Dinesh B. Ekanayake; Douglas J. LaFountain; Boris Petracovici
Finite convergence into a convex polytope via facet reflections

Applications of Mathematics, Vol. 68 (2023), No. 4, 387–404

Persistent URL: `http://dml.cz/dmlcz/151701`

68 (2023) APPLICATIONS OF MATHEMATICS No. 4, 387–404

FINITE CONVERGENCE INTO A CONVEX POLYTOPE VIA FACET REFLECTIONS

Dinesh B. Ekanayake, Douglas J. LaFountain,
Boris Petracovici, Macomb

Received June 11, 2022. Published online November 3, 2022.

Abstract. The problem of utilizing facet reflections to bring a point outside of a convex polytope to inside has not been studied explicitly in the literature. Here we introduce two algorithms that complete the task in finite iterations. The first algorithm generates multiple solutions on the plane, and can be readily utilized in creating games on a plane or as a level generation method for video games. The second algorithm is a new efficient way to bring infeasible starting points of an optimization problem to inside a feasible region defined by constraints. Using simulations, we demonstrate many desirable properties of the algorithm. Specifically, more edges do not lead to more iterations in $\mathbb{R}^2$, the algorithm is extremely efficient in high dimensions, and it can be employed to discretize the feasibility region using a grid of points outside the region.

Keywords: convex geometry; optimization; infeasible start; strategy games

MSC 2020: 52-08, 52B11, 52B12, 90C59

1. Introduction

In this paper, we consider bringing an arbitrary point outside of a convex polytope to inside, by performing only reflections over the hyperplanes defining the facets. We introduce two algorithms: a generating algorithm $\mathscr{A}$ that generates many choices to complete the task in $\mathbb{R}^2$, all in finitely many steps; and an algorithm $\mathscr{B}$ which also always generates finitely convergent sequences, but now in higher dimensions. Algorithm $\mathscr{A}$ is particularly useful in creating games that offer multiple choices to win. Algorithm $\mathscr{B}$ is a new efficient way to bring an arbitrary point to inside a feasible region defined by constraints, which is important for optimization algorithms such as interior point methods.

DOI: 10.21136/AM.2022.0134-22

Our immediate motivation is that optimization algorithms for constrained optimization problems need a feasible initial point from which to start. In linear programming, this is solved by either the two-phase approach or the big M method [4], [2]. In the two-phase approach, we start with an infeasible point and solve a secondary optimization problem to find a feasible point. In the big M method, we solve a modified and parametrized version of the original problem for which a feasible initial solution is known. For convex and nonlinear constrained optimization problems, interior-point methods and trust-region methods are some of the most important classes of algorithms. In such algorithms, a feasible point is obtained utilizing methods such as the two-phase approach [3] and the self-dual embedding techniques [8]. Significantly, all of these methods rely on secondary or modified optimization problems. In contrast, the reflection algorithms proposed in the current work are exceedingly simple, and computationally inexpensive, alternatives to find an interior point for linear inequality constrained problems with nonlinear objective functions. Most importantly, the proposed algorithms can be utilized in searching for a global optimal solution by constructing a mesh inside the feasible region.

Apart from initialization applications, the algorithms can be readily utilized in creating games on a plane using reflections over the edges of a polygon. Games may be created on a game board with tessellations generated by reflections of polygons or divisions generated by the extended edges of a polygon. Here, we give an example of such a game, and also indicate how the algorithm allows for an underlying level generation method for more complex video games.

With these two distinct applications in mind, the outline of the paper is as follows. In Section 2, we provide the algorithm $\mathscr{A}$ as a solution for the reflection problem in $\mathbb{R}^2$. We then prove finite termination for all the solutions generated by the algorithm. In Section 3, we provide the algorithm $\mathscr{B}$ as a solution for the reflection problem in $\mathbb{R}^p$, $p \in \mathbb{N}$. We show that in $\mathbb{R}^2$, $\mathscr{B}$ is in fact a particular implementation for $\mathscr{A}$. We further prove that the algorithm generates a finite sequence that guarantees that the end point is in the polytope. In Section 4, we present our applications for gaming, as well as initialization applications for some optimization problems to search for local and global solutions.

2. Algorithm in $\mathbb{R}^2$

We begin by referring to Figure 1, which indicates a gray set $P \subset \mathbb{R}^2$ as the convex hull of the points $V_1, V_2, \ldots, V_m$, each of which is a vertex for the boundary polygon, and which are ordered cyclically clockwise around ∂P. In notation that follows, whenever we consider consecutive indices i and $i+1$ for vertices or associated objects, this will be with the understanding that the cyclic ordering applies, so that if $i = m$, then $i+1 = 1$. With this in mind, each edge of ∂P joining adjacent vertices V_i, V_{i+1} has been extended to an infinite line $\ell_{i,i+1}$. Note that each line $\ell_{i,i+1}$ bounds an open half-plane which is disjoint from P as a subset; we will refer to this open half-plane as $H_{i,i+1}$. We then have two types of regions of interest.

For each $i \in \{1, \ldots, m\}$, define

$$C_i = H_{i-1,i} \cap H_{i,i+1}.$$

So C_i is an open cone not containing its vertex V_i, and examples are indicated in Figure 1; note that the various C_i may have nonempty intersection, but there is a neighborhood of V_i which is disjoint from any other C_j for $j \neq i$.

Along an edge of the polygon corresponding to a line $\ell_{i,i+1}$ we define the region

$$S_{i,i+1} = H_{i,i+1} \setminus (C_i \cup C_{i+1}).$$

So $S_{i,i+1}$ is a region adjacent to P, C_i and C_{i+1} as indicated in Figure 1, and is neither open nor closed, as it contains two of its boundary lines, but not the edge between V_i and V_{i+1}. The region $S_{i,i+1}$ may be bounded or unbounded depending on whether its included boundary lines intersect; regardless, by definition the various $S_{i,i+1}$ will have empty intersection with each other (see Figure 3).

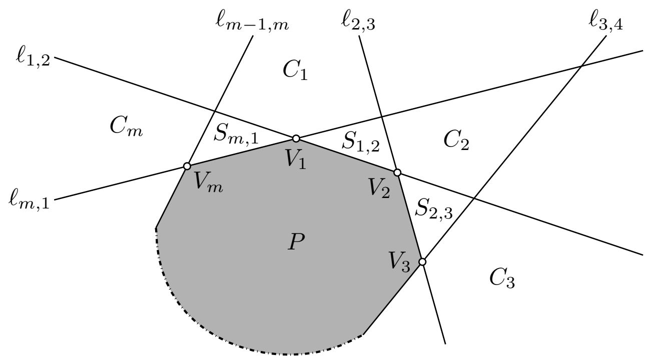


Figure 1. Regions of importance for the algorithm.

If a point x in $\mathbb{R}^2$ is reflected over one of the lines $\ell_{i,i+1}$, we will slightly abuse notation and refer to the image of x under that reflection as $\ell_{i,i+1}(x)$. We then define an iterative algorithm $\mathscr{A}$, which, given a seed $x_0 \in \mathbb{R}^2 \setminus P$, produces a family of sequences, all of which we will prove to be finite.

Definition 2.1 (The generating algorithm $\mathscr{A}$). Let $x_0 \in \mathbb{R}^2 \setminus P$; we have the following steps:

1. If $x_n \in S_{i,i+1}$, then $x_{n+1} = \ell_{i,i+1}(x_n)$; thus, this point is unique in the algorithm. If $x_n \in C_i$ for $i \in I$, where $I \subset \{1, \dots, m\}$, then we may choose $x_{n+1} = \ell_{i-1,i}(x_n)$ or $x_{n+1} = \ell_{i,i+1}(x_n)$. In other words, we may reflect x_n over either of the lines bounding C_i, and we may do this for any C_i containing x_n. We refer to the set of all possibilities for x_{n+1} as $\mathscr{A}(x_n)$.
2. If $x_{n+1} \in P$, stop. If not, set $n = n + 1$ and repeat Step 1.

Our goal is therefore to show that this algorithm $\mathscr{A}$ terminates for any sequence of points that it generates. To do so, denote by $d_i(x) = d(x, V_i)$ the distance from x to V_i. We then have the following lemma.

Lemma 2.2. *For any x_n in a sequence generated by $\mathscr{A}$ in $\mathbb{R}^2 \setminus P$, we have*

$$d_i(x_{n+1}) \leqslant d_i(x_n)$$

for all $i \in \{1, \dots, m\}$, and the inequality is strict for all but two $i \in \{1, \dots, m\}$.

P r o o f. In Step 1 of $\mathscr{A}$, if x_n is reflected over $\ell_{i,i+1}$ with $x_{n+1} = \ell_{i,i+1}(x_n)$, then $d_i(x_{n+1}) = d_i(x_n)$ and $d_{i+1}(x_{n+1}) = d_{i+1}(x_n)$ since $V_i, V_{i+1} \in \ell_{i,i+1}$. Thus, it remains to show that $d_j(x_{n+1}) < d_j(x_n)$ for $j \notin \{i, i+1\}$. To see this, first observe that Step 1 of Definition 2.1 ensures us that x_n does not lie on $\ell_{i,i+1}$, for neither of C_i, C_{i+1} nor $S_{i,i+1}$ contain points on $\ell_{i,i+1}$. We also know that since P is convex, V_j lies on the same side of $\ell_{i,i+1}$ as x_{n+1}; we thus have the following Figure 2. Specifically, line segment $\overline{V_j x_n}$ must intersect $\ell_{i,i+1}$ at the point indicated y, and we can construct line segment $\overline{y x_{n+1}}$ which must necessarily lie inside the convex $\Delta V_j x_n x_{n+1}$. From Euclid it is known that proving $d_j(x_{n+1}) < d_j(x_n)$ is equivalent to proving $\alpha < \beta$ in $\Delta V_j x_n x_{n+1}$. Since $x_{n+1} = \ell_{i,i+1}(x_n)$, we know that $\ell_{i,i+1}$ is the perpendicular bisector of line segment $\overline{x_n x_{n+1}}$, so that we must have that $\alpha = \angle x_n x_{n+1} y$. Since the whole is greater than the part, we conclude that $\beta = \angle x_n x_{n+1} V_j > \angle x_n x_{n+1} y = \alpha$. This proves the lemma. $\square$

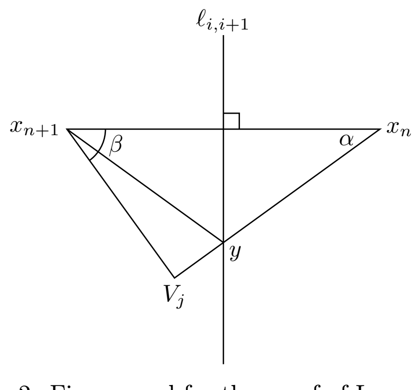


Figure 2. Figure used for the proof of Lemma 2.2.

Define

$$D(x) = \sqrt{d_1(x)^2 + \ldots + d_m(x)^2}. \tag{2.1}$$

Since $|D(x_n) - D(x)| \to 0$ as $x_n \to x$, $D(x)$ is continuous on $\mathbb{R}^2$. We then have the following important corollary of Lemma 2.2, whose proof is immediate.

Corollary 2.3. *If $x \in \mathbb{R}^2 \setminus P$ and $y \in \mathscr{A}(x)$, then $D(y) < D(x)$.*

We will ultimately prove that $\mathscr{A}$ is finite by contradiction, and therefore we first examine infinite sequences generated by $\mathscr{A}$, setting some notation.

Let $\{x_n\}$ be a sequence generated by $\mathscr{A}$. If $\{x_{n_s}\}_{s=0}^{\infty}$ is any subsequence of $\{x_n\}$, the subsequence $\{x_{n_s+1}\}$ satisfies $x_{n_s+1} \in \mathscr{A}(x_{n_s})$ for all s. Since $\mathscr{A}$ uses only finitely many reflections, and there are infinitely many terms in this subsequence, there must be a further subsequence of $\{x_{n_s}\}$, which we will denote simply as $\{x_k\}$, and associated further subsequence of $\{x_{n_s+1}\}$, which we denote simply as $\{\ell(x_k)\}$, where the terms $\ell(x_k)$ are coming from a fixed reflection ℓ of the x_k. We will call $\{\ell(x_k)\}$ the *reflection* of $\{x_k\}$ through ℓ, and observe we can always pass to such a subsequence and its reflection.

With this notation and definition, the following proposition will be essential.

Proposition 2.4. *If $\{x_n\}$ is an infinite sequence in $\mathbb{R}^2 \setminus P$ generated by $\mathscr{A}$, then any convergent subsequence of $\{x_n\}$ converges to a point in ∂P.*

Proof. Let $\{x_k\}$ be a convergent subsequence of $\{x_n\}$, where we may assume it is paired with its reflection $\{\ell(x_k)\}$ through ℓ, and suppose for contradiction that $\{x_k\}$ converges to a point $x^* \in \mathbb{R}^2 \setminus P$. Then since the reflection across ℓ is a continuous transformation of $\mathbb{R}^2$, the subsequence $\{\ell(x_k)\}$ must converge to $\ell(x^*)$. We note that although the action of ℓ on x_k is coming from the algorithm $\mathscr{A}$, we do not know that the action of ℓ on x^* is from $\mathscr{A}$; that is, ℓ might not be applicable to x^* as prescribed by $\mathscr{A}$ due to the location of x^*. This will be the key question to resolve throughout this proof.

We first assume that $\{x_k\}$ is in a region $S_{i,i+1}$ for some i. Then we know from Step 1 of the algorithm $\mathscr{A}$ that $\ell = \ell_{i,i+1}$, and since $S_{i,i+1}$ contains its two boundary edges disjoint from P, we know x^* is in $S_{i,i+1}$ as well. Then the action of ℓ on x^* is coming from the algorithm $\mathscr{A}$, so that $\ell(x^*) \in \mathscr{A}(x^*)$, so by Corollary 2.3, $D(\ell(x^*)) < D(x^*)$. But by the same corollary, D is monotonic decreasing on the entire sequence $\{x_n\}$, therefore $\{D(x_n)\}$ must be convergent in the positive real numbers, and must converge to $D(x^*)$. Since this is true for any subsequence of $\{D(x_n)\}$, we obtain $D(\ell(x^*)) = D(x^*)$, contradicting $D(\ell(x^*)) < D(x^*)$. This proves the proposition in this case.

Thus, if we can always show $\ell(x^*) \in \mathscr{A}(x^*)$, for either this subsequence or a new convergent subsequence of $\{x_n\}$ and its reflection, we will be able to reach the contradiction $D(\ell(x^*)) = D(x^*)$ and $D(\ell(x^*)) < D(x^*)$, and the proof of the proposition will be complete. With this in mind, for the remainder of the proof we may assume without loss of generality that $\{x_k\}$ is in a region C_i, that $\ell \in \{\ell_{i-1,i}, \ell_{i,i+1}\}$, and the limit point x^* is on one of the boundary lines $\ell_{i-1,i}$ or $\ell_{i,i+1}$; see Figure 3, where we show both the cases where $S_{i,i+1}$ is bounded in part (a), and where it is unbounded in part (b). Without loss of generality we will assume x^* is on $\ell_{i-1,i}$, indicated by the $\star$ in Figure 3, and we observe from that figure along with Figure 1 and the accompanying definitions that we know $x^* \in C_{i+1} \cup S_{i,i+1}$. If $\ell = \ell_{i,i+1}$, then since $x^* \in C_{i+1} \cup S_{i,i+1}$, in fact $\ell_{i,i+1}$ acts on x^* via $\mathscr{A}$, so that again we can still conclude $\ell(x^*) \in \mathscr{A}(x^*)$, and we are done.

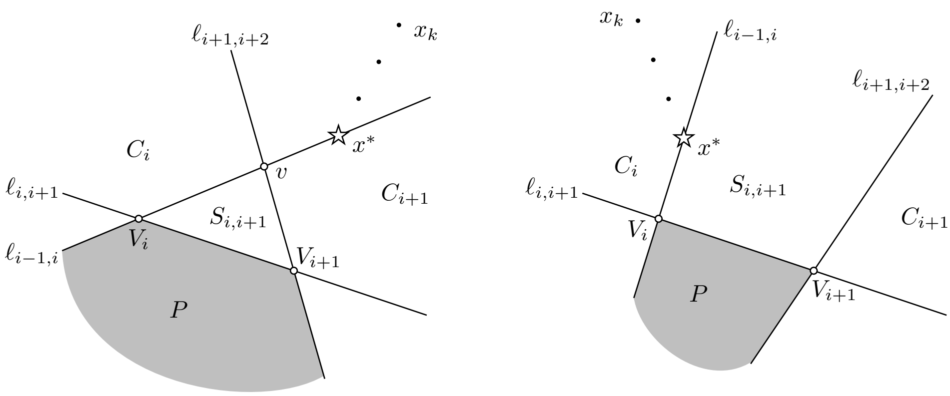
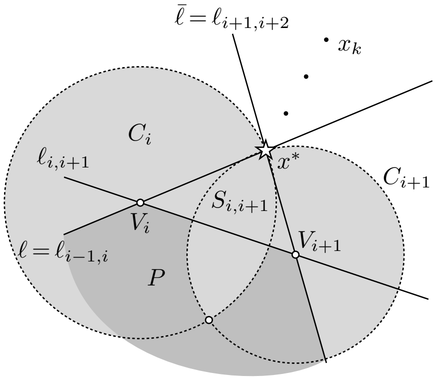


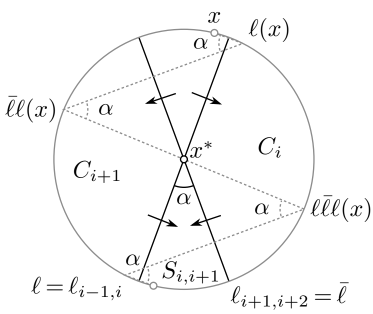
Figure 3. First figure used for the proof of Proposition 2.4.

Thus, we only need to consider the case where $\ell = \ell_{i-1,i}$. Since x^* is on $\ell_{i-1,i}$, then $x^* = \ell(x^*)$. We will first assume that x^* is distinct from the point v indicated in Figure 3 (a), where the lines $\ell_{i-1,i}$ and $\ell_{i+1,i+2}$ may intersect. In this case, eventually the reflection sequence $\{\ell(x_k)\}$ will be contained in the same region R as $\ell(x^*)$, where $R \in \{C_{i+1}, S_{i,i+1}\}$. Then we can pass to a subsequence of $\{\ell(x_k)\}$, which we call $\{y_k\}$ converging to $y^* = \ell(x^*)$, but where $\{y_k\}$ is now itself paired with its reflection $\{\bar{\ell}(y_k)\}$ converging to $\bar{\ell}(y^*)$ via the reflection $\bar{\ell}$, which is a boundary line for R. The key is that since $y^*, \{y_k\} \in R$, the action of $\bar{\ell}$ on y^* comes from $\mathscr{A}$, so that $\bar{\ell}(y^*) \in \mathscr{A}(y^*)$, and again we are done since these are all subsequences of $\{x_n\}$.

Finally, we consider the case where $x^* = v$ in Figure 4 (a), where for convenience of notation we will set $\bar{\ell} = \ell_{i+1,i+2}$. Note that the only reflection of x^* compatible with $\mathscr{A}$ is across $\ell_{i,i+1}$. Observe there are open discs centered at V_i and V_{i+1}, whose boundary circles pass through x^*, and since by Lemma 2.2 we know $d_j(x_n) \geqslant d_j(x^*)$ for all $j \in \{1, \ldots, m\}$, no element in $\{x_n\}$ can be contained in these discs. We still are under the assumption that $\ell = \ell_{i-1,i}$ as in the paragraph above, so that the sequence $\{\ell(x_k)\}$ will be contained in C_{i+1}, since the region $S_{i,i+1} \setminus \{x^*\}$ is necessarily covered by the two discs. Yet the argument in the preceding paragraph does not immediately work; the key obstacle is that if we pass to a subsequence of $\{\ell(x_k)\}$ and its reflection, this reflection could be across $\bar{\ell}$ back into C_i, which is still incompatible with $\mathscr{A}$ for x^*. Thus, a priori we could alternate reflections across ℓ and $\bar{\ell}$ ad infinitum; our goal is to prove this cannot happen.



(a)



(b)

Figure 4. Second figure used for the proof of Proposition 2.4.

We zoom in on x^* in Figure 4 (b), where we have oriented the figure so that the vertical line passing through x^* bisects the angle $\alpha > 0$ in $S_{i,i+1}$ formed by $\ell_{i-1,i}$ and $\ell_{i+1,i+2}$. We take an arbitrary point $x \in C_i$ at some small distance r from x^*, and consider alternately reflecting x over ℓ, and then $\ell(x)$ over $\bar{\ell}$, and continue alternating between reflections over ℓ and $\bar{\ell}$. We have two observations. First, the images of x under these alternating reflections must stay on the circle of radius r centered at x^*, but must also proceed monotonically downward toward the region $S_{i,i+1}$, since at every reflection they proceed parallel to the downward directed arrows perpendicular to ℓ and $\bar{\ell}$ as indicated in Figure 4 (b). Moreover, as indicated in Figure 4 (b), the angle subtended by the paths of two successive reflections must equal α, so that by Euclid, the central angle subtended by the point x and its subsequent image under two successive reflections is equal to 2α. As a result, we conclude that after at most $\lceil \frac{\pi}{\alpha} \rceil$ reflections, the image of x must land in $S_{i,i+1}$, an example of which is shown in Figure 4 (b).

We can now conclude the proof of the proposition. With $\{x_k\}$ in C_i, $x^* = v$, and $\ell = \ell_{i-1,i}$, we know $\{\ell(x_k)\}$ is in C_{i+1}, and we observe that if we repeatedly pass to subsequences and their reflections, where these reflections alternate between

$\ell = \ell_{i-1,i}$ and $\bar{\ell} = \ell_{i+1,i+2}$, this can happen at most $\lceil \frac{\pi}{\alpha} \rceil$ times, since we know that $S_{i,i+1}$ is covered by the two discs in Figure 4 (a). Thus, at some point, when we pass to a subsequence and its reflection, that reflection must be over $\ell_{i,i+1}$, and then since $x^* \in S_{i,i+1}$, the action of $\ell_{i,i+1}$ on x^* will be due to the algorithm $\mathscr{A}$.

Therefore, we can always find $\ell(x^*) \in \mathscr{A}(x^*)$ for some subsequence of $\{x_n\}$ and this concludes the proof of the proposition. □

We can now prove our main theorem.

Theorem 2.5. *If $\{x_n\}$ is a sequence generated by $\mathscr{A}$ in $\mathbb{R}^2 \setminus P$, then $\{x_n\}$ is a finite sequence, meaning at some point it enters into P.*

Proof. Suppose for contradiction that $\mathscr{A}$ generates an infinite sequence $\{x_n\}_{n=0}^{\infty}$ in $\mathbb{R}^2 \setminus P$. By Lemma 2.2 this infinite sequence is bounded within a compact set, and thus must have a convergent subsequence $\{x_{n_s}\}_{s=0}^{\infty}$. Suppose $x_{n_s} \mapsto x^*$ as $s \to \infty$. By Proposition 2.4, we know $x^* \in \partial P$. If $x^* \notin \{V_1, \ldots, V_m\}$, then it must be the case that there is an open ball $B_\varepsilon(x^*)$ which is entirely contained in $S_{i,i+1} \cup P$ for some $i \in \{1, \ldots, m\}$. Since $x_{n_s} \mapsto x^*$, for some $s \geqslant 0$ we must have $x_{n_s} \in B_\varepsilon(x^*) \cap S_{i,i+1}$, with the result that the algorithm $\mathscr{A}$ will reflect x_{n_s} over $\partial P \cap B_\varepsilon(x^*)$ to lie in P, contradicting the assumption that $\{x_n\}_{n=0}^{\infty}$ lies in $\mathbb{R}^2 \setminus P$. As a result it must be that $x^* = V_i$ for some $i \in \{1, \ldots, m\}$.

So suppose finally for contradiction that $x_{n_s} \mapsto V_i$. First observe that there is a non-zero minimum distance δ between V_i and any points in $Q := \bigcup_{j \neq i} C_j \cup \bigcup_{j \notin \{i-1,i\}} S_{j,j+1}$ (refer to Figure 1 and its discussion). Then there is an $\varepsilon > 0$ with $\varepsilon < \delta$ such that the open ball $B_\varepsilon(V_i)$ is disjoint from Q. Since $x_{n_s} \mapsto V_i$, for some $s \geqslant 0$ we must have $x_{n_s} \in B_\varepsilon(V_i) \setminus P$, with $0 < d_i(x_{n_s}) < \varepsilon$. However, now observe that if $t > 0$, the point x_{n_s+t} in the infinite sequence $\{x_n\}$ generated by $\mathscr{A}$ cannot be in Q, for then we would have $d_i(x_{n_s+t}) \geqslant \delta > \varepsilon > d_i(x_{n_s})$, which would violate Lemma 2.2. Thus, for all $t > 0$, the points $x_{n_s+t} \in C_i \cup S_{i-1,i} \cup S_{i,i+1}$, and thus are generated by $\mathscr{A}$ via reflections over the lines $\ell_{i-1,i}$ or $\ell_{i,i+1}$, both of which contain V_i. As a result, from the proof of Lemma 2.2 we must have that $d_i(x_{n_s+t}) = d_i(x_{n_s}) > 0$ for all $t > 0$, meaning our subsequence $\{x_{n_s}\}$ cannot converge to V_i, which by assumption it does. This is a contradiction and proves the theorem. □

To conclude this section, we provide an initial proposition concerning the number of steps needed in algorithm $\mathscr{A}$. Let $\operatorname{diam}(P)$ be the diameter of the convex polygon P, and let $d(x, P)$ be the distance from a point $x \in \mathbb{R}^2$ to P.

Proposition 2.6. *The number of iterations needed for a sequence $\{x_n\}$ generated by $\mathscr{A}$ before it enters P is at least $\lceil d(x_0, P)/\operatorname{diam}(P) \rceil$.*

Proof. We will show that given a point x_n in the sequence $\{x_n\}$ generated by $\mathscr{A}$, we have $|d(x_n, P) - d(x_{n+1}, P)| \leqslant \operatorname{diam}(P)$. Then $d(x_0, P) \leqslant \sum_{n=0}^{N-1} |d(x_n, P) - d(x_{n+1}, P)| \leqslant N\operatorname{diam}(P)$ whenever $x_N \in P$, which gives us the desired result that the number of iterations in the sequence is at least $\lceil d(x_0, P)/\operatorname{diam}(P) \rceil$.

To this end, suppose $\mathscr{A}$ reflects x_n over a line $\ell_{i,i+1}$ to obtain x_{n+1}. Then consider $V_i \in \partial P$, one of the vertices on $\ell_{i,i+1}$, and observe that $d(x_n, V_i) = d(x_{n+1}, V_i)$. Let $q \in \partial P$ be the point such that $d(x_n, P) = d(x_n, q)$, and let $q' \in \partial P$ be the point such that $d(x_{n+1}, P) = d(x_{n+1}, q')$. Using the triangle inequality we obtain

$$d(x_n, P) \leqslant d(x_n, V_i) = d(x_{n+1}, V_i) \leqslant d(x_{n+1}, q') + d(q', V_i).$$

Substituting $d(x_{n+1}, P)$ for $d(x_{n+1}, q')$ and rearranging yields

$$d(x_n, P) - d(x_{n+1}, P) \leqslant d(q', V_i) \leqslant \operatorname{diam}(P),$$

where the last inequality is because P is convex.

Interchanging the roles of x_n and x_{n+1}, and q and q' in the above calculation yields

$$d(x_{n+1}, P) \leqslant d(x_{n+1}, V_i) = d(x_n, V_i) \leqslant d(x_n, q) + d(q, V_i),$$

so that substituting $d(x_n, P)$ for $d(x_n, q)$ and rearranging leads us to conclude

$$d(x_{n+1}, P) - d(x_n, P) \leqslant d(q, V_i) \leqslant \operatorname{diam}(P).$$

Thus, $|d(x_n, P) - d(x_{n+1}, P)| \leqslant \operatorname{diam}(P)$ and the proposition is proved. □

We note that we will observe in Example 4.3 that this least bound for the length of the sequence for any given $x_0 \in \mathbb{R}^2$ may be attained for a regular n-gon as $n \to \infty$.

3. Implementation in $\mathbb{R}^p$

Algorithm $\mathscr{A}$ is useful in making applications that offer multiple choices to achieve the same objective. For optimization applications, extending the implementation of $\mathscr{A}$ to higher dimensions is important. In this section, given intersecting half-spaces in $\mathbb{R}^p$ defining a polytope P, we introduce a modified iterative algorithm $\mathscr{B}$, where in the special case of $p = 2$ we obtain $\mathscr{B}(x) \subset \mathscr{A}(x)$.

Let $P \subset \mathbb{R}^p$ be a convex polytope of dimension p. Let $P_i = \{x \in \mathbb{R}^p \colon\ w_i^\top x + b_i \leqslant 0\}$, for $i = 1, \ldots, m$, be the facet-defining closed half-spaces corresponding to the minimal half-space representation of P, where $m > p$. We thus have

$$P = \bigcap_{i=1}^{m} P_i.$$

We observe that the complement of P_i is an open half-space $H_i = \{x \in \mathbb{R}^p : w_i^\top x + b_i > 0\}$, and if $x_0 \in H_i$, then the function

$$f_i(x) = \frac{w_i^\top x + b_i}{\sqrt{w_i^\top w_i}}$$

is positive at x_0 and $f_i(x_0)$ is in fact the distance from x_0 to the hyperplane $w_i^\top x + b_i = 0$. We note that for any $x_0 \in \mathbb{R}^p \setminus P$, $f_i(x_0) > 0$ for at least one $i \in \{1, \ldots, m\}$.

We also recall a useful function for reflecting x_0 across the hyperplane $w_i^\top x + b_i = 0$; namely, if we choose a point y_i on that hyperplane, then the reflection of x_0 across $w_i^\top x + b_i = 0$ is given by

$$r_i(x_0) = x_0 - 2\frac{w_i^\top (x_0 - y_i)}{w_i^\top w_i} w_i.$$

We can therefore define the following algorithm $\mathscr{B}$.

Definition 3.1 (The modified algorithm $\mathscr{B}$)**.** Let $x_0 \in \mathbb{R}^p \setminus P$, and choose a point y_i on each hyperplane $w_i^\top x + b_i = 0$ for $i = 1, \ldots, m$. Setting $n = 0$, we then have the following steps:

1. Let
$$j_n = \underset{i \in \{1,\ldots,m\}}{\arg\max}\ f_i(x_n),$$
where we observe that j_n may not necessarily be unique if x_n is in multiple H_i's and is maximally equidistant from two or more hyperplanes. Set $x_{n+1} = r_{j_n}(x_n)$.
2. If $x_{n+1} \in P$, stop. If not, set $n = n + 1$ and repeat Step 1.

For the purposes of the implementation of $\mathscr{B}$ in Section 4 for examples and applications, in Step 1 we will choose the minimum index $j_n \in \underset{i \in \{1,\ldots,m\}}{\arg\max} f_i(x_n)$ at each step of the algorithm so as to generate a single sequence.

We now prove that the modified algorithm $\mathscr{B}$ is in fact a particular implementation of the algorithm $\mathscr{A}$ for the case of $p = 2$.

Proposition 3.2. *If $x_n \in \mathbb{R}^2 \setminus P$, then $\mathscr{B}(x_n) \subset \mathscr{A}(x_n)$.*

Proof. In $\mathbb{R}^2$, each hyperplane $w_i^\top x + b_i = 0$ is the line $\ell_{i,i+1}$, and the region H_i corresponds to $H_{i,i+1}$; refer to Figure 1 and its discussion. Observe that if $x_n \in S_{i,i+1}$, then x_n is only in H_i and no other H_j, so that only $f_i(x_n) > 0$, leading us to conclude that $j_n = i$. In this case, the reflection of x_n in the algorithm $\mathscr{B}$ is guaranteed to be over $\ell_{i,i+1}$, just as in the algorithm $\mathscr{A}$. On the other hand, if

$x_n \in C_i$, then at least $f_{i-1}(x_n) > 0$ and $f_i(x_n) > 0$, but x_n may be in other C_j's as well. In fact, if $j_n \in \underset{i\in\{1,\ldots,m\}}{\arg\max}\ f_i(x_n)$, then it must be the case that $x_n \in C_{j_n}$, so that the reflection over the hyperplane $w_{j_n}^\top x + b_{j_n} = 0$ in the algorithm $\mathscr{B}$ is in fact reflection over the line ℓ_{j_n,j_n+1} in the algorithm $\mathscr{A}$. Therefore, if $p = 2$, we have $\mathscr{B}(x_n) \subset \mathscr{A}(x_n)$. □

Next we show that any sequence generated by $\mathscr{B}$ is finitely terminal in general $\mathbb{R}^p$. In doing so, we will follow the same line of argumentation as in Section 2, but with appropriate adjustments for our modified algorithm $\mathscr{B}$ in $\mathbb{R}^p$.

Theorem 3.3. *If $\{x_n\}$ is a sequence generated by $\mathscr{B}$ in $\mathbb{R}^p \setminus P$, then it is a finite sequence.*

Proof. Suppose for contradiction that $\{x_n\}$ is an infinite sequence generated by $\mathscr{B}$ in $\mathbb{R}^p \setminus P$. We first show that the function $D(x)$ as defined in equation (2.1) is monotonically decreasing on $\{x_n\}$, mimicking Lemma 2.2 and Corollary 2.3. If $x_n \notin P$, then from the algorithm $\mathscr{B}$ we have $f_{j_n}(x_n) > 0$. Take any vertex V_i that is not on the hyperplane $w_{j_n}^\top x + b_{j_n} = 0$. Then since P is convex, we have $f_{j_n}(V_i) < 0$, and since x_{n+1} is the reflection of x_n over the hyperplane $w_{j_n}^\top x + b_{j_n} = 0$, we have that $f_{j_n}(x_{n+1}) < 0$ as well, so that x_{n+1} lies on the same side of $w_{j_n}^\top x + b_{j_n} = 0$ as V_i. Consider the plane passing through x_n, x_{n+1} and V_i. Following Figure 2, we obtain $d_i(x_{n+1}) < d_i(x_n)$. Furthermore, every vertex on the hyperplane $w_{j_n}^\top x + b_{j_n} = 0$ is equidistant from x_n and x_{n+1}. We conclude that $D(x_{n+1}) < D(x_n)$.

Since $D(x_n) < D(x_0)$ for all n, the infinite sequence $\{x_n\}$ is bounded within a compact set and there exists a convergent subsequence $\{x_k\}$ of $\{x_n\}$ that converges to some x^*. As in Proposition 2.4, we want to first show that $x^* \in \partial P$, and we will see that the definition of $\mathscr{B}$ makes this straightforward. To this end, assume for contradiction that $x^* \in \mathbb{R}^p \setminus P$. As in Section 2, we may assume the subsequence $\{x_k\}$ is paired with its reflection $\{r_j(x_k)\}$ through a fixed hyperplane $w_j^\top x + b_j = 0$, with $r_j(x_k) \to r_j(x^*)$, and where now in the algorithm $\mathscr{B}$ we know that the fixed function f_j is maximal for x_k for all $k \in \mathbb{N}$. Since the f_i's are continuous functions, as $x_k \to x^*$, we must have $f_i(x_k) \to f_i(x^*)$, and we conclude that f_j must be maximal for x^* as well. Thus, $r_j(x^*)$ is actually coming from the algorithm $\mathscr{B}$ and we conclude that $D(r_j(x^*)) < D(x^*)$; however, since both $\{x_k\}$ and $\{r_j(x_k)\}$ are subsequences of $\{x_n\}$, and D is monotonically decreasing on the entire sequence, we also conclude $D(r_j(x^*)) = D(x^*)$. This is the desired contradiction, proving that $x^* \in \partial P$.

We therefore must have $x^* \in \partial P$ and $w_i^\top x^* + b_i \leqslant 0$ for all i; we work toward our final contradiction to prove the theorem. Let J be the largest subset of $\{1, \ldots, m\}$ such that $w_i^\top x^* + b_i = 0$ for all $i \in J$. Note that since $m > p$, J is a proper subset, and J is nonempty since $x^* \in \partial P$. Define $U = \bigcap_{i \notin J} \{x\colon\ w_i^\top x + b < 0\}$; since U is open and $x^* \in U$, there exists an open ball $B_\varepsilon(x^*)$ such that $B_\varepsilon(x^*) \subset U$. Since $x_k \to x^*$, there exists some N such that $x_N \in B_\varepsilon(x^*) \setminus P$. Let x_{N+1} be the reflection of x_N over the hyperplane $w_i^\top x + b = 0$, where the algorithm $\mathscr{B}$ and the definition of U guarantee that $i \in J$. As a result, we know that x^* lies on the hyperplane $w_i^\top x + b = 0$, so that x_N and x_{N+1} are equidistant from x^*, so that $x_{N+1} \in B_\varepsilon(x^*) \subset U$. Inducting on $t \in \mathbb{N}$ we can conclude that the algorithm $\mathscr{B}$ guarantees that $d(x^*, x_{N+t}) = d(x^*, x_N) > 0$ for all $t > 0$. We conclude that x^* is not a limit point of $\{x_k\}$, contradicting our original assumption that $\{x_n\}$ is an infinite sequence. Therefore, it is a finite sequence. □

The minimal half-space representation of P is not necessary. We may add any number of additional half-spaces containing P. Eventually they are not indexed in the set J and $\mathscr{B}$ will not utilize them thereafter. We also observe that the proof of Proposition 2.6 does not depend on $\mathbb{R}^2$ and holds in $\mathbb{R}^p$ for the algorithm $\mathscr{B}$, with reflection over hyperplanes instead of lines. We thus have the following corollary, where now $\operatorname{diam}(P)$ is the diameter of the convex polytope P.

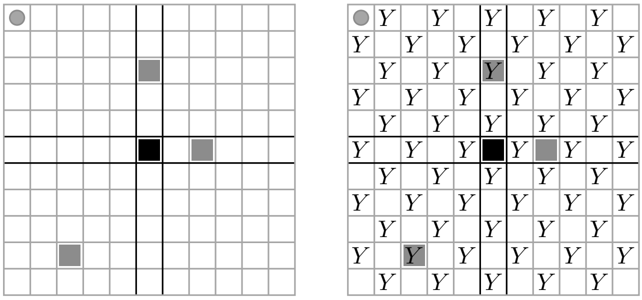
Corollary 3.4. *The number of iterations needed for a sequence $\{x_n\}$ generated by $\mathscr{B}$ before it enters $P \subset \mathbb{R}^p$ is at least $\lceil d(x_0, P)/\operatorname{diam}(P) \rceil$.*

4. Applications

In this section, we discuss some examples of applications of algorithms $\mathscr{A}$ and $\mathscr{B}$. All computations here were carried out on a 64-bit laptop with a Core i7 - 4 core processor at 1.8 GHz and 8 GB memory. We first consider two examples of how choices generated by $\mathscr{A}$ can create intriguing game play easily programmed for an app or video game.

Example 4.1 (A simple two-player game). Consider the two-player game board given in Figure 5 (a). The goal is to move the disc to the black square. Suppose algorithm $\mathscr{A}$ defines the set of rules for the game; the players alternate moving the same disc, and the player who makes the final move that lands the disc in the black square wins. The gray squares are randomly generated by a computer program or placed by the players on possible solution paths. If a player ends up in such a square, that player gets an extra turn.

The game always ends in exactly 10 moves, since if we label each square with an integer coordinate (m, n) for $-5 \leqslant m,\ n \leqslant 5$, any move reduces exactly one of $|m|$ or $|n|$ by 1. Because of this, it is easy to identify the squares having winning



(a) (b)

Figure 5. In part (a) is the game board, and in part (b) are winning positions.

positions, as shown in Figure 5 (b). However, the number and placement of the gray squares adds additional complexity. For example, with the given placement of gray squares, it is a perfect game for the starting Player 1, even though its starting position is a losing one. A perfect game means that, for any move of Player 2, there exists a move for Player 1 that guarantees the win. In this case, the strategy for Player 1 to guarantee a win is two-fold: avoid the bottom left gray square, and force one of the (m, n) coordinates to 0. Once that occurs, either Player 1 or Player 2 must land on one of the remaining two gray squares and move twice, and in doing so Player 1 secures a guaranteed winning position.

However, another example of complexity created by the gray squares is that Player 1 could adopt a strategy where they move so as to land on the bottom left gray square to earn an extra move and switch to a winning square. However, if that happens, the game actually becomes a perfect game for Player 2, as then they can always make a move that forces one of the (m, n) coordinates to 0. Once that occurs, either Player 1 or Player 2 must move twice, and in doing so Player 2 secures the guaranteed winning position.

Example 4.2 (Underlying domains for video games). Another example is the use of algorithm $\mathscr{A}$ with an associated convex polygon P as an underlying procedural generation method to create rich paths through levels for replayable video games. In this application, the underlying architecture for the various levels of a video game would be a convex polygon P, which would be the final level of the game, with each $S_{i,i+1}$ and C_i (and intersections thereof) a level that the gamer must pass once in it. Upon completion of a level, either the gamer uses an interface to intentionally choose, or the computer randomly chooses, which reflection to use to move to the subsequent level. Provided the gamer continues to pass levels, they are guaranteed to arrive at the final level P. This could allow for a rich variety of strategies, and be more complex than a simple linear path of levels.

With the rest of the examples, we compare the performance of the algorithm $\mathscr{B}$ as a phase 1 method for the two-phase approach to solve optimization problems with infeasible starting points. Since algorithm $\mathscr{B}$ does not incorporate the objective function, we compare the performance with the phase 1 method given by

$$\begin{aligned} &\text{minimize} \quad z \\ &\text{subject to} \quad w_i^\top x + b_i \leqslant z, \quad i = 1, \ldots, m. \end{aligned} \tag{4.1}$$

Problem 4.1 is the minimization problem that drives the maximum z below zero, where z is interpreted as a bound on the maximum infeasibility of the inequalities. We solve it using MATLAB's interior point linear program algorithm and the dual simplex algorithm. Next we include an example on $\mathbb{R}^2$ to demonstrate the impact of the number of edges and the distance between the point and the polygon.

Example 4.3. (Regular convex polygons in $\mathbb{R}^2$) We create regular convex polygons with center at origin and n sides to bring a search point x_0 that is not in the polygon to inside the polygon. We set the area of each polygon to 25π so that as $n \to \infty$, the polygons converge to a circle of radius 5. By changing n and x_0, we compare the number of iterations for algorithm $\mathscr{B}$ to that of the interior point and dual simplex methods to solve Problem 4.1. The results are tabulated in Table 1. Figure 6 shows $\mathscr{B}$ solution paths for $x_0 = (60, 0)$.

For algorithm $\mathscr{B}$, more edges leads to fewer iterations. Further, the ratio between the distance to the polygon and the diameter of the converging circle matches the number of iterations for larger values of n. In contrast, changes in x_0 do not result in any changes for the other methods, except that more edges leads to more iterations. Also, the largest elapsed time for algorithm $\mathscr{B}$ is about 0.0004 seconds, whereas the interior point and dual simplex methods take at least 0.003 and 0.008 seconds, respectively. During the average elapsed time it takes for other methods to complete one iteration, $\mathscr{B}$ can complete as many as 100 iterations.

Edges	3	5	7	11	13	17	3	5	7	11	13	17
	$x_0 = (20, 0)$						$x_0 = (40, 0)$					
Algorithm $\mathscr{B}$	4	3	2	2	2	2	7	5	4	4	4	4
Interior point	3	6	10	9	9	12	3	6	10	9	9	12
Dual simplex	3	4	6	8	9	10	3	4	6	8	9	10
	$x_0 = (60, 0)$						$x_0 = (80, 0)$					
Algorithm $\mathscr{B}$	11	7	7	6	6	6	14	10	9	8	8	8
Interior point	3	6	10	9	9	12	3	6	10	9	9	12
Dual simplex	3	4	6	8	9	10	3	4	6	8	9	10

Table 1. Number of iterations to bring x_0 to inside the polygon for algorithm $\mathscr{B}$ and for the linear program with the interior point and dual simplex methods.

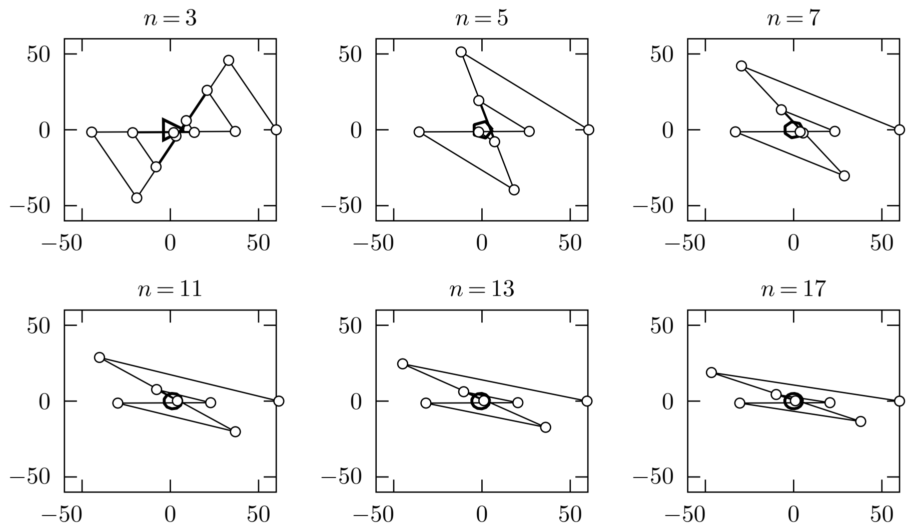


Figure 6. Reflections of $x_0 = (60, 0)$ over the edges of regular convex polygons with n sides.

As we see in Example 4.3, we may expect fewer iterations for a greater number of constraints with algorithm $\mathscr{B}$ since we have more reflections from which to choose. This is highly desirable and in contrast to other methods. The simplex algorithm is almost always $O(\max(n,p))$ for the number of iterations, where p is the number of variables and n is the number of constraints [6]. Primal-dual interior point methods converge to an ε-accurate solution with a worst case bound of $O(\sqrt{n+p}\ln(1/\varepsilon))$ for the number of iterations of the method [7]. Therefore, we expect an increase in iterations with more constraints for linear programming problems. The same observation may be an indication that the worst case problem for $\mathscr{B}$ may correspond to simplexes in higher dimensions. In the next example, we compare the algorithms for Klee-Minty polytopes.

Example 4.4 (Klee-Minty polytopes in $\mathbb{R}^p$). For $x \in \mathbb{R}^p$, consider the Klee-Minty polytopes [5] given by

$$\begin{aligned} [1,0,\ldots,0]x &\leqslant 5,\\ [2^2,1,0,\ldots,0]x &\leqslant 5^2,\\ [2^3,2^2,1,0,\ldots,0]x &\leqslant 5^3,\\ &\vdots\\ [2^p,2^{p-1},\ldots,2^2,1]x &\leqslant 5^p,\\ x &\geqslant 0. \end{aligned}$$

We consider moving two initial points, $x^{(1)} = -250\,\mathbf{1}_p$ and $x^{(2)} = 250\,\mathbf{1}_p$, where $\mathbf{1}_p = [1,1,\ldots,1]^\top \in \mathbb{R}^p$. The results are tabulated in Table 2 for different values of p.

p	3	5	10	15	20	40
			$x_0 = x^{(1)}$			
Algorithm $\mathscr{B}$	139	99	104	109	114	134
Interior point	5	7	9	failed	failed	failed
Dual simplex	4	6	11	16	21	failed
			$x_0 = x^{(2)}$			
Algorithm $\mathscr{B}$	115	85	85	85	85	85
Interior point	5	7	9	failed	failed	failed
Dual simplex	4	6	11	16	21	failed

Table 2. Number of iterations to bring $x^{(1)}$ and $x^{(2)}$ to inside the polytopes for algorithm $\mathscr{B}$ and for the linear program with the interior point and dual simplex methods.

For small p, the interior point and dual simplex algorithms complete the task in much fewer steps than $\mathscr{B}$. However, the interior point algorithm fails for $p \geqslant 15$, as the solution converges to an infeasible point. Truncation error due to the limited number of bits available for the solver leads to the loss of feasibility. The dual simplex method fails for $p > 35$, as the constraint matrix coefficients become too large in magnitude. In contrast, $\mathscr{B}$ completed the task even for $p = 1000$, finishing 1094 iterations in 0.5 seconds when $x_0 = x^{(1)}$. Unlike other methods, $\mathscr{B}$ does not utilize already-satisfied half spaces, as evidenced in the case $x_0 = x^{(2)}$.

As we see in Example 4.4, the simplicity of $\mathscr{B}$ allows us to limit errors arising from truncations and large matrix manipulations. This is quite advantageous for high dimensional problems. With $x_0 = x^{(1)}$, the number of iterations converges to the number of dimensions. This is similar to the simplex method, for which the complexity is $O(p)$ for average cases. However, the $x_0 = x^{(2)}$ case demonstrates that it can be much smaller for problems where many inequalities are idle for the algorithm. In the last example, we use algorithm $\mathscr{B}$ to move an array of points in each iteration to construct a mesh to search for the global optimal.

Example 4.5 (The Ackley function). The Ackley function is used as a performance test function for optimization algorithms [1]. Here we consider the Ackley function of the form

$$\begin{aligned} f(x) = & -20\exp\left[-0.2\sqrt{0.5((x_1-2)^2+(x_2-2)^2)}\right] \\ & -\exp\left[0.5(\cos 2\pi(x_1-2)+\cos 2\pi(x_2-2))\right] + \mathrm{e} + 20 \end{aligned}$$

to find the global optimal of $f(x)$ inside a regular convex hexagon, where $x = [x_1, x_2]^\top$. Function $f(x)$ has many local minima, as shown in the contour plot in

Figure 7. Its global minimum is $f(2,2) = 0$. Depending on the initial point, optimization algorithms can converge to any of the local optimal. If the initial point is infeasible, there is no way to establish a search region with existing approaches.

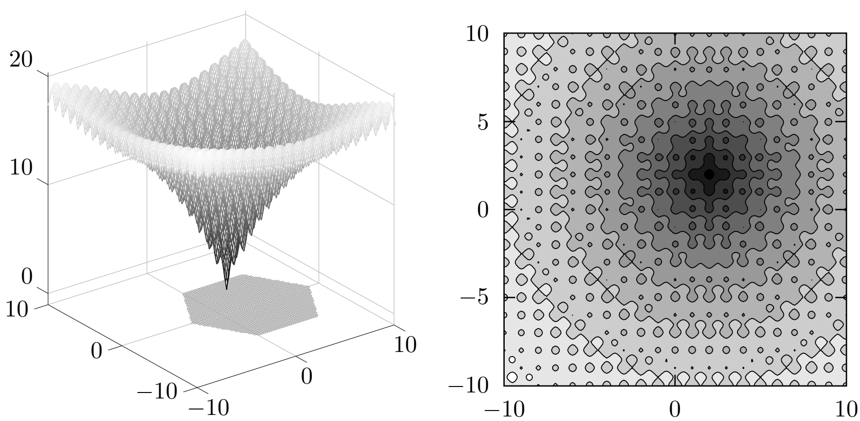


Figure 7. Ackley function and its contour plot.

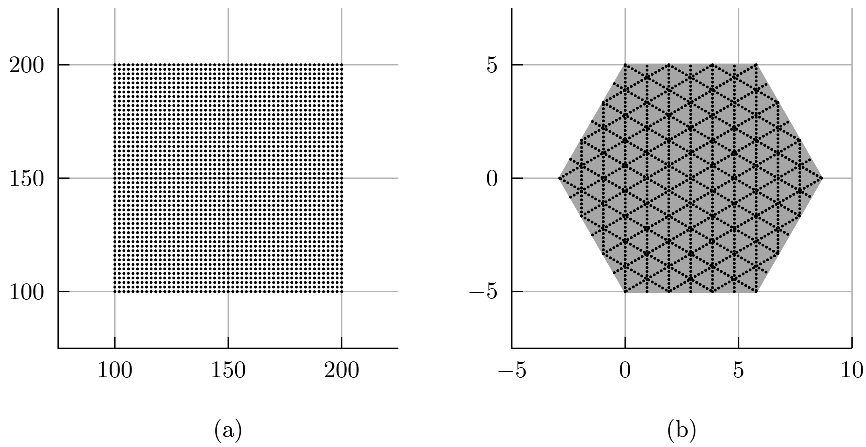


Figure 8. Generation of a mesh grid inside the hexagon using grid points outside the hexagon.

Since reflections are global isometries on $\mathbb{R}^n$, we expect algorithm $\mathscr{B}$ to preserve some of the structure of the set of initial points. Here, we bring a 51×51 grid of points, as shown in Figure 8 (a), to inside the hexagon. It takes less than $1/3$ of a second to complete 30 iterations to bring all 2601 points inside. The corresponding mapping is shown in Figure 8 (b). By evaluating function values, we obtain $(2, 1.92)$ as an approximation to the global optimal. Then we solve the optimization problem using MATLAB's *fmincon* interior-point algorithm with a logarithmic barrier function. The algorithm converges to the global optimal point $(2,2)$ in 12 iterations. In fact, even with an 11×11 grid of points, the algorithms converge to the global optimal solution. For comparison, we use the four edges of the grid to obtain the optimal solutions using the interior-point method and Sequential Quadratic Programming (the SQP method). The results are in Table 3. None of the points converge to the global optimal solution.

x_0	$(100,100)$	$(100,200)$	$(200,200)$	$(200,100)$
Interior point	$(2.989,4.967)$	$(0.035,2.982)$	$(2.000,2.952)$	$(8.982,0.005)$
SQP	$(2.952,2.000)$	$(3.965,2.982)$	$(2.969,1.032)$	$(2.000,2.952)$

Table 3. Optimal solution for four distinct initial points for the interior point and SQP algorithms.

Example 4.5 is a unique application of $\mathscr{B}$ that other existing initialization methods cannot accomplish. By mapping a discretized hypercube to inside a polytope, we can easily construct a mesh in higher dimensions.

Acknowledgments. The authors wish to thank the anonymous reviewers for their helpful comments which have helped to improve the paper.

References

[1] *D. H. Ackley*: A Connectionist Machine for Genetic Hillclimbing. The Springer International Series in Engineering and Computer Science 28. Springer, New York, 2012. doi
[2] *M. S. Bazaraa, J. J. Jarvis, H. D. Sherali*: Linear Programming and Network Flows. John Wiley & Sons, Hoboken, 2005. zbl MR doi
[3] *S. Boyd, L. Vandenberghe*: Convex Optimization. Cambridge University Press, Cambridge, 2004. zbl MR doi
[4] *G. B. Dantzig, M. N. Thapa*: Linear Programming II: Theory and Extensions. Springer Series in Operations Research. Springer, New York, 2003. zbl MR doi
[5] *H. J. Greenberg*: Klee-Minty Polytope Shows Exponential Time Complexity of Simplex Method. University of Colorado at Denver, Denver, 1997.
[6] *W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery*: Numerical Recipes: The Art of Scientific Computing. Cambridge University Press, Cambridge, 2007. zbl MR
[7] *J. Renegar*: A polynomial-time algorithm, based on Newton's method, for linear programming. Math. Program., Ser. A *40* (1988), 59–93. zbl MR doi
[8] *Y. Ye, M. J. Todd, S. Mizuno*: An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. Math. Oper. Res. *19* (1994), 53–67. zbl MR doi

Authors' address: *Dinesh B. Ekanayake* (corresponding author), *Douglas J. LaFountain*, *Boris Petracovici*, Western Illinois University, 1 University Cir, Macomb, IL 61455, Illinois, USA, e-mail: `db-ekanayake@wiu.edu`, `d-lafountain@wiu.edu`, `b-petracovici@wiu.edu`.