough to cover all the points including the outliers. This is guaranteed by generating large ellipsoids. See Figure 5.
8. GA implementation for LVEE
In contrast to the MVEE problem, here, we are interested in finding the largest volume ellipsoid that does not cover any points, or LVEE. In other words, given a set of points in $\mathbb{R}^n$, the problem is to find the largest ellipsoid that can fit between the points without including any. It is important to add that ellipsoids which are all or partially outside the convex hull of the given points, do not qualify. To tackle this problem with GA, we propose Equation (10) as fitness function. Let $\phi_E$ be the fitness of ellipsoid $E$, $|C|$ the size of the given set of points, $|C_E|$ the size of the set of points outside ellipsoid $E$, $V_E$ the volume of ellipsoid $E$, $\alpha$ a constant which is also $\alpha \geq |C|^2$, $LB$ the vector of lower bounds on all coordinates of points of $C$ and $UB$ the vector of upper bounds on all coordinates of points of $C$, and $c$ the centre of ellipsoid $E$.

\[
\text{if } (|C_E| < |C|) \quad \text{then} \\
\quad \phi_E = (|C| - |C_E| + V_E + \alpha)^2
\]
\[
\text{elseif } \quad \quad LB[i] < c[i] < UB[i], \quad \forall i \\
\quad \phi_E = |C_E| - V_E
\]
\[
\text{else} \\
\quad \phi_E = 1/(|C_E| - V_E)
\]

To reduce the computational burden, we generate the ellipsoids of the initial population having their centres within the lower and upper bounds of the coordinates of the points of set $C$. Note that, the ellipsoids which contain some or all of the points do not compete well with empty and large ones. The final one is the maximum volume ellipsoid. Figure 6 illustrates that. The results for small sets of points in different dimensions is shown in Table 8.

**Table 8. Performance of GA for MVEE and the exact approach**

| Algorithm | Genetic Algorithm | Path-Following Algorithm |
|-----------|-------------------|-------------------------|
| Dim Points | Border Time(s) | Gens | $\in$ | Volume | Volume Time(s) | Error(%) |
| 2 1000 | 151 | 100 | 0.04 | 155.448 | 154.863 | 0.3777 |
| 2 | 9.602 | | | | 195.231 | 5.641 |
| 10 | 177 | 70 | 0.015 | 312.540 | 295.851 | 1.0631e+04 |
| 20 | 23.873 | | | | 272.85 | 7.2136e+06 |
| 30 | 943 | 70 | 0.023 | 2.3054e+10 | 2.1482e+08 | * |
| 60 | 185.910 | | | | 60.846 | * |
| 100 | 1546 | 100 | 0.02 | 1.8120e+21 | 2.5119e+16 | * |
| 2 | 600.590 | | | | 304.817 | * |
| 100000 | 330 | 100 | 0.002 | 2.6415e+34 | * | * | * |
| 3 | 403.556 | | | | * | * | * |
| 10 | 823 | 100 | 0.003 | 2.2161e+85 | * | * | * |
| 20 | 1987.033 | | | | * | * | * |
| 60 | 465 | 100 | 0.0009 | 8.7299e+147 | * | * | * |
| 100 | 988.899 | | | | * | * | * |
| 2 | 414 | 100 | 0.01 | 156.719 | * | * | * |
| 500000 | 23.379 | | | | * | * | * |
| 3 | 942 | 70 | 0.0073 | 327.155 | * | * | * |
| 10 | 63.981 | | | | * | * | * |
| 20 | 410 | 300 | 0.002 | 1.1588e+14 | * | * | * |
| 30 | 646.600 | | | | * | * | * |
| 60  | 1515 | 100 | 0.05 | 7.4049e+11 | * | * | * |
|-----|------|-----|------|------------|---|---|---|
| 100 | 1194.008 | * | * | * |
| 2   | 1270 | 100 | 0.00049 | 1.8894e+85 | * | * | * |
| 100000 | 1437.099 | * | * | * |
| 3   | 1848 | 100 | 0.0004 | 7.0457e+147 | * | * | * |
| 10  | 4866.48 | * | * | * |
| 20  | 510 | 70 | 0.0025 | 157.415 | * | * | * |
| 30  | 23.467 | * | * | * |
| 60  | 1169 | 70 | 0.002 | 337.205 | * | * | * |
| 100 | 84.234 | 300 | 0.00065 | 7.5971e+10 | * | * | * |
| 537.260 | 3902 | 100 | 0.00095 | 4.9726e+21 | * | * | * |
| 3111.631 | 1954 | 100 | 0.0003 | 1.3715e+34 | * | * | * |
| 4527.244 | 3136 | 100 | 0.00024 | 1.7157e+85 | * | * | * |
| 3447.410 | 2151 | 100 | 0.0001 | 3.6392e+147 | * | * | * |
| 4044.211 | 307 | 100 | 0.0006 | 160.557 | * | * | * |
| 34.397 | 382 | 170 | 0.0003 | 558.389 | * | * | * |
| 144.237 | 425 | 200 | 0.0002 | 1.7173e+15 | * | * | * |
| 1538.542 | 2882 | 100 | 0.00035 | 2.7976e+23 | * | * | * |
| 931.156 | 5274 | 100 | 0.00044 | 1.9077e+39 | * | * | * |
| 6125.831 | 5362 | 100 | 0.00022 | 2.1627e+85 | * | * | * |
| 9992.16 | 2139 | 100 | 0.00005 | 1.6744e+147 | * | * | * |

**Figure 6.** Maximum volume ellipsoids embedded within points in 2d and 3d
9. Conclusion
We have looked at the L-J ellipsoid otherwise known as the MVEE problem and suggested a different approach to solving it which doesn't consider the standard convex programming formulation. The approach is a novel implementation of GA for the propose. This implementation works well for sets of points of low cardinalities in low dimensions. But, it also works for large sets in high dimensions, particularly if some adequate preprocessing is applied. For instance, by targeting only the relevant points, using the border strategy, we are able to improve the algorithm substantially. Computational results show that this approach is efficient. It produces feasible solutions in less time than other methods. We have also considered a related problem which seeks to compute the maximum volume ellipsoid embedded in a given set of points. Computational results show that our approach works well. Note that an implementation of GA with the border strategy, coupled with parallel processing will solve potentially very large problem instances in terms of cardinality of sets of points and space dimensions.

Acknowledgements:
We are grateful to the Iraqi Ministry of High Education for sponsoring this work.

| Dim | Points | Generations | Initial Pop | Time(s) |
|-----|--------|-------------|-------------|---------|
| 2   | 20     | 100         | 50          | 4.959   |
| 2   | 50     | 100         | 50          | 6.062   |
| 2   | 100    | 60          | 9.192       |
| 3   | 20     | 100         | 50          | 6.805   |
| 3   | 50     | 100         | 8.364       |
| 3   | 100    | 50          | 12.274      |
| 4   | 50     | 100         | 12.966      |
| 4   | 100    | 50          | 20.059      |
| 5   | 50     | 60          | 23.489      |
| 5   | 100    | 60          | 17.117      |
| 6   | 50     | 60          | 18.452      |
| 6   | 100    | 60          | 26.271      |
| 7   | 50     | 60          | 19.461      |
| 7   | 100    | 60          | 28.012      |
| 8   | 50     | 60          | 21.525      |
| 8   | 100    | 60          | 31.109      |
| 9   | 50     | 60          | 24.374      |
| 9   | 100    | 60          | 34.939      |
| 10  | 50     | 60          | 25.889      |
| 10  | 100    | 60          | 36.843      |
References

[1] ANSTREICHER, K. M. Ellipsoidal approximations of convex sets based on the volumetric barrier. *Mathematics of Operations Research* 24, 1 (1999), 193-203.

[2] BAKER, J. E. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the second international conference on genetic algorithms* (1987), pp. 14-21.

[3] BARNES, E. An algorithm for separating patterns by ellipsoids. IBM *Journal of Research and Development* 26, 6 (1982), 759-764.

[4] BLUM, S., PUISA, R., RIEDEL, J., AND WINTERMANTEL, M. Adaptive mutation strategies for evolutionary algorithms. In *The Annual Conference: EVEN at Weimarer Optimierungsund Stochastiktage* (2001), vol. 2.

[5] BOYD, S., AND VANDENBERGHE, L. *Convex optimization*. Cambridge university press, 2004.

[6] DYER, M., FRIEZE, A., AND KANNAN, R. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)* 38, 1 (1991), 1-17.

[7] GÄRTNER, B., AND SCHÖNHERR, S. Smallest enclosing ellipses-fast and exact. Proc. 13. *Annual ACM Symposium on Computational Geometry*, 430-432.

[8] GRANT, M., BOYD, S., AND YE, Y. CVX users guide, 2009.

[9] GRÖTSCHEL, M., LOVÁSZ, L., AND SCHRIJVER, A. *Geometric algorithms and combinatorial optimization*, vol. 2. Springer Science & Business Media, 2012.

[10] JOHN, F. Extremum problems with inequalities as subsidiary conditions. *Courant Anniversary Volume, Interscience, New York, 1948, Mathematical Institute, Hungarian Academy of Sciences 1053*, 187-204.

[11] KHACHIYAN, L. G. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research* 21, 2 (1996), 307-320.

[12] KHACHIYAN, L. G., and Todd, M. J. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming 61*, 1 (1993), 137-159.

[13] KUMAR, P., AND YILDIRIM, E. A. Minimum-volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications 126*, 1 (2005), 1-21.

[14] LENSTRA JR, H. W. Integer programming with a fixed number of variables. *Mathematics of Operations Research* 8, 4 (1983), 538-548.

[15] LOVÁSZ, L., AND SIMONOVITS, M. On the randomized complexity of volume and diameter. In *Foundations of Computer Science, 1992. Proceedings., 33rd Annual Symposium on Foundations of Computer Science* (1992), IEEE, pp. 482-492.

[16] SALHI, A., GLASER, H., AND DE ROURE, D. Parallel implementation of a genetic-programming based tool for symbolic regression. *Information Processing Letters* 66, 6 (1998), 299-307.

[17] SCHLIERKAMP-VOOSEN, D., AND MÜHLENBEIN, H. Predictive models for the breeder genetic algorithm. *Evolutionary Computation 1*, 1 (1993), 25-49.
[18] SEIDEL, R. Linear programming and convex hulls made easy. In Proceedings of the Sixth Annual Symposium on Computational Geometry (1990), ACM, pp. 211-215.

[19] SHARIR, M., AND WELZL, E. A combinatorial bound for linear programming and related problems. In STACS 92. Springer, 1992, pp. 567-579.

[20] SILVEY, S., AND TITTERINGTON, D. A geometric approach to optimal design theory. Biometrika 60, 1 (1973), 21-32.

[21] SUN, P., AND FREUND, R. M. Computation of minimum-volume covering ellipsoids. Operations Research 52, 5 (2004), 690-706.

[22] TITTERINGTON, D. Optimal design: Some geometrical aspects of Doptimality. Biometrika 62, 2 (1975), 313-320.

[23] TODD, M. J., AND YILDIRIM, E. A. On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. Discrete Applied Mathematics 155, 13 (2007), 1731-1744.

[24] VAIDYA, P. M. A new algorithm for minimizing convex functions over convex sets. Mathematical Programming 73, 3 (1996), 291-341.

[25] WELZL, E. Smallest enclosing disks (balls and ellipsoids). In New Results and New Trends in Computer Science: Graz, Austria, June 20-21, 1991 Proceedings (1991), vol. 555, Springer Science & Business Media, p. 359.

[26] WU, Y. Software engineering and knowledge engineering: theory and practice, vol. 2. Springer Science & Business Media, 2012.

[27] YILDIRIM, E. A. On the minimum volume covering ellipsoid of ellipsoids. SIAM Journal on Optimization 17, 3 (2006), 621-641.

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

[29] ZHANG, Y., AND GAO, L. On numerical solution of the maximum volume ellipsoid problem. SIAM Journal on Optimization 14, 1 (2003), 53-76.