Communications in Combinatorics and Optimization
Vol. xx, No. x (xxxx), pp. 1-24
https://doi.org/10.22049/cco.2026.30665.2567

CCO
Commun. Comb. Optim.

Research Article

A simplicial homology interior-point algorithm for nonlinear programming

Ayat Ababneh[†], **Baha Alzalg**[*]

Department of Mathematics, The University of Jordan, Amman, Jordan
[†]a.ababneh@ju.edu.jo
[*]b.alzalg@ju.edu.jo

Received: 28 May 2025; Accepted: 14 April 2026
Published Online: 8 May 2026

Abstract: In this paper, we investigate the use of computational algebraic topology and simplicial homology within the framework of Sperner's lemma to solve box- and inequality-constrained nonlinear (nonconvex) programming problems. The homology clustering method we consider identifies optimal or near-optimal candidate solutions from locally convex subdomains in the search space by computing the homology groups of a simplicial complex constructed on a level set that is topologically equivalent to the structure induced by the objective function. Each candidate point then serves as an initial point for a local iterative interior-point optimization technique that performs well on the corresponding locally convex subdomain. The best solution found by the parallel interior-point searches provides the global optimum for the nonconvex problem. In addition, we compare the proposed method with the existing topographical interior-point approach and other solvers across various test problems. The computational results demonstrate that the simplicial interior-point algorithm performs well in finding the global minimum and outperforms both the topographical interior-point algorithm and the MIDACO solver in practice.

Keywords: Methods of algebraic topology, Simplicial homology, Nonlinear programming, Interior-point methods.

AMS Subject classification: 46M20, 13F55, 90C30, 90C51

1. Introduction

The purpose of this paper is to develop a hybrid simplicial homology and interior-point algorithm to solve the box- and inequality-constrained nonlinear programming (NLP for short) problem in its most general form:

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & g(x) \leq 0, \\ & l \leq x \leq u. \end{array} \tag{1}$$

* *Corresponding author:* B. Alzalg (b.alzalg@ju.edu.jo)

Here, the notation $l \leq x \leq u$ denotes componentwise bound constraints on the decision variable $x \in \mathbb{R}^n$, that is, $l_i \leq x_i \leq u_i$ for all $i = 1, 2, \ldots, n$. In Problem (1) and throughout the paper, $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ are nonlinear functions, $x \in \mathbb{R}^n$ is the decision variable, and the vectors l and u in $\mathbb{R}^n$ are the lower and upper bounds on the decision variable x, respectively. The real-valued functions $f, g_1, g_2, \ldots, g_m$ are assumed to be continuously differentiable but not necessarily convex.

NLPs with nonconvex objectives and constraints are fundamental in numerous scientific and engineering applications, including chemical process optimization, mechanical design, machine learning, and economics (see, for example, [18, 19]). Finding globally optimal solutions to such problems is critical for performance and efficiency, yet it remains a significant computational challenge. The primary difficulty arises from objective function landscapes that may contain multiple local minima, saddle points, and extensive regions where gradient-based methods stagnate. While classical local optimization approaches–such as sequential quadratic programming and interior-point methods–are highly effective for convex problems or when good initial points are available, they lack mechanisms to escape local minima in nonconvex settings. This limitation underscores the need for robust methods that can reliably find global or near-global solutions to constrained nonconvex NLPs.

Existing approaches to global optimization can be broadly categorized into deterministic methods, stochastic/metaheuristic algorithms, and hybrid techniques. Deterministic methods, such as spatial branch-and-bound [4] and interval-based approaches, provide convergence guarantees but often suffer from exponential complexity in high dimensions, limiting their practical applicability. Stochastic and metaheuristic methods–including genetic algorithms [9], particle swarm optimization [15], and differential evolution [24]–perform widespread exploration of the feasible region and are often effective in practice. However, they typically lack deterministic convergence proofs and may require prohibitively many function evaluations for reliable results. Hybrid methods that combine global exploration with local refinement have emerged as a promising direction. Notably, the topographical global optimization method [8] uses concepts from Morse theory to build a graph representation of the function landscape. More recently, tools from computational topology and topological data analysis have been applied to understand optimization landscapes [21]. These approaches can identify persistent topological features (connected components, cycles) that correspond to promising regions in the search space.

Despite these advances, the direct integration of simplicial homology–a concrete, computable tool from algebraic topology–with high-performance interior-point solvers for constrained nonconvex NLPs remains relatively unexplored. While topological ideas show promise for global optimization, there is a need for a practical, computationally efficient framework that: (1) uses simplicial complexes to rigorously analyze constrained feasible domains, (2) systematically generates high-quality starting points for local search, and (3) leverages modern parallel computing architectures. Our work aims to fill this gap by developing a novel hybrid algorithm that combines computational algebraic topology with robust interior-point methods.

The main contributions of this paper are fourfold. First, we propose a simplicial

homology-based global sampling framework that constructs a simplicial complex on the constrained feasible region via Sobol sequence sampling and Delaunay triangulation. By analyzing the oriented graph of this complex, we identify a minimizer pool–a set of candidate points located in topological basins likely to contain local minima. Second, we design a hybrid solver architecture that integrates this topological exploration phase with a high-performance interior-point method. Each point in the minimizer pool serves as a warm start for an independent local search, creating an embarrassingly parallel procedure where multiple local solvers explore distinct promising regions concurrently. Third, we provide comprehensive numerical validation comparing the proposed simplicial interior-point algorithm against state-of-the-art alternatives, including the topographical interior-point method and the MIDACO solver. Our results demonstrate that the simplicial interior-point algorithm consistently finds global or near-global solutions and often outperforms compared methods in solution quality and reliability. Fourth, our method is grounded in Sperner's lemma and simplicial homology, providing a rigorous topological rationale for why vertices in the minimizer pool correspond to potential local minimizers.

1.1. Notations and preliminaries

Throughout this paper, we use I and O to denote the identity and zero matrices of appropriate dimensions, respectively. We also use 0 for the zero vector and $e \triangleq (1, 1, \ldots, 1)^\top$ for a vector of ones with appropriate dimensions. The set $\mathbb{S}^n$ is used to describe the set of n-th order real symmetric matrices. For $A \in \mathbb{S}^n$, we write $A \succeq O$ or $O \preceq A$ (respectively, $A \succ O$ or $O \prec A$) to mean that A is positive semidefinite (respectively, positive definite).

For any vector $x \in \mathbb{R}^n$, we define $X \triangleq \mathrm{Diag}\,(x_1, x_2, \ldots, x_n)$. That is, X denotes the $n \times n$ diagonal matrix whose diagonal entries are $x_1, x_2, \ldots, x_n$. We use "∇_x" to denote the gradient vector (the vector of the first derivatives) of a real-valued function with respect to the vector x. We use "∇^2_{xx}" to denote the Hessian matrix (the matrix of the second derivatives) of a real-valued function with respect to x. We also use "J_x" to denote the Jacobian matrix of a vector-valued function with respect to x. That is, for functions $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$, the gradient vector and Hessian matrix of f, and the Jacobian matrix of g are, respectively, defined as

$$\nabla_x f(x) \triangleq \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}, \nabla^2_{xx} f(x) \triangleq \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}, \mathrm{J}_x g(x) \triangleq \begin{bmatrix} (\nabla_x g_1(x))^\top \\ (\nabla_x g_2(x))^\top \\ \vdots \\ (\nabla_x g_m(x))^\top \end{bmatrix}.$$

Note that the Hessian of a real-valued function is the Jacobian of its gradient. That is,

$$\mathrm{J}_x \nabla_x f(x) = \nabla^2_{xx} f(x).$$

If a real-valued function is twice continuously differentiable, then its Hessian matrix is symmetric. Moreover, if a real-valued function is a twice continuously differentiable convex function, then its Hessian matrix is positive semidefinite (see, for example, [3, Section 8.5]).
The following lemma, which is a restatement of [17, Theorem 2.1] (see also [25, Theorem 2.4]), provides a technical tool for our computations.

Lemma 1. *Consider the partitioned matrix*

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}.$$

(i) If the matrix D is nonsingular, then M is nonsingular if and only if the Schur complement, $A - BD^{-1}C$, of D is nonsingular, and

$$M^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & (A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(D - CA^{-1}B)^{-1}BD^{-1} \end{bmatrix}.$$

(ii) If the matrix A is nonsingular, then M is nonsingular if and only if the Schur complement, $D - CA^{-1}B$, of A is nonsingular, and

$$M^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix}.$$

2. A simplicial homology method for global sampling search

In this section, we are interested in using a simplicial homology method for solving the inequality- and box-constrained NLP problem (1).
We consider the feasible points within the hyperrectangle $[l, u]^n \subset \mathbb{R}^n$. That is, we define the feasibility set

$$\mathcal{S}_1 \triangleq \{x \in \mathbb{R}^n : l \leq x \leq u\}.$$

To approximate $\mathcal{S}_1$ by a simplicial complex, we proceed in two steps: sampling and triangulation. First, we generate a finite set of sample points $S \subseteq \mathcal{S}_1$ using a Sobol sequence. Second, we construct a simplicial complex on S via Delaunay triangulation, which yields a piecewise-linear approximation of $\mathcal{S}_1$ suitable for homological analysis.
A Sobol sequence is a deterministic low-discrepancy sequence in the unit cube $[0, 1]^n$ designed to generate sample points that are uniformly distributed over the domain. Unlike pseudorandom sampling, Sobol sequences minimize discrepancy and avoid clustering effects, ensuring improved coverage of high-dimensional regions. In this work, we scale the Sobol sequence to the hyperrectangle $[l, u]^n$ to generate the point set $S \subseteq \mathcal{S}_1$. This choice ensures that the resulting simplicial complex captures the global geometric structure of the feasible region with relatively few sample points.

Given a finite set of points $S \subset \mathbb{R}^n$, the Delaunay triangulation of S is a simplicial complex in which a simplex is formed whenever the circumsphere of its vertices contains no other points of S in its interior. Equivalently, the Delaunay triangulation is dual to the Voronoi diagram associated with S. The Delaunay triangulation is widely used in computational geometry due to its favorable geometric properties, including the avoidance of degenerate simplices and its ability to adapt to local point density. In our setting, it provides a natural simplicial structure on the Sobol sample points and yields a triangulation of $\mathcal{S}_1$ that serves as the underlying simplicial complex for simplicial homology computations. After constructing the simplicial complex, we assign orientations to its edges. We now recall the necessary notions from algebraic topology that will be used throughout this section; see, for example, [10, 12].

Definition 1. An abstract simplicial complex defined on a set $S \triangleq \{x_1, x_2, \ldots, x_k\} \subseteq \mathbb{R}^n$ is a set $\mathcal{X}$ that satisfies the following conditions:

(i) $\{x_i\} \in \mathcal{X}$ for all $x_i \in S$.

(ii) $\mathcal{X}$ is closed under inclusion, i.e., if $\sigma \in \mathcal{X}$ and $\tau \subseteq \sigma$, then we must have $\tau \in \mathcal{X}$.

Nonempty finite subsets of S that are in $\mathcal{X}$ are called simplices, and the dimension of a simplex σ is $\dim(\sigma) = |\sigma| - 1$, where $|\sigma|$ is the size of σ. We often conflate an abstract simplicial complex $\mathcal{X}$ with its geometric realization $|\mathcal{X}|$.

Definition 2. For a finite abstract simplicial complex $\mathcal{X}$, its standard geometric realization is the topological space formed by taking the union, over all $\sigma \in \mathcal{X}$, of the standard geometric σ-simplices in $\mathbb{R}^{V(\mathcal{X})}$, where $V(\mathcal{X})$ denotes the vertex set of $\mathcal{X}$. Any topological space that is homeomorphic to the standard geometric realization of $\mathcal{X}$ is called the geometric realization of $\mathcal{X}$, and is denoted by $|\mathcal{X}|$.

Definition 3. The standard $n-$simplex, Δ^n, is the convex hull of the set of endpoints of the standard basis vectors of $\mathbb{R}^{n+1}$.

So for example, the geometric realization of the standard $1-$simplex is the line segment in $\mathbb{R}^2$ connecting $(1, 0)$ to $(0, 1)$, and the geometric realization of the standard $2-$simplex is the triangle in $\mathbb{R}^3$ with vertices at the points $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$.

Definition 4. The n-skeleton of a simplicial complex $\mathcal{X}$, denoted by $\mathcal{X}^{(n)}$, is the set of all simplices of dimension at most n.

From the above definition, $\mathcal{X}^{(0)}$ is the set of vertices and $\mathcal{X}^{(1)}$ is the set of all vertices and edges in the simplicial complex $\mathcal{X}$.
We assign an orientation to the set of edges in $\mathcal{X}^{(1)}$ as follows: Consider two vertices $x_i, x_j \in \mathcal{X}^{(0)}$ with corresponding edge $(x_i, x_j) \in \mathcal{X}^{(1)}$. We orient this edge based on their function values: If $f(x_i) < f(x_j)$, direct the edge from x_i to x_j. In the case $f(x_i) = f(x_j)$, we assign a direction to the edge connecting x_i and x_j, orienting it from

the vertex generated earlier in the sampling sequence to the one generated later. Higher-dimensional simplices are given an arbitrary but fixed orientation. Under this orientation scheme, a vertex $x^* \in \mathcal{X}^{(0)}$ is identified as a potential minimizer if all edges incident to x^* are directed away from it. The set of such vertices is called the minimizer pool, denoted by $\mathcal{P}$. In the following example, we find the minimizer pool $\mathcal{P}$.

Example 1. In this example, we consider the modified Becker and Lago problem [11]

$$\begin{array}{ll} \min & f(x,y) = (|x|-5)^2 + (|y|-5)^2 \\ \text{s.t.} & x^2 - 2y^2 \leq 0, \\ & x + y + 2xy - 63 \leq 0, \\ & -10 \leq x, y \leq 10. \end{array}$$

We visually show the graph of this function in Figure 1 with the feasible region on the ground. Using the Sobol sequence with $N = 64$ randomly generated points then evaluating the objective function f at each one of them, we get Table 1.
Plotting the points of S in the rectangular region $[-10, 10] \times [-10, 10]$ then triangulating the region using Delaunay triangulation and giving the edges in the resulting simplicial complex an orientation in the way described previously we get the simplicial complex in Figure 2.

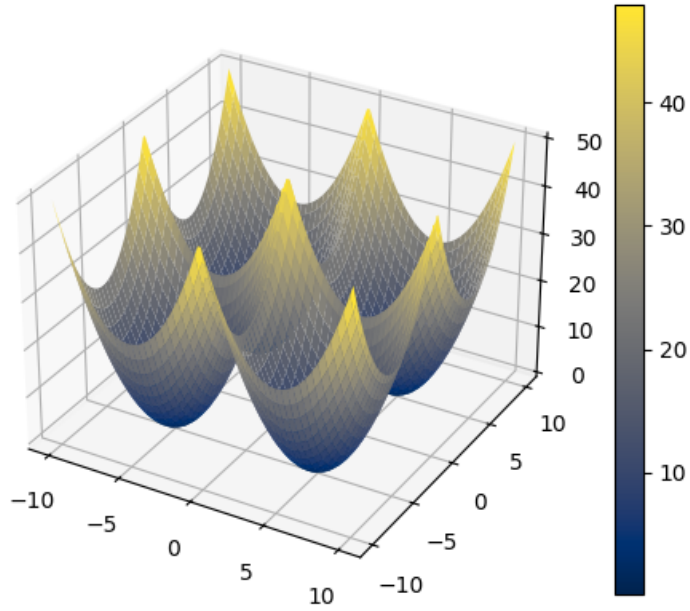

Figure 1. **The graph of the objective function in Example 1.**

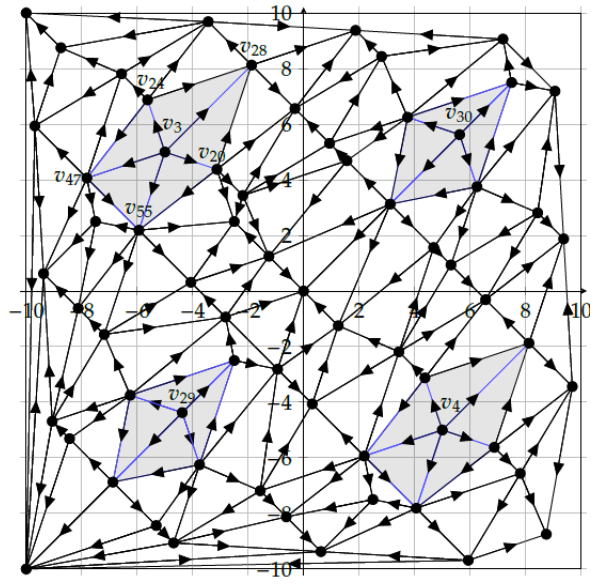

Figure 2. **The simplicial complex in Example 1 and the vertices in the minimizer pool $\mathcal{P}$.**

The vertices v_3, v_4, v_{29} and v_{30} are the minimizers, so the minimizer pool is $\mathcal{P} = \{v_3, v_4, v_{29}, v_{30}\}$.

After identifying the minimizer pool $\mathcal{P}$, the next step is to test the vertices in $\mathcal{P}$ for satisfying some hypothesis; the hypothesis comes from applying Sperner's lemma.

Theorem 1 (Sperner's Lemma [23]). *Every Sperner's labeling of a triangulation of an $n-$dimensional simplex contains a cell labeled with a complete set of labels:* $1, 2, \ldots, n+1$.

Applying Sperner's Lemma helps in determining the set of vertices in $\mathcal{P}$ that are potential for having the real minimum in some neighborhood of them known as the star of a vertex.

Definition 5. A star of a vertex $x_i \in \mathcal{X}$, denoted by $st(x_i)$, is the set of all the simplices (with their interiors) in $\mathcal{X}$ that have x_i as one of their vertices.

In Figure 2, the shaded regions represent the stars of the vertices v_3, v_4, v_{29}, and v_{30}. We then use each vertex that remains after applying Sperner's Lemma as a starting point for an interior-point algorithm applied within its star domain to search for the global minimum.
Going back to Example 1, we divide the plane into three directions $[0, \pi/2), [\pi/2, \pi)$ and $[\pi, 2\pi)$. Note that these intervals are arbitrary; any set of three affinely independent directions covering the plane may be chosen.Now consider the first minimizer v_3, we can see that there is an edge in each of these three directions (see Figure 3): the

Table 1. **The random set S and the function value at each of its points.**

i	v_i	$f_i = f(v_i)$	i	v_i	$f_i = f(v_i)$
1	$(-10, -10)$	50	32	$(-9.0625, -4.6875)$	16.6015625
2	$(0, 0)$	50	33	$(0.9375, 5.3125)$	16.6015625
3	$(5, -5)$	0	34	$(5.9375, -9.6875)$	22.8515625
4	$(-5, 5)$	0	35	$(-4.0625, 0.3125)$	22.8515625
5	$(-2.5, -2.5)$	12.5	36	$(-1.5625, -7.1875)$	16.6015625
6	$(7.5, 7.5)$	12.5	37	$(8.4375, 2.8125)$	16.6015625
7	$(2.5, -7.5)$	12.5	38	$(3.4375, -2.1875)$	10.3515625
8	$(-7.5, 2.5)$	12.5	39	$(-6.5625, 7.8125)$	10.3515625
9	$(-6.25, -3.75)$	3.125	40	$(-5.3125, -8.4375)$	11.9140625
10	$(3.75, 6.25)$	3.125	41	$(4.6875, 1.5625)$	11.9140625
11	$(8.75, -8.75)$	28.125	42	$(9.6875, -3.4375)$	24.4140625
12	$(-1.25, 1.25)$	28.125	43	$(-0.3125, 6.5625)$	24.4140625
13	$(-3.75, -6.25)$	3.125	44	$(-2.8125, -0.9375)$	21.2890625
14	$(6.25, 3.75)$	3.125	45	$(7.1875, 9.0625)$	21.2890625
15	$(1.25, -1.25)$	28.125	46	$(2.1875, -5.9375)$	8.7890625
16	$(-8.75, 8.75)$	28.125	47	$(-7.8125, 4.0625)$	8.7890625
17	$(-8.125, -0.625)$	28.90625	48	$(-8.4375, -5.3125)$	11.9140625
18	$(1.875, 9.375)$	28.90625	49	$(1.5625, 4.6875)$	11.9140625
19	$(6.875, -5.625)$	3.90625	50	$(6.5625, -0.3125)$	24.4140625
20	$(-3.125, 4.375)$	3.90625	51	$(-3.4375, 9.6875)$	24.4140625
21	$(-0.625, -8.125)$	28.90625	52	$(-0.9375, -2.8125)$	21.2890625
22	$(9.375, 1.875)$	28.90625	53	$(9.0625, 7.1875)$	21.2890625
23	$(4.375, -3.125)$	3.90625	54	$(4.0625, -7.8125)$	8.7890625
24	$(-5.625, 6.875)$	3.90625	55	$(-5.9375, 2.1875)$	8.7890625
25	$(-6.875, -6.875)$	7.03125	56	$(-7.1875, -1.5625)$	16.6015625
26	$(3.125, 3.125)$	7.03125	57	$(2.8125, 8.4375)$	16.6015625
27	$(8.125, -1.875)$	19.53125	58	$(7.8125, -6.5625)$	10.3515625
28	$(-1.875, 8.125)$	19.53125	59	$(-2.1875, 3.4375)$	10.3515625
29	$(-4.375, -4.375)$	0.78125	60	$(-4.6875, -9.0625)$	16.6015625
30	$(5.625, 5.625)$	0.78125	61	$(5.3125, 0.9375)$	16.6015625
31	$(0.625, -9.375)$	38.28125	62	$(0.3125, -4.0625)$	22.8515625
32	$(-9.375, 0.625)$	38.28125	63	$(-9.6875, 5.9375)$	22.8515625

edge v_3v_{28} has direction in $[0, \pi/2)$, the edge v_3v_{24} has direction in $[\pi/2, \pi)$, and the three edges v_3v_{47}, v_3v_{55}, and v_3v_{20} have directions in $[\pi, 2\pi)$. Similarly for the other minimizers. So from Sperner's lemma and by applying the Brouwer fixed point theorem, we guarantee that there is some Sperner simplex within the star domain of each minimizer, and thus a stationary point ($\nabla f = 0$) exists within each star domain. From Sperner's lemma and the Brouwer fixed point theorem, we guarantee the existence of a Sperner simplex within the star domain of each minimizer. Consequently, there exists a point in each star domain where $\nabla f = 0$; that is, a stationary point lies within each minimizer's star. We therefore use each minimizer as a starting point for an interior-point search.

Algorithm accuracy is an important consideration. One way to improve accuracy is to reduce the size of the region where a minimum can occur—specifically, to shrink the star domains. This can be achieved by increasing the sample size $|\mathcal{S}|$. A natural question arises: if we increase $|\mathcal{S}|$ to obtain smaller stars, what happens to the size of the minimizer pool $|\mathcal{P}|$? A crucial theorem in [7] shows that increasing $|\mathcal{S}|$ does not increase $|\mathcal{P}|$.

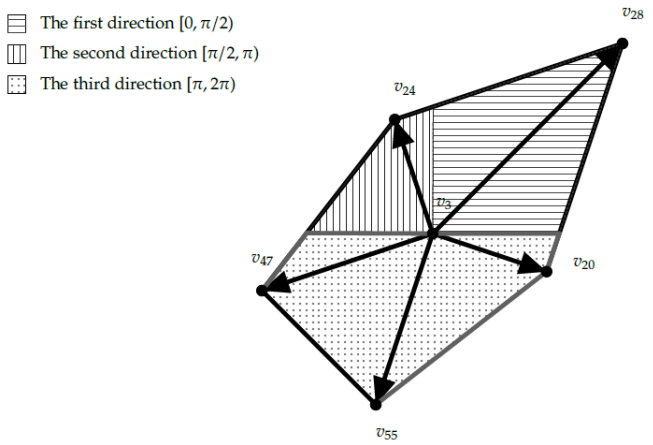

Figure 3. Detailed structure of star(v_3) showing the classification of edges into three angular sectors. The vertex v_3 has edges pointing away in all three sectors: v_3v_{28} in sector 1, v_3v_{24} in sector 2, and v_3v_{47}, v_3v_{55}, v_3v_{20} in sector 3.

3. A feasible direction interior-point method for local search

In this section, we are interested in using a feasible direction interior-point method (see for example [1, 13, 16]) for solving the inequality-constrained NLP problem:

$$\begin{aligned} &\min \quad f(x) \\ &\text{s.t.} \quad g_i(x) \leq 0, \ i = 1, 2, \ldots, m. \end{aligned} \tag{2}$$

Define the feasibility sets

$$\begin{aligned} \mathcal{S}_2 &\triangleq \{x \in \mathbb{R}^n : g_i(x) \leq 0,\ i = 1, 2, \ldots, m\}, \\ \mathcal{S}_2^\circ &\triangleq \{x \in \mathbb{R}^n : g_i(x) < 0,\ i = 1, 2, \ldots, m\}. \end{aligned}$$

Now, we make the following assumption.

Assumption 1. *The set $\mathcal{S}_2^\circ$ is nonempty.*

Based on Assumption 1, Problem (2) is strictly feasible. Given a starting point $(x, \lambda) \in \mathcal{S}_2^\circ \times \mathbb{R}^m$, an interior-point method generates a sequence of interior points in $\mathcal{S}_2$ with decreasing values of the objective function $f(x)$.
If $x \in \mathcal{S}_2$ is a local minimizer of f in $\mathcal{S}_2$, then there are Lagrange multipliers $\lambda_1, \lambda_2, \ldots, \lambda_m$ such that the Karush–Kuhn–Tucker (KKT) conditions are satisfied at x (see [18, Section 10.6]):

$$\nabla_x f(x) + \sum_{i=1}^{m} \lambda_i \nabla_x g_i(x) = 0 \quad \text{(stationarity condition)}, \tag{3}$$

$$\lambda_i g_i(x) = 0, i = 1, 2, \ldots, m, \quad \text{(complementary slackness)}, \tag{4}$$

$$g_i(x) \leq 0, i = 1, 2, \ldots, m, \quad \text{(primal feasibility)}, \tag{5}$$

$$\lambda_i \geq 0, i = 1, 2, \ldots, m, \quad \text{(dual feasibility)}. \tag{6}$$

So, by introducing a vector of Lagrange multipliers $\lambda = (\lambda_1, \ldots, \lambda_m) \in \mathbb{R}^m$, the Lagrangian function in vector form is

$$\mathscr{L}(x, \lambda) \triangleq f(x) + \lambda^\mathsf{T} g(x),$$

and its gradient and Hessian (with respect to x) are

$$\begin{aligned} \nabla_x \mathscr{L}(x, \lambda) &= \nabla_x f(x) + \mathrm{J}_x^\mathsf{T} g(x)\, \lambda, \\ \nabla_{xx}^2 \mathscr{L}(x, \lambda) &= \nabla_{xx}^2 f(x) + \textstyle\sum_{i=1}^m \lambda_i \nabla_{xx}^2 g_i(x). \end{aligned}$$

To compute new iterates of the optimization vector x and the vector of Lagrange multipliers λ, we apply a Newton or Newton-like method to the nonlinear system described by Equations (3) and (4). This system of nonlinear equations in (x, λ) can be written in compact form as

$$\begin{aligned} \nabla_x f(x) + \mathrm{J}_x^\mathsf{T} g(x)\, \lambda &= 0, \\ G(x)\, \lambda &= 0, \end{aligned} \tag{7}$$

where, as indicated in Subsection 1.1, $G(x)$ is a diagonal matrix whose diagonal entries are $g_1(x), g_2(x), \ldots, g_m(x)$, and $\mathrm{J}_x g(x)$ is the Jacobian matrix of $g(x)$.
Let $\mathscr{H}(x,\lambda)$ be the Hessian matrix $\nabla^2_{xx}\mathscr{L}(x,\lambda)$ or a quasi-Newton estimate of $\nabla^2_{xx}\mathscr{L}(x,\lambda)$. The following assumption is needed to guarantee a global convergence of Problem (2).

Assumption 2. *For any $x \in \mathcal{S}_2^\circ$ and $\lambda \geq 0$, we have $\mathscr{H}(x,\lambda) \succ O$.*

Applying a quasi-Newton method to System (7), we get

$$\begin{aligned} \mathscr{H}(x,\lambda)\Delta_\alpha x + \nabla^2_{x\lambda}\mathscr{L}(x,\lambda)\Delta_\alpha\lambda &= -\nabla_x\mathscr{L}(x,\lambda), \\ \nabla_x(\Lambda g(x))\Delta_\alpha x + \nabla_\lambda(G(x)\lambda)\Delta_\alpha\lambda &= -G(x)\lambda, \end{aligned}$$

where $\Delta_\alpha x$ and $\Delta_\alpha\lambda$ are the search directions in the spaces of the optimization vector x and the vector of Lagrange multipliers λ, respectively. This leads us to the system

$$\begin{aligned} \mathscr{H}(x,\lambda)\Delta_\alpha x + \mathrm{J}_x^{\mathsf{T}} g(x)\Delta_\alpha\lambda &= -\nabla_x f(x) - \mathrm{J}_x^{\mathsf{T}} g(x)\,\lambda, \\ \Lambda\,\mathrm{J}_x g(x)\Delta_\alpha x + G(x)\Delta_\alpha\lambda &= -G(x)\lambda, \end{aligned} \tag{8}$$

where Λ is a diagonal matrix whose diagonal entries are $\lambda_1, \lambda_2, \ldots, \lambda_m$,
In matrix form, System (8) is written as

$$\begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\alpha x \\ \Delta_\alpha\lambda \end{bmatrix} = - \begin{bmatrix} \nabla_x f(x) + \mathrm{J}_x^{\mathsf{T}} g(x)\,\lambda \\ G(x)\lambda \end{bmatrix}. \tag{9}$$

Let $x(\alpha) \triangleq \lambda + \Delta_\alpha x$ and $\lambda(\alpha) \triangleq \lambda + \Delta_\alpha\lambda$. System (9) can be equivalently written as

$$\begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\alpha x \\ \lambda(\alpha) \end{bmatrix} = \begin{bmatrix} -\nabla_x f(x) \\ 0 \end{bmatrix}. \tag{10}$$

Solving System (10) provides the pair $(\Delta_\alpha x, \lambda(\alpha))$. Based on the following lemma, which is due to a lemma in [20], the pair $(\Delta_\alpha x, \lambda(\alpha))$ can be uniquely determined.

Lemma 2. *Given any $x \in \mathcal{S}_2^\circ$, any $\mathscr{H}(x,\lambda) \succ O$ and any $\lambda \geq 0$, the partitioned matrix*

$$\begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix}$$

is nonsingular.

Note that, from Assumption 2, the matrix $\mathscr{H}(x,\lambda)$ is nonsingular. Note also that the matrix $G(x)$ is nonsingular since $x \in \mathcal{S}_2^\circ$. By mimicking Gaussian elimination and solving System (10) for $\lambda(\alpha)$, we get

$$\lambda(\alpha) \;=\; -G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)\,\Delta_\alpha x. \tag{11}$$

This provides a new iterate of the vector of Lagrange multipliers λ. We find an explicit expression of the search direction $\Delta_\alpha x$ by solving the system

$$\Big(\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)\Big)\,\Delta_\alpha x \;=\; -\nabla_x f(x). \tag{12}$$

From item (i) in Lemma 1 and Lemma 2, the Schur complement, $\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)$, of $G(x)$ is nonsingular, hence

$$\Delta_\alpha x \;=\; -\Big(\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)\Big)^{-1}\,\nabla_x f(x). \tag{13}$$

The direction $\Delta_\alpha x$ is a descent direction since, for a sufficiently small step size, moving along the direction $\Delta_{\alpha x}$ guarantees a decrease in the objective function. Specifically, it satisfies the first-order descent condition $\nabla f(x)^\top \Delta_{\alpha x} < 0$ whenever x is not a stationary point. However, the direction $\Delta_\alpha x$ may not be a feasible direction for Problem (2). To tackle this issue, Herskovits [13] proposes an approach to deflect the descent direction and obtain a feasible descent direction by solving linear system in the new estimate vectors Δx and $\bar{\lambda}$.

$$\begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta x \\ \bar{\lambda} \end{bmatrix} \;=\; \begin{bmatrix} -\nabla_x f(x) \\ -\rho\lambda \end{bmatrix}. \tag{14}$$

Here, ρ is a positive scalar factor.
Introducing the vectors $\Delta_\beta x$ and $\lambda(\beta)$ such that

$$\Delta x \;\triangleq\; \Delta_\alpha x + \rho\Delta_\beta x \quad \text{and} \quad \bar{\lambda} \;\triangleq\; \lambda(\alpha) + \rho\lambda(\beta). \tag{15}$$

Using (10), (15) and (14), we have

$$\begin{aligned}
\begin{bmatrix} -\nabla_x f(x) \\ 0 \end{bmatrix} &= \begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\alpha x \\ \lambda(\alpha) \end{bmatrix} \\
&= \begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta x - \rho\Delta_\beta x \\ \bar{\lambda} - \rho\lambda(\beta) \end{bmatrix} \\
&= \begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta x \\ \bar{\lambda} \end{bmatrix} - \rho \begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\beta x \\ \lambda(\beta) \end{bmatrix} \\
&= \begin{bmatrix} -\nabla_x f(x) \\ -\rho\lambda \end{bmatrix} - \rho \begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda\,\mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\beta x \\ \lambda(\beta) \end{bmatrix}.
\end{aligned}$$

It follows that

$$\begin{bmatrix} \mathscr{H}(x,\lambda) & \mathrm{J}_x^{\mathsf{T}} g(x) \\ \Lambda \mathrm{J}_x g(x) & G(x) \end{bmatrix} \begin{bmatrix} \Delta_\beta x \\ \lambda(\beta) \end{bmatrix} = \begin{bmatrix} 0 \\ -\lambda \end{bmatrix}. \tag{16}$$

Solving System (16) provides the pair $(\Delta_\beta x, \lambda(\beta))$. Note that Systems (10) and (16) have the same coefficient matrix. So, based on Lemma 2, the pair $(\Delta_\beta x, \lambda(\beta))$ can also be uniquely determined.
By mimicking Gaussian elimination and solving System (16) for $\lambda(\beta)$, we get

$$\lambda(\beta) = -G^{-1}(x)\,\Lambda\left(e + \mathrm{J}_x g(x)\,\Delta_\beta x\right), \tag{17}$$

where $e \triangleq (1, 1, \ldots, 1)^{\mathsf{T}} \in \mathbb{R}^m$. We find an explicit expression of the search direction $\Delta_\beta x$ by solving the system

$$\left(\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)\right)\Delta_\beta x = \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\lambda. \tag{18}$$

From item (i) in Lemma 1 and Lemma 2, the Schur complement, $\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)$, of $G(x)$ is nonsingular, hence

$$\Delta_\beta x = \left(\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)\right)^{-1} \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\lambda. \tag{19}$$

The expressions of $\Delta_\alpha x$ and $\Delta_\beta x$ in (13) and (19) are given using the Schur complement, $\mathscr{H}(x,\lambda) - \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\Lambda\,\mathrm{J}_x g(x)$, of $G(x)$. These expressions can instead be given using the Schur complement, $G(x) - \Lambda\,\mathrm{J}_x g(x) \mathscr{H}^{-1}(x,\lambda) \mathrm{J}_x^{\mathsf{T}} g(x)$, of $\mathscr{H}(x,\lambda)$ by applying the result of item (ii) in Lemma 1 to the coefficient matrix of Systems (10) and (16) and obtain

$$\begin{aligned} \Delta_\alpha x = -\Big[& \mathscr{H}^{-1}(x,\lambda) + \mathscr{H}^{-1}(x,\lambda)\,\mathrm{J}_x^{\mathsf{T}} g(x) \\ & \left(G(x) - \Lambda\,\mathrm{J}_x g(x) \mathscr{H}^{-1}(x,\lambda) \mathrm{J}_x^{\mathsf{T}} g(x)\right)^{-1} \Lambda\,\mathrm{J}_x g(x) \mathscr{H}^{-1}(x,\lambda)\Big] \nabla_x f(x), \end{aligned} \tag{20}$$

and

$$\begin{aligned} \Delta_\beta x = & \Big[\mathscr{H}^{-1}(x,\lambda) + \mathscr{H}^{-1}(x,\lambda)\,\mathrm{J}_x^{\mathsf{T}} g(x) \\ & \left(G(x) - \Lambda\,\mathrm{J}_x g(x) \mathscr{H}^{-1}(x,\lambda) \mathrm{J}_x^{\mathsf{T}} g(x)\right)^{-1} \Lambda\,\mathrm{J}_x g(x) \mathscr{H}^{-1}(x,\lambda)\Big] \mathrm{J}_x^{\mathsf{T}} g(x) G^{-1}(x)\,\lambda. \end{aligned} \tag{21}$$

Finally, the feasible descent search direction Δx is computed using the first equation in (15), where $\Delta_\alpha x$ and $\Delta_\beta x$ are calculated in (13) and (19) (equivalently, (20) and (21)), respectively.

4. A simplicial homology feasible direction interior-point algorithm

A simplicial homology interior-point algorithm (SHIPA, for short) for solving Problem (2) is formally presented in Algorithms 1–4.

4.1. SHIPA description

The minimizer pool is constructed in Algorithm 1. Algorithm 2 employs an inexact line search to determine the step length. Algorithm 3 is an interior-point scheme that uses the step length computed by Algorithm 2 to locate a local minimum starting from a pool point. Algorithm 4 serves as the main routine, calling both Algorithms 1 and 3. See Figure 4, which illustrates the basic scheme of SHIPA.

We have expanded the description of the minimizer pool construction in Section 2. Specifically, we now clarify that the minimizer pool $\mathcal{P}$ is formed by first generating a set of sample points S using a Sobol sequence, then constructing a Delaunay triangulation of these points, and orienting the edges according to the objective function values. A vertex is included in the minimizer pool if all edges connected to it point away from the vertex, indicating a local minimum in the discrete complex. To ensure full transparency and reproducibility, we have added a detailed step-by-step explanation, illustrated by Example 1 and Figure 2 in Section 2.

We initialize Algorithm 3 with $\mathscr{H}(x, \lambda) = I$ and $\lambda = e$. The parameter values used in Algorithms 2 and 3 are listed in Table 2.

As discussed in Section 3, the search direction Δx is computed in two steps: first, the vector $\Delta_\alpha x$ is calculated (line 3 of Algorithm 3); second, the vector $\Delta_\beta x$ is obtained (line 5). Algorithm 3 uses (13) and (19) to compute $\Delta_\alpha x$ and $\Delta_\beta x$, respectively. Alternative expressions for these vectors are provided in (20) and (21) (see Section 3).

Algorithm 2 employs an inexact line search to select the step length, ensuring a sufficient decrease condition known as the Armijo condition [19]. Accordingly, in line 16 of Algorithm 3, the new point is updated as $x \leftarrow x + t\Delta x$, where t is the step length.

In line 25 of Algorithm 3, the symmetric positive definite matrix $\mathscr{H}$ is updated using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm, a member of the quasi-Newton family [5]. To maintain positive definiteness, the BFGS update requires the curvature condition $\delta^\top \gamma > 0$, where $\delta \triangleq x_+ - x_c$ and $\gamma \triangleq \nabla_x \mathscr{L}(x_+, \lambda) - \nabla_x \mathscr{L}(x_c, \lambda)$, with x_c and x_+ denoting the current and new points, respectively. However, this condition may not hold when the objective function is the Lagrangian, as in our case. To address this issue, we follow Herskovits [13] and use the BFGS formula with the Powell modification [22], as implemented in line 23 of Algorithm 3.

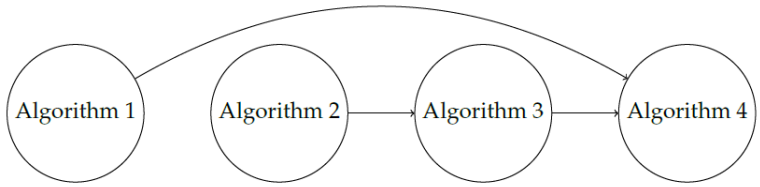

Figure 4. A basic scheme of SHIPA: Algorithm 2 is called by Algorithm 3, while Algorithms 1 and 3 are both called by Algorithm 4.

Algorithm 1: Construction of the minimizer pool via simplicial homology

Input: Objective function f, hyperrectangle $[l,u]^n$, constraint function g, number of sampling points N
Output: Minimizer pool $\mathcal{P}$

$\mathcal{S} \leftarrow \emptyset$;
$\mathcal{S}_1 \leftarrow \{x \in \mathbb{R}^n : l \le x \le u\}$;
$\mathcal{S}_2 \leftarrow \{x \in \mathbb{R}^n : g(x) \le 0\}$;
while $|\mathcal{S}| < N$ **do**
    Generate $N - |\mathcal{S}|$ Sobol sampling points $\mathcal{X}$ in $[0,1]^n$;
    Scale $\mathcal{X}$ to fit inside $\mathcal{S}_1$;
    $\mathcal{X} \leftarrow$ intersect$(\mathcal{X}, \mathcal{S}_2^\circ)$;
    $\mathcal{S} \leftarrow$ union$(\mathcal{S}, \mathcal{X})$;
end while
Build simplicial complex $\mathcal{K}$ by triangulating $\mathcal{S}_1 \cap \mathcal{S}_2^\circ$;
$\mathcal{F} \leftarrow \{f(x) : x \in \mathcal{S}\}$;
Form directed simplicial complex $\mathcal{H}$ using $\mathcal{F}$ and $\mathcal{K}$;
Construct minimizer pool $\mathcal{P} \leftarrow \{p_1, p_2, \ldots, p_s\}$ using $\mathcal{H}$;
return $\mathcal{P}$;

Algorithm 2: Feasible line search

Input: x, search direction Δx, multipliers $\bar{\lambda}$
Output: Step length t
Parameters: $\eta \in (0,1), \nu \in (0,1)$

Find first $t \in \{1, \nu, \nu^2, \ldots\}$ such that

$$f(x + t\Delta x) \le f(x) + t\eta \Delta x^{\mathsf{T}} \nabla_x f(x)$$

$$\text{and} \begin{cases} g_i(x + t\Delta x) < 0, & \text{if } \bar{\lambda}_i \ge 0, \\ g_i(x + t\Delta x) \le g_i(x), & \text{if } \bar{\lambda}_i < 0; \end{cases}$$

return t

Table 2. **The numerical values of the parameters used in Algorithms 2 and 3.**

Parameter	ε	ϵ	φ	ξ	η	ν
Domain	$(0,\infty)$	$(0,\infty)$	$(0,\infty)$	$(0,1)$	$(0,1)$	$(0,1)$
Value	10^{-8}	0.2	0.8	0.8	0.7	0.833

4.2. Convergence discussion

Based on Assumption 1, $\mathcal{S}_2^\circ \neq \emptyset$, i.e., the strict interior of the feasible set is nonempty. Starting from any $(x, \lambda) \in \mathcal{S}_2^\circ \times \mathbb{R}^m$, the interior-point refinement stage of SHIPA generates a sequence of points in $\mathcal{S}_2$ with nonincreasing objective function values. Assumption 2 ensures convergence to a local minimizer by requiring that the Hessian of the Lagrangian, $\mathscr{H}(x,\lambda) = \nabla^2_{xx}\mathscr{L}(x,\lambda)$ (or its quasi-Newton approximation), is symmetric positive definite for all $x \in \mathcal{S}_2^\circ$ and $\lambda \ge 0$.
In addition, the search directions are chosen to satisfy a descent condition, and the

Algorithm 3: Interior-point local minimization from a pool point

Input: Starting point p_j, parameters, and problem data
Output: Local minimum p_j
Data: $0 < \lambda \in \mathbb{R}^m, O < \mathscr{H} \in \mathbb{R}^{n\times n}$
Parameters: $\varepsilon > 0, \epsilon > 0, \varphi > 0, \xi \in (0,1)$

$x \leftarrow p_j$;
$\nabla_x \mathscr{L}_c(x,\lambda) \leftarrow \nabla_x f(x) + \mathrm{J}_x^{\mathsf{T}} g(x)\,\lambda$;
Compute search direction $\Delta_\alpha x$ using (13);
while $\|\Delta_\alpha x\| > \varepsilon$ **do**
    Compute $\Delta_\beta x$ using (19);
    if $\Delta_\beta x^{\mathsf{T}} \nabla_x f(x) > 0$ **then**
$\rho \leftarrow \min\left\{\varphi\|\Delta_\alpha x\|^2 : (\xi-1)\dfrac{\Delta_\alpha x^{\mathsf{T}} \nabla_x f(x)}{\Delta_\beta x^{\mathsf{T}} \nabla_x f(x)}\right\}$;
    else
        $\rho \leftarrow \varphi\|\Delta_\alpha x\|^2$;
    end if
end if
$\Delta x \leftarrow \Delta_\alpha x + \rho \Delta_\beta x$;
Compute multiplier directions $\lambda(\alpha)$ and $\lambda(\beta)$ using (11) and (17);
$\bar{\lambda} \leftarrow \lambda(\alpha) + \rho\lambda(\beta)$;
Call Algorithm 2 to compute step length t;
$x \leftarrow x + t\Delta x$;
for $(i = 1; i < m; i++)$ **do**
    $\lambda_i \leftarrow \max\left\{\lambda_{\alpha_i}, \epsilon\|\Delta_\alpha x\|^2\right\}$;
end for
$\nabla_x \mathscr{L}_+(x,\lambda) \leftarrow \nabla_x f(x) + \mathrm{J}_x^{\mathsf{T}} g(x)\,\lambda$;
$\gamma \leftarrow \nabla_x \mathscr{L}_+(x,\lambda) - \nabla_x \mathscr{L}_c(x,\lambda)$;
$\delta \leftarrow t\Delta x$;
$\varphi \leftarrow \begin{cases} 1, & \text{if } \delta^{\mathsf{T}}\gamma \geq 0.2\,\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)\delta, \\ \dfrac{0.8\,\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)\delta}{\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)\delta - \delta^{\mathsf{T}}\gamma}, & \text{if } \delta^{\mathsf{T}}\gamma < 0.2\,\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)\delta; \end{cases}$
$\sigma \leftarrow \varphi\gamma + (1-\varphi)\mathscr{H}(x,\lambda)\delta$;
$\mathscr{H}(x,\lambda) \leftarrow \mathscr{H}(x,\lambda) - \dfrac{\mathscr{H}(x,\lambda)\delta\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)}{\delta^{\mathsf{T}}\mathscr{H}(x,\lambda)\delta} + \dfrac{\sigma\sigma^{\mathsf{T}}}{\delta^{\mathsf{T}}\sigma}$;
$\nabla_x \mathscr{L}_c(x,\lambda) \leftarrow \nabla_x \mathscr{L}_+(x,\lambda)$;
Update gradient and recompute $\Delta_\alpha x$ using (13);
end while
$p_j \leftarrow x$;
return p_j

Algorithm 4: Simplicial homology interior-point framework

Input: Problem data
Output: Global minimum $x^\star$ of Problem (1)

Call Algorithm 1 to construct a minimizer pool $\mathcal{P}$;
forall $j \in [s]$ in parallel **do**
    Call Algorithm 3 to find a local minimum p_j;
end forall
$x^\star \leftarrow \arg\min\{f(p_i)\}_{i=1}^s$;
return $x^\star$

step sizes are determined using an appropriate line search (e.g., satisfying the Wolfe conditions). Under these assumptions, each iteration of the interior-point method reduces a merit function associated with the Lagrangian, ensuring convergence to a stationary point of Problem (2).
Although global convergence to the overall global minimizer is not guaranteed, the first-stage minimizer pool increases the likelihood of locating high-quality solutions by providing diverse starting points for local refinement.

5. Numerical results

In this section, we present numerical results to demonstrate that SHIPA stated in Section 4 performs well in finding the global minimum. We also compare the proposed method with an existing algorithm and other solvers across various test problems. All numerical results presented in this study were generated using a computer operating on the Windows 10 system. Additionally, all computational codes utilized for obtaining these results were written and executed in the Python programming language after being properly compiled.
First, we apply SHIPA to 6 test problems, with their formulations provided in Table 3. These problems involve inequality constraints in the form of a hyperrectangle in $\mathbb{R}^n$. The numerical results are presented in Table 5. Next, we implement SHIPA on 39 test problems from Ali et al. [2], which involve more general inequality constraints. These numerical results are displayed in Table 6, which also compares the SHIPA of this paper with the topographical global interior-point algorithm (TGIPA) of [11] and the MIDACO solver.
We note that although the theoretical convergence analysis of the proposed algorithm assumes continuously differentiable functions, several benchmark problems considered in the numerical experiments involve piecewise smooth objective functions containing absolute value terms (namely, the Modified Becker and Lago problem and the Cross-in-tray problem). These functions are differentiable almost everywhere, and such test problems are widely used in the global optimization literature to evaluate the robustness of gradient-based algorithms in the presence of mild nonsmoothness.
In Tables 5 and 6, n is the number of variables in the optimization problem, m is the number of inequality constraints and N is the number of sample points generated in the hyperrectangle $\mathcal{S}_1$ at each test. In Table 5, for each test problem, we show the vertices of the minimizer pool, and the local minimizer points and the global minimizer points of the objective function f in $\mathcal{S}_1 \cap \mathcal{S}_2$, and the global minimum value of f denoted as f_{opt}. In Table 6, the performance of the methods is measured using CPU time.
Analyzing the results in Table 5, we can observe that SHIPA was successful in finding the global minimum for each of the optimization test problems in Table 5. Analyzing the results in Table 6, we can observe that in 28 problems out of the 38 test problems, the SHIPA method was more efficient than the other two methods, and in only 2 problems MIDACO was more efficient than SHIPA and in the rest of the problems (8

Table 3. **Inequality- and box-constrained NLP test problems under study.**

Problem name	Problem formulation	
Modified Becker and Lago problem [11]	min	$(\|x\|-5)^2+(\|y\|-5)^2$
	s.t.	$x^2-2y^2\leq 0,$
		$x+y+2xy-63\leq 0,$
		$-10\leq x,y\leq 10.$
Cross-in-tray problem [11]	min	$-0.0001\left(\left\|\sin(x_1)\sin(x_2)\,e^{\left\|100-\frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\|}\right\|+1\right)^{0.1}$
	s.t.	$x_1(1-x_2)-(x_2+3)^2-x_1^2\leq 0,$
		$-10\leq x_1,x_2\leq 10.$
Hock and Schittkowski 29 problem [14]	min	$-x_1x_2x_3$
	s.t.	$x_1^2+x_2^2+4x_3^2-48\leq 0,$
		$-5\leq x_1\leq 5,$
		$-4\leq x_2\leq 4,$
		$-3\leq x_3\leq 3.$
Dekkers and Aarts problem [6]	min	$10^5x_1^2+x_2^2-(x_1^2+x_2^2)^2+10^{-5}(x_1^2+x_2^2)^4$
	s.t.	$-20\leq x_1,x_2\leq 20.$
Modified Branin problem [11]	min	$(x_2-0.1292x_1^2+1.5915x_1-6)^2$
		$+9.602\cos x_1+10$
	s.t.	$x_1x_2-23.5\leq 0,$
		$x_1+x_2-15\leq 0,$
		$-4\leq x_1\leq 10,$
		$1\leq x_2\leq 13.$
Modified six-hump camel back problem [2]	min	$(4-2.1x_1^2+x_1^4/3)x_1^2+x_1x_2+(-4+4x_2^2)x_2^2$
	s.t.	$x_1x_2^3\leq 0,$
		$x_1^3-x_2^2\leq 0,$
		$x_1+x_2^2+2x_2-3\leq 0,$
		$-3\leq x_1\leq 3,$
		$-2\leq x_2\leq 2.$

problems) the TGIPA was more efficient. So, we can say in almost 74% of the problems that SHIPA was more or as effective as the other two methods. A visualization of some of the results in Table 6 can be seen in Figure 5.

The construction of the minimizer pool, which involves generating Sobol sample points, building a Delaunay triangulation, and computing homology groups, represents the most computationally intensive step in SHIPA. To quantify its cost relative to the interior-point refinement, Table 7 presents representative timings for selected test problems. As observed, the minimizer pool accounts for approximately 43–46% of the total computational time, with the remaining time spent on interior-point iterations starting from the pool points. Importantly, the minimizer pool is computed only once at the beginning and can be efficiently parallelized, mitigating its computational

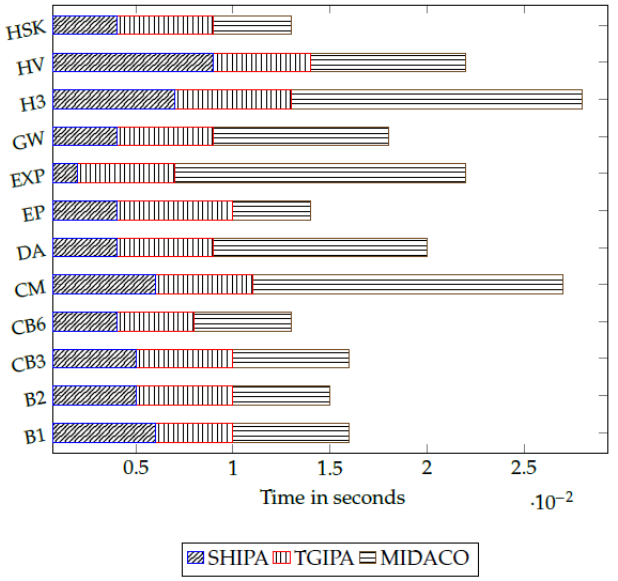

Figure 5. **A visualization of the results on 12 test problems from Table6**

impact for larger problems. This demonstrates that, while the homology-based construction is relatively expensive, it provides high-quality initial points that improve the overall efficiency and reliability of the interior-point refinement.

The numerical experiments presented in this paper focus on low-dimensional test problems. The primary purpose of these experiments is to provide a controlled environment that clearly demonstrates the behavior of the proposed SHIPA framework and, in particular, the effectiveness of the minimizer pool construction, which constitutes the main contribution of this work.

From a methodological standpoint, the proposed algorithm is not restricted to small-scale settings. The construction of the minimizer pool and the interior-point framework are formulated in a dimension-independent manner and therefore remain applicable to general problem sizes. In particular, the computational complexity of the SHIPA components grows polynomially with respect to the problem dimension, similarly to classical interior-point methods. It should be noted that computational time comparisons in low-dimensional settings may be influenced by implementation details and programming environments. Nevertheless, the reported results provide a consistent indication of the computational behavior of SHIPA and demonstrate its competitiveness when compared with established optimization methods.

Overall, the computational experiments indicate that the simplicial interior-point algorithm demonstrates reliable practical performance and achieves competitive efficiency and solution quality when compared with the topographical interior-point algorithm and the MIDACO solver.

Table 5. The numerical results obtained for 6 test problems showing performance of SHIPA, TGIPA, and MIDACO

Test problem	m	n	N	Points in the minimizer pool	Local minimizers	Global minimizer?	f_{opt}
Modified Becker and Lago problem	2	2	64	$(5,-5),(-5,5),$ $(-5,-5),(5,5)$	$(5,-5),(-5,5),$ $(-5,-5),(5,5)$	✓ ✓	0
Cross-in-tray problem	1	2	465	$(1.34941,1.34941),$ $(1.34941,-1.34941),$ $(-1.34941,1.34941),$ $(-1.34941,-1.34941)$	$(1.34941,1.34941),$ $(1.34941,-1.34941),$ $(-1.34941,1.34941),$ $(-1.34941,-1.34941)$	✓ ✓ ✓ ✓	−2.06261
Hock and Schittkowski 29 problem	1	3	151	$(4,2\sqrt{2},2),$ $(-4,2\sqrt{2},-2),$ $(-4,-2\sqrt{2},2),$ $(4,-2\sqrt{2},-2)$	$(4,2\sqrt{2},2),$ $(-4,2\sqrt{2},-2),$ $(-4,-2\sqrt{2},2),$ $(4,-2\sqrt{2},-2)$	✓ ✓ ✓ ✓	$-16\sqrt{2}$
Dekkers and Aarts problem	2	2	178	$(0,15),(0,-15),$ $(0,0)$	$(0,15),(0,-15),$ $(0,0)$	✓ ×	−24780
Modified Branin problem	2	2	182	$(-\pi,12.275),$ $(\pi,2.275),$ $(9.425,2.477)$	$(-\pi,12.275),$ $(\pi,2.275),$ $(9.42478,2.475)$	✓ ✓ ✓	0.397887
Modified six-hump camel back problem	3	2	233	$(-0.08984,0.7127),$ $(0.08984,-0.7126)$	$(-0.0898,0.7126),$ $(0.0898,-0.7126)$	✓ ✓	−1.0316

Table 6. **Performance of SHIPA, TGIPA and MIDACO on 38 test problems.**

Test problem	n	N	time (s)		
			SHIPA	TGIPA	MIDACO
BL	2	52	0.004	0.005	0.004
ACK	4	123	0.006	0.005	0.687
AP	2	59	0.004	0.005	0.003
B1	2	115	0.006	0.004	0.006
B2	2	283	0.005	0.005	0.005
BR	2	67	0.008	0.005	0.011
CB3	2	304	0.005	0.005	0.006
CB6	2	89	0.004	0.004	0.005
CM	4	30	0.006	0.005	0.016
DA	2	66	0.004	0.005	0.011
EP	2	45	0.004	0.006	0.004
EXP	4	114	0.002	0.005	0.015
GP	2	86	0.007	0.005	0.010
GW	4	152	0.004	0.005	0.009
GRP	3	324	0.612	0.895	0.785
H3	3	201	0.007	0.006	0.015
HV	3	131	0.009	0.005	0.008
HSK	2	136	0.004	0.005	0.004
KL	4	128	0.006	0.008	0.019
LM1	3	46	0.006	0.007	0.007
LM2	4	74	0.010	0.039	0.011
MC	2	127	0.003	0.015	0.004
MRP	2	445	0.004	0.004	0.007
MGP	2	309	0.003	0.005	0.007
NF2	4	87	0.017	0.029	0.025
NF3	4	96	0.005	0.005	0.007
PRD	2	203	0.004	0.005	0.014
PQ	4	242	0.009	0.004	0.102
RG	2	195	0.004	0.005	0.010
RB	4	604	0.011	0.014	0.040
SAL	4	122	0.006	0.005	0.203
SF1	2	452	0.005	0.004	0.016
SF2	2	66	0.004	0.005	0.617
SBT	2	128	0.008	0.010	0.021
S5	4	773	0.009	0.010	0.106
S7	4	398	0.012	0.021	0.104
S10	4	278	0.008	0.014	0.017
SIN	4	315	0.008	0.009	0.019
WP	4	487	0.010	0.012	0.026

Table 7. Representative computational cost of minimizer pool construction and interior-point refinement for selected test problems.

Test problem	# Sample points N	Pool construction time (ms)	Interior-point time (ms)	Pool fraction of total time (%)
WP	487	4.31	5.69	43
RB	604	5.06	5.94	46
S5	773	4.05	4.95	45

6. Conclusions

In this work, we have developed a simplicial homology interior-point method for solving box- and inequality-constrained nonlinear (nonconvex) programming problems. This method uses the simplicial homology technique to generate initial starting points for a local interior-point algorithm, facilitating the subsequent search for the global optimal solution within the complex nonconvex domain. We have also implemented the proposed simplicial interior-point method to evaluate its efficiency and compared it against existing methods, including the topographical interior-point algorithm and other well-known solvers, across a range of test problems. The computational results demonstrate that the simplicial interior-point algorithm performs well in practice and consistently achieves superior performance, outperforming both the topographical interior-point algorithm and the MIDACO solver.

Acknowledgments: The authors sincerely thank the anonymous referees for their careful review and valuable suggestions, which have significantly improved the presentation of this paper.

Competing Interests: The authors have no competing interests to declare that are relevant to the content of this article.

Data availability: Data sharing does not apply to this article, as no datasets were generated or analyzed during this study.

References

[1] H. Abedi and B. Kheirfam, *An efficient second-order predictor–corrector infeasible primal–dual IPM algorithm with large iteration path updates for solving well-known SDO problems*, J. Comput. Appl. Math. **459** (2025), no. C, 116379. https://doi.org/10.1016/j.cam.2024.116379.

[2] M.M. Ali, C. Khompatraporn, and Z.B. Zabinsky, *A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems*, J. Global

Optim. **31** (2005), 635–672.
https://doi.org/10.1007/s10898-004-9972-2.

[3] B. Alzalg, *Combinatorial and Algorithmic Mathematics: From Foundation to Optimization*, John Wiley and Sons, 2024.

[4] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter, *Branching and bounds tighteningtechniques for non-convex MINLP*, vol. 24, Taylor and Francis, Inc., USA, 2009.

[5] E.K.P. Chong and S.H. Żak, *An Introduction to Optimization*, John Wiley and Sons, 2008.

[6] A. Dekkers and E. Aarts, *Global optimization and simulated annealing*, Math. Program. **50** (1991), no. 1, 367–393.
https://doi.org/10.1007/BF01594945.

[7] S.C Endres, C. Sandrock, and W.W. Focke, *A simplicial homology algorithm for Lipschitz optimisation*, J. Global Optim. **72** (2018), no. 2, 181–217.
https://doi.org/10.1007/s10898-018-0645-y.

[8] C. Floudas and P.M. Pardalos, *Recent Advances in Global Optimization*, Princeton University Press, 1992.

[9] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley series in artificial intelligence, Addison-Wesley, 1989.

[10] A. Hatcher, *Algebraic Topology*, Cambridge University Press, Cambridge, 2002.

[11] N. Henderson, M. de Sá Rêgo, W.F. Sacco, and R.A. Rodrigues Jr, *A new look at the topographical global optimization method and its application to the phase stability analysis of mixtures*, Chemical Engineering Science **127** (2015), 151—-174.
https://doi.org/10.1016/j.ces.2015.01.029.

[12] M. Henle, *A Combinatorial Introduction to Topology*, Dover, New York, NY, 1994.

[13] J. Herskovits, *Feasible direction interior-point technique for nonlinear optimization*, J. Optim. Theory Appl. **99** (1998), no. 1, 121–146.
https://doi.org/10.1023/A:1021752227797.

[14] W. Hock and K. Schittkowski, *Test Examples for Nonlinear Programming Codes*, Springer-Verlag, Berlin, Heidelberg, 1981.

[15] J. Kennedy and R. Eberhart, *Particle swarm optimization*, Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948.
https://doi.org/10.1109/ICNN.1995.488968.

[16] B. Kheirfam, A. Nasrollahi, and M. Mohammadi, *A second-order corrector infeasible interior-point method for semidefinite optimization based on a wide neighborhood*, J. Sci. Comput. **86** (2021), no. 1, 13.
https://doi.org/10.1007/s10915-020-01384-w.

[17] T.T. Lu and S.H. Shiou, *Inverses of 2 x 2 block matrices*, Computers and Mathematics with Applications **43** (2002), no. 1-2, 119–129.

[18] D.G. Luenberger, *Introduction to Linear and Nonlinear Programming*, Addison-Wesley Pub. Co., Reading, Mass, 1973.

[19] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, pp. 1–664, Springer Nature, 2006.

[20] E.R. Panier, A.L. Tits, and J.N. Herskovits, *Feasible direction interior-point technique*

for nonlinear optimization, SIAM J. Control Optim. **24** (1988), no. 4, 788–811. https://doi.org/10.1137/0326046.

[21] G. Petelin, G. Cenikj, and T. Eftimov, *Tinytla: Topological landscape analysis for optimization problem classification in a limited sample setting*, Swarm and Evolutionary Computation **84** (2024), 101448. https://doi.org/10.1016/j.swevo.2023.101448.

[22] M.J.D. Powell, *The convergence of variable metric methods for nonlinearly constrained optimization calculations*, In Nonlinear Programming 3, Proceedings of the Special Interest Group on Mathematical Programming Symposium (University of Wisconsin–Madison), Elsevier, 1978, pp. 27–63.

[23] E. Sperner, *Neuer beweis für die invarianz der dimensionszahl und des gebietes*, Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg **6** (1928), no. 1, 265–272. https://doi.org/10.1007/BF02940617.

[24] R. Storn and K. Price, *Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces*, J. Global Optim. **11** (1997), no. 4, 341–359. https://doi.org/10.1023/A:1008202821328.

[25] F. Zhang, *Matrix Theory: Basic Results and Techniques*, Universitext, Springer New York, 2011.