LEMON: LANDSCAPE-EQUIVALENT CONTINUOUS RELAXATION FOR MULTI-STATE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Many combinatorial problems require assigning one of several discrete values to each variable, with an objective that couples pairs or larger groups of variables. Finding an assignment that globally minimizes this objective is often intractable, so a practical alternative is to seek one that cannot be improved by changing a single variable. Searching for such assignments can still be expensive, motivating the use of continuous relaxations and gradient-based optimization. The challenge is that local minima of a relaxation need not correspond to locally optimal discrete assignments. We introduce LEMON (Landscape Equivalent Multi-State Optimization), a continuous relaxation that preserves this correspondence. We represent each discrete variable by a probability vector over its possible values. The discrete objective is then extended to these vectors and augmented with a regularization term that promotes discrete assignments. Our main result shows that the local minima of the resulting continuous objective correspond exactly to the local optima of the original discrete problem. LEMON uses a projected variant of ADAM to optimize the probability vectors, decodes them into a discrete assignment, and then checks feasibility and local optimality. We evaluate LEMON on maximum independent set, graph coloring, Max-k-Cut, and k-way number partitioning. It matches the reference optimum on $53/61$ maximum independent set instances and achieves the known chromatic number on $27/30$ coloring instances. It also closely matches public-best Max-k-Cut values and outperforms specialized number-partitioning baselines, while certifying local optimality across all tasks.

1 INTRODUCTION

Discrete optimization. Many combinatorial optimization problems in scheduling, graph design, and inference involve assigning discrete states to objects subject to competing costs or constraints. In max-k-cut, graph coloring, and k-way number partitioning problem (NPP), these states represent groups, colors, and subsets, respectively. Binary problems, such as maximum independent set (MIS), instead decide whether to select each vertex. Although these problems share a finite-state structure, their objectives and the moves used to improve an assignment differ. Finding a global optimum is NP-hard in general, motivating both exact solvers and methods that seek good solutions within a limited computational budget. General-purpose solvers such as Gurobi and CP-SAT can certify global optimality, but may require substantial time on difficult instances (Gurobi Optimization, LLC, 2024; Perron & Didier, 2025).

Problem-specific methods. Practical solvers often exploit the structure of individual problems to improve solution quality and runtime. For max-cut, local search, physics-inspired Ising machines, and graph neural networks provide heuristic approaches (Benlic & Hao, 2013; Goto et al., 2021; Shen et al., 2025; Schuetz et al., 2022), while max-k-cut has motivated methods based on multiple search operators and evolutionary strategies (Ma & Hao, 2017; Shao & Yang, 2026). For MIS, ReduMIS combines graph reductions with local search (Lamm et al., 2017), alongside approaches based on differentiable optimization, reinforcement learning, GFlowNets, diffusion, and GPU sampling (Alkhouri et al., 2025; Ahn et al., 2020; Zhang et al., 2023; Sun & Yang, 2023; Sanokowski et al., 2024; Sun et al., 2023; Wu et al., 2026). Graph coloring methods include DSATUR, TabuCol, hybrid evolutionary search, and the learning-guided independent-set extraction (Brélaz, 1979; Hertz & de Werra, 1987; Galinier & Hao, 1999; Arora et al., 2026). Multi-way number partitioning similarly

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

has specialized recursive and local search methods (Korf, 2009; 2011; Chen et al., 2026; Zhang, 2024).

Continuous relaxations. Continuous relaxations replace discrete choices with continuous variables, making optimization methods that use gradients available. They can also provide bounds and approximation guarantees. For max-cut, the Goemans–Williamson algorithm solves a convex semidefinite relaxation and uses randomized rounding to obtain a provable approximation guarantee (Goemans & Williamson, 1995). Frieze and Jerrum develop a related approach for max-k-cut (Frieze & Jerrum, 1997), while continuous relaxations have also been used for multi-label inference (Lê-Hữu & Paragios, 2018). Semidefinite programs involve matrix variables, however, and general-purpose interior-point methods can become expensive as instances grow.

This cost motivates formulations that relax the original variables more directly. Penalty terms can incorporate discrete constraints into the objective, while simple remaining constraints can be enforced by projection. Such formulations are generally nonconvex, but they permit first-order search when gradients and projections are inexpensive. Because this search may approach a local minimum, the local landscape matters as much as the values attained at global optima. A relaxation may introduce a fractional local minimum or make a discrete assignment locally minimal even though changing one original variable improves it.

Preserving local minima has been studied in the broader setting of smooth parametrizations by Levin et al. (2025). For binary Ising models, Banerjee et al. (2026) establish *landscape equivalence*, under which the continuous local minima are exactly the embedded discrete one-flip local minima. Quadratic penalties and continuous MIS formulations provide related results under binary or graph-specific assumptions (Huang et al., 2026; Alkhouri et al., 2025). These results do not establish the same correspondence for general interactions among variables with three or more states. Although many such problems can be encoded as quadratic unconstrained binary optimization (QUBO) (Lucas, 2014), reassigning one original variable may require several bit flips. The encoding can therefore preserve global optima while changing the neighborhood that defines local optimality. We seek a continuous formulation that handles multiple states and higher-order interactions directly and preserves local minima under changes to one original variable.

Our approach. We introduce LEMON (Landscape Equivalent Multi-State Optimization) for finite-state objectives with interactions among subsets of variables. For each variable, we use a probability vector over its possible states and extend each interaction by taking its expectation under independent assignments. This extension is affine in each vector, and a simplex edge represents the reassignment of one variable. We add an attractor that excludes fractional local minima and control its inward slope to retain exactly the discrete one-flip local minima. We derive the sharp threshold for a shared slope and construct families of attractors with different curvature profiles.

For continuous search, LEMON uses a projected variant of ADAM. Its first-order updates avoid forming a Hessian and support batched computation when gradients and projections are efficient (Kingma & Ba, 2015). The solver then decodes the probability vectors and checks feasibility and one-flip local optimality using the original objective and exact energy changes. It applies discrete corrections only when these checks fail, allowing the same search procedure to work with problem-specific objectives and verification routines. The landscape theorem characterizes the minima of the formulation, but does not guarantee convergence of projected ADAM or bound the number of corrections.

Contributions. Our main contributions are summarized below.

- We develop a common solver for binary and categorical problems and derive exact one-flip certificates for MIS, clique, vertex cover, coloring, max-cut, max-k-cut, and k-way NPP. For NPP, the certificate checks feasible item reassignments using the maximum set discrepancy (MSD), the difference between the largest and smallest subset sums (Section 5 and Appendix C).

- We prove that, for finite-state objectives with interactions of arbitrary finite order, the continuous local minima are exactly the embedded discrete one-flip local minima. Every continuous local minimum is strict, and global minima remain preserved even when the attractor slope exceeds the range required for landscape equivalence (Sections 3 and 4).

- We determine the exact range of a shared attractor slope that preserves landscape equivalence and give a simpler sufficient bound from the smallest positive one-flip improvement. We also construct

normalized-curvature and label-resolved attractor families with different curvature profiles and label-dependent slopes (Section 4 and Appendix B).

- We compare solution quality and runtime with specialized solvers on public graph benchmarks, random graph ensembles, and generated NPP instances. Ablations examine the attractor slope, the original multi-state neighborhood, and the contribution of continuous search (Section 6 and Appendix E).

2 MULTI-STATE OPTIMIZATION

Many combinatorial problems assign vertices or items to discrete states, but differ in how they measure the quality of an assignment. Max-k-cut introduces the assignment and energy viewpoint; coloring, independent set, and number partitioning show how it extends to other objectives.

Max-k-cut. Given an undirected graph $G = (V, E)$ with $V = \{1, \dots, n\}$ and edge weights w_{ij}, max-k-cut divides the vertices into at most k groups. A label $s_i \in \{1, \dots, k\}$ specifies the group of vertex i, and the assignment $\boldsymbol{s} = (s_1, \dots, s_n)$ has cut value

$$\mathcal{C}(\boldsymbol{s}) = \sum_{\{i,j\} \in E} w_{ij} \mathbb{1}\{s_i \neq s_j\}, \tag{1}$$

where $\mathbb{1}\{\cdot\}$ is one when its condition holds and zero otherwise. The objective maximizes the total weight of edges joining different groups. For $k = 2$, this is the usual max-cut problem.

Equivalently, we can minimize the weight of edges within groups, expressed as the energy

$$\mathcal{E}_{k\text{-cut}}(\boldsymbol{s}) = \sum_{\{i,j\} \in E} w_{ij} \mathbb{1}\{s_i = s_j\}. \tag{2}$$

Each edge contributes to exactly one of the two sums, so $\mathcal{C}(\boldsymbol{s}) + \mathcal{E}_{k\text{-cut}}(\boldsymbol{s}) = \sum_{\{i,j\} \in E} w_{ij}$. The energy in (2) is a weighted Potts energy (Potts, 1952).

Graph coloring. The group labels can instead represent colors. For a fixed k, a proper coloring assigns different labels to adjacent vertices. Its conflict energy

$$\mathcal{E}_{\text{col},k}(\boldsymbol{s}) = \sum_{\{i,j\} \in E} \mathbb{1}\{s_i = s_j\} \tag{3}$$

has the same form as the max-k-cut energy with unit weights. Here the goal is to reach zero conflicts. The chromatic number is the smallest k for which that is possible.

Maximum independent set. Binary states describe whether each vertex is selected. An independent set contains no adjacent selected vertices, and the maximum independent set problem seeks one of largest cardinality. With $s_i \in \{0, 1\}$ indicating selection, define

$$\mathcal{E}_{\text{IS}}(\boldsymbol{s}) = -\sum_i s_i + 2 \sum_{\{i,j\} \in E} s_i s_j. \tag{4}$$

The first term rewards selection, while the second penalizes selecting adjacent vertices. Removing a selected vertex that has a selected neighbor lowers the energy, so every global minimizer is independent. On an independent set, the energy equals minus its cardinality; hence minimizing (4) gives a maximum independent set. A maximum independent set in $\overline{G}$ is a maximum clique in G, while the vertices outside a maximum independent set in G form a minimum vertex cover.

Number partitioning. The objects need not be graph vertices. In k-way number partitioning, a label $s_i \in \{1, \dots, k\}$ assigns a number $a_i \in \mathbb{R}$ to one of k subsets, where $2 \leqslant k \leqslant n$. We require nonempty subsets and seek to minimize the maximum set discrepancy (MSD), the difference between their largest and smallest sums

$$D(\boldsymbol{s}) = \max_{r \in \{1,\dots,k\}} S_r(\boldsymbol{s}) - \min_{r \in \{1,\dots,k\}} S_r(\boldsymbol{s}), \qquad S_r(\boldsymbol{s}) = \sum_{i:\, s_i = r} a_i. \tag{5}$$

Let $N_0(\boldsymbol{s})$ count the empty subsets. We incorporate the nonemptiness requirement by choosing $\Lambda > \sum_i |a_i|$ and defining

$$\mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) = D(\boldsymbol{s}) + \Lambda N_0(\boldsymbol{s}). \tag{6}$$

Since $D(\boldsymbol{s}) \leqslant \sum_i |a_i|$ and a nonempty partition exists when $k \leqslant n$, any assignment with an empty subset has greater energy than some valid partition. Thus, every global minimizer minimizes MSD over nonempty partitions. Unlike the graph energies above, this energy depends jointly on all item assignments.

General formulation. The examples vary both in the number of states available to a variable and in the number of variables involved in a cost. To describe these choices together, write $[m] = \{1, \dots, m\}$ and let variable $i \in [n]$ have $m_i \geqslant 2$ states labeled by $[m_i]$. With $\boldsymbol{m} = (m_1, \dots, m_n)$, the assignment space is

$$\mathbb{H}_{\boldsymbol{m}} = \prod_{i=1}^{n} [m_i].$$

An assignment $\boldsymbol{s} = (s_1, \dots, s_n) \in \mathbb{H}_{\boldsymbol{m}}$ specifies one state for every variable. max-k-cut, fixed-k coloring, and k-way number partitioning have $m_i = k$ for every variable. MIS has two states per vertex, conventionally labeled $\{0, 1\}$ rather than $\{1, 2\}$. Thus, $\mathbb{H}_{\boldsymbol{m}}$ generalizes the binary Hamming cube to variables with different numbers of states.

Each edge in max-k-cut contributes a cost involving the labels of its two endpoints. We call the set of variables involved in one cost a *factor scope* and collect these scopes in a finite set $\mathcal{F}$ of subsets of $[n]$. For each $f \in \mathcal{F}$, the function

$$\psi_f : \prod_{i \in f} [m_i] \to \mathbb{R}$$

assigns a cost to their joint state $\boldsymbol{s}_f = (s_i)_{i \in f}$. The general optimization problem is

$$\min_{\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}} \mathcal{E}(\boldsymbol{s}), \qquad \mathcal{E}(\boldsymbol{s}) := \sum_{f \in \mathcal{F}} \psi_f(\boldsymbol{s}_f). \tag{7}$$

For max-k-cut and coloring, every factor scope is an edge, so $\mathcal{F} = E$. Both use the edge cost

$$\psi_{\{i,j\}}(a, b) = w_{ij} \mathbb{1}\{a = b\}.$$

With the graph weights, this recovers the max-k-cut energy in (2); setting every $w_{ij} = 1$ gives the coloring conflict energy in (3).

MIS uses both vertex and edge scopes. After relabeling its two states as $\{0, 1\}$, we assign the costs

$$\psi_{\{i\}}(z_i) = -z_i, \qquad \psi_{\{i,j\}}(z_i, z_j) = 2 z_i z_j.$$

The first rewards selecting vertex i, while the second penalizes selecting both endpoints of an edge. For number partitioning, the discrepancy depends on the subset sums formed by all item assignments. We therefore use one factor with scope $[n]$ and cost $\psi_{[n]}(\boldsymbol{s}) = D(\boldsymbol{s}) + \Lambda N_0(\boldsymbol{s})$.

3 Continuous relaxation

The factor objective (7) assigns a cost to each discrete assignment. To preserve its local structure in a continuous formulation, we first specify which changes count as local moves. We then represent assignments in a continuous domain and extend the objective to that domain.

Local optimality. In max-k-cut, a local move transfers one vertex to another group. In coloring, it changes the color of one vertex; in number partitioning, it moves one item to another subset; and in MIS, it changes whether a vertex is selected. Each move changes one coordinate of the assignment while leaving the others fixed. We call such a move a *flip*.

To express a flip in the general notation, fix an assignment $\boldsymbol{s}$, a coordinate $i \in [n]$, and an alternative state $b \in [m_i] \setminus \{s_i\}$. Let $\boldsymbol{e}_i \in \mathbb{R}^n$ be the ith standard basis vector. The assignment after the flip is $\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i$, and its energy change is

$$\Delta_i^b(\boldsymbol{s}) = \mathcal{E}\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big) - \mathcal{E}(\boldsymbol{s}).$$

The flip improves the assignment when $\Delta_i^b(\boldsymbol{s}) < 0$.

In the factor formulation, only scopes containing i contribute to $\Delta_i^b(\boldsymbol{s})$. For max-$k$-cut, these scopes are the edges incident to vertex i. Writing $N(i)$ for its set of neighbours, we obtain

$$\Delta_i^b(\boldsymbol{s}) = \sum_{j \in N(i)} w_{ij}\big(\mathbb{1}\{s_j = b\} - \mathbb{1}\{s_j = s_i\}\big). \tag{8}$$

This is the weight of edges from i to its new group minus the weight of edges from i to its current group. A negative value therefore increases the cut.

Definition 3.1 (One-flip local minimum)**.** An assignment $\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}$ is a *one-flip local minimum* of $\mathcal{E}$ if $\Delta_i^b(\boldsymbol{s}) \geqslant 0$ for every $i \in [n]$ and $b \in [m_i] \setminus \{s_i\}$. The minimum is *strict* if every such gain is positive. We denote the set of one-flip local minima by $\mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$.

A flip changes one original variable, even when that variable has more than two states. Encoding its states with bits may require several bit flips for the same reassignment and can therefore change which assignments are locally optimal. For binary variables, the definition reduces to one-flip local optimality for Ising models (Banerjee et al., 2026).

Simplex representation. To represent every possible flip as a continuous direction, we give each state of variable i its own vertex. A probability vector $p_i = (p_{i,a})_{a\in[m_i]}$ lies in the simplex

$$\Delta_{m_i-1} := \Big\{ q \in \mathbb{R}^{m_i} : q_a \geqslant 0,\ \sum_{a=1}^{m_i} q_a = 1 \Big\}.$$

The relaxed assignment $p = (p_1, \ldots, p_n)$ belongs to $\mathbb{Q}_{\boldsymbol{m}} = \prod_{i=1}^n \Delta_{m_i-1}$. The standard basis vector $\boldsymbol{e}_a \in \mathbb{R}^{m_i}$ assigns probability one to state a, so the discrete assignment $\boldsymbol{s}$ embeds as the vertex $v_{\boldsymbol{s}} = (\boldsymbol{e}_{s_1}, \ldots, \boldsymbol{e}_{s_n})$.

The separate vertices matter when a variable has more than two states. If we instead represent $m > 2$ labels by scalar values $\nu_1, \ldots, \nu_m$, their convex hull is an interval with only two extreme points. Scalar-product costs are also restrictive: after removing unary row and column terms, the matrix $(\nu_a\nu_b)_{a,b}$ has rank at most one, whereas the centered Potts equality matrix has rank $m - 1$. The simplex representation retains every label as a vertex and accommodates general interaction costs.

Continuous extension. To extend the discrete energy, draw a state S_i for each variable independently according to p_i, and let $S = (S_1, \ldots, S_n)$ be the resulting assignment. We define the continuous energy as the average value of $\mathcal{E}(S)$. Since $\mathcal{E}$ is a sum of factor costs, this average can be computed factor by factor:

$$\begin{aligned} \mathcal{E}_{\mathbb{Q}}(p) &= \mathbb{E}[\mathcal{E}(S)] \\ &= \sum_{f\in\mathcal{F}} \sum_{a_f \in \prod_{i\in f}[m_i]} \psi_f(a_f) \prod_{i\in f} p_{i,a_i}, \end{aligned} \tag{9}$$

where $a_f = (a_i)_{i\in f}$ runs over the possible joint states of the variables in f. The product $\prod_{i\in f} p_{i,a_i}$ is the probability of that joint state.

At an embedded assignment $v_{\boldsymbol{s}}$, every state choice is fixed, so $\mathcal{E}_{\mathbb{Q}}(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s})$. With all other probability vectors fixed, $\mathcal{E}_{\mathbb{Q}}$ is affine in p_i. These properties uniquely determine the extension because each p_i is a convex combination of simplex vertices (Lemma A.1). To connect this extension to a flip, let $\boldsymbol{e}_{ia}$ have $\boldsymbol{e}_a$ in block i and zeros in every other block. The edge from $v_{\boldsymbol{s}}$ toward state b has direction $\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i}$. Block affinity gives (Lemma A.2)

$$\mathcal{E}_{\mathbb{Q}}\big(v_{\boldsymbol{s}} + t(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i})\big) = \mathcal{E}(\boldsymbol{s}) + t\Delta_i^b(\boldsymbol{s}), \qquad t \in [0,1]. \tag{10}$$

Thus, the slope along this edge is exactly the energy change of the corresponding discrete flip.

Attractor. The continuous extension $\mathcal{E}_{\mathbb{Q}}$ agrees with $\mathcal{E}$ at every embedded assignment, and its slope along the simplex edge for a flip $i \to b$ equals $\Delta_i^b(\boldsymbol{s})$. It can nevertheless have fractional local minima. We add a term that is strictly concave in each probability vector to exclude such minima. Its inward slope must also be controlled, since an excessive slope could make an assignment locally minimal despite an improving flip.

Definition 3.2 (Admissible attractor)**.** Fix an inward slope $\rho_i > 0$ for each $i \in [n]$. An *admissible attractor* on $\mathbb{Q}_{\boldsymbol{m}}$ is a separable function $\mathcal{A}(p) = \sum_{i=1}^n \alpha_i(p_i)$ whose block functions α_i are continuously differentiable and strictly concave on a neighborhood of Δ_{m_i-1}. Each block vanishes at the simplex vertices and has inward slope ρ_i along every edge:

$$\begin{aligned} \alpha_i(\boldsymbol{e}_a) &= 0 \quad (a \in [m_i]), \\ \frac{d}{dt}\alpha_i\big((1-t)\boldsymbol{e}_a + t\boldsymbol{e}_b\big)\Big|_{t=0^+} &= \rho_i \quad (a, b \in [m_i],\ a \neq b). \end{aligned}$$

We call each α_i an *admissible block attractor*.

The continuous objective that we minimize over $\mathbb{Q}_{\boldsymbol{m}}$ is the relaxed Hamiltonian

$$\mathcal{H}(p) = \mathcal{E}_{\mathbb{Q}}(p) + \mathcal{A}(p), \qquad p \in \mathbb{Q}_{\boldsymbol{m}}. \tag{11}$$

Since $\mathcal{A}$ vanishes at every embedded assignment, $\mathcal{H}(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s})$. Whether these vertices remain locally minimal depends on the balance between the flip gains and the inward attractor slopes, which we examine next.

4 Landscape equivalence

At an embedded assignment $v_{\boldsymbol{s}}$, the inward slope of $\mathcal{H}$ toward an alternative state b is $\Delta_i^b(\boldsymbol{s}) + \rho_i$. An attractor slope that is too large can therefore block a discrete improving move. A sufficient choice is to keep each slope below the smallest positive decrease available from any flip. We define this margin by

$$\gamma(\mathcal{E}) = \min\Big\{ -\Delta_i^b(\boldsymbol{s}) : \boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}},\ i \in [n],\ b \in [m_i] \setminus \{s_i\},\ \Delta_i^b(\boldsymbol{s}) < 0 \Big\}, \tag{12}$$

If $\mathcal{E}$ is nonconstant, this set is nonempty: any two assignments can be connected by flips, so some flip must change the energy and is improving when taken from its higher-energy endpoint. If $\mathcal{E}$ is constant, we set $\gamma(\mathcal{E}) = +\infty$.

We now use this improvement margin to relate local minima of the relaxed Hamiltonian to one-flip local minima of the discrete energy. Let $\mathrm{loc}_{\mathbb{Q}_{\boldsymbol{m}}}(\mathcal{H})$ denote the local minima of $\mathcal{H}$ relative to $\mathbb{Q}_{\boldsymbol{m}}$. The set $\mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$ contains the one-flip local minima defined in Definition 3.1. The following theorem shows when the continuous local minima are exactly the embedded discrete ones.

Theorem 4.1 (Landscape equivalence)**.** *Suppose every α_i is admissible and $0 < \rho_i \leqslant \gamma(\mathcal{E})$ for every $i \in [n]$. Then*

$$\mathrm{loc}_{\mathbb{Q}_{\boldsymbol{m}}}(\mathcal{H}) = \big\{ v_{\boldsymbol{s}} : \boldsymbol{s} \in \mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E}) \big\}.$$

Moreover, every local minimum of $\mathcal{H}$ on $\mathbb{Q}_{\boldsymbol{m}}$ is strict.

To see the role of the two assumptions, fix all but one probability vector. The extension is affine in that vector, while its attractor is strictly concave, so a fractional block cannot occur at a local minimum. At a vertex $v_{\boldsymbol{s}}$, a discrete one-flip local minimum has $\Delta_i^b(\boldsymbol{s}) \geqslant 0$ for every flip. Its inward edge slopes are therefore positive, making $v_{\boldsymbol{s}}$ a strict local minimum. If $\boldsymbol{s}$ admits an improving flip, the bound $\rho_i \leqslant \gamma(\mathcal{E})$ makes the corresponding inward slope nonpositive. Strict concavity supplies a decreasing direction even when that slope is zero. The full proof appears in Appendix A.

Global optimality. The slope bound is needed for local equivalence, but preserving global minimizers requires only admissible attractors with positive slopes. Strict concavity within each block excludes fractional global minimizers, and $\mathcal{H}(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s})$ at every vertex. Consequently,

$$\operatorname*{arg\,min}_{p \in \mathbb{Q}_{\boldsymbol{m}}} \mathcal{H}(p) = \big\{ v_{\boldsymbol{s}} : \boldsymbol{s} \in \operatorname*{arg\,min}_{\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}} \mathcal{E}(\boldsymbol{s}) \big\}. \tag{13}$$

A slope above the local-equivalence threshold can still create additional local minima at vertices, even though it leaves the global minimizers unchanged.

Sharp slope condition. The sufficient condition $\rho_i \leqslant \gamma(\mathcal{E})$ keeps every improving flip decreasing in the relaxation. Landscape equivalence requires less: at each assignment that is not one-flip locally optimal, one decreasing edge from its embedded vertex is enough. For an assignment $\boldsymbol{s}$, define

$$M(\boldsymbol{s}) = \max_{\substack{i \in [n] \\ b \in [m_i] \setminus \{s_i\}}} -\Delta_i^b(\boldsymbol{s}).$$

An improving flip exists exactly when $M(\boldsymbol{s}) > 0$. For such an assignment, $M(\boldsymbol{s})$ is its largest one-flip improvement. The smallest of these best improvements is

$$\Gamma(\mathcal{E}) = \min_{\boldsymbol{s}:\, M(\boldsymbol{s}) > 0} M(\boldsymbol{s}),$$

with $\Gamma(\mathcal{E}) = +\infty$ when every assignment is one-flip locally optimal.

For a common inward slope, $\Gamma(\mathcal{E})$ is the candidate threshold beyond which a nonoptimal assignment can lose its last decreasing edge. The following theorem establishes the exact threshold and extends the criterion to slopes that vary across blocks.

Theorem 4.2 (Sharp common slope)**.** *Let $\mathcal{A}$ be admissible with inward slopes $\rho_i > 0$. For a common inward slope $\rho_i = \rho$, landscape equivalence holds if and only if $\rho \leqslant \Gamma(\mathcal{E})$. For possibly unequal slopes, landscape equivalence holds if and only if every assignment with $M(\boldsymbol{s}) > 0$ admits a one-flip move $\boldsymbol{s} \to \boldsymbol{s}'$ in coordinate i such that $\mathcal{E}(\boldsymbol{s}) - \mathcal{E}(\boldsymbol{s}') \geqslant \rho_i$.*

Equality is included in the threshold. An edge with zero initial slope still decreases $\mathcal{H}$ because the attractor is strictly concave.

The margin $\gamma(\mathcal{E})$ controls every improving flip, whereas the sharp threshold $\Gamma(\mathcal{E})$ requires only one descent edge at each assignment that is not locally optimal. Since $\gamma(\mathcal{E})$ can be easier to bound, the following comparison shows what may be lost by using it.

Proposition 4.3. *The sufficient margin $\gamma(\mathcal{E})$ in* (12) *and the sharp threshold $\Gamma(\mathcal{E})$ satisfy $\gamma(\mathcal{E}) \leqslant \Gamma(\mathcal{E})$. The inequality can be strict.*

For a common slope, $(0, \gamma(\mathcal{E})]$ is therefore always within the exact admissible range $(0, \Gamma(\mathcal{E})]$. The proofs are in Appendix A. Figure 4 in Appendix E illustrates the additional continuous local minima that appear when the slope exceeds $\Gamma(\mathcal{E})$.

Computing $\Gamma(\mathcal{E})$ may require examining every discrete assignment. A lattice of factor costs provides a sufficient slope without computing either margin. Suppose $\psi_f(a_f) \in \delta\mathbb{Z}$ for every factor f and joint state a_f, where $\delta > 0$. Every nonzero one-flip energy change then has magnitude at least δ, so $\gamma(\mathcal{E}) \geqslant \delta$. Consequently, $0 < \rho_i \leqslant \delta$ guarantees landscape equivalence by Theorem 4.1. For integer-valued factor energies, any slope in $(0, 1]$ is sufficient (Corollary A.6).

Admissible attractors. The slope conditions leave room for different curvature profiles. Appendix B develops normalized-curvature and label-resolved families. For the solver, we use the quadratic attractor

$$\mathcal{A}_\eta(p) = \eta \sum_{i=1}^{n} \big(1 - \|p_i\|_2^2\big), \qquad \nabla_{p_i} \mathcal{A}_\eta(p) \quad = -2\eta p_i, \tag{14}$$

where $\eta > 0$. Each block term vanishes at the vertices, is strictly concave, and has inward slope 2η. Hence $2\eta \leqslant \gamma(\mathcal{E})$ is sufficient for landscape equivalence, while $2\eta \leqslant \Gamma(\mathcal{E})$ is the sharp common-slope condition. The global minimizers remain preserved for every $\eta > 0$ by (13).

5 Optimization and certification

Landscape equivalence gives LEMON a precise target for continuous search. When the quadratic attractor slope 2η satisfies Theorem 4.1, every local minimum of the exact Hamiltonian $\mathcal{H}$ is an embedded one-flip local minimum of $\mathcal{E}$. In particular, the formulation has no fractional local minima. A finite search run need not reach a local minimum, however, so LEMON checks the decoded assignment using the original discrete problem. Figure 3 summarizes this procedure, and Table 3 lists the quantities supplied for each application.

Search and decoding. For MIS, coloring, and max-k-cut, LEMON searches the exact Hamiltonian $\mathcal{H} = \mathcal{E}_{\mathbb{Q}} + \mathcal{A}_\eta$ with a projected variant of ADAM. After each update, it projects every probability vector onto its simplex. The NPP energy also has an exact extension, but its factor couples all items. Appendix D derives its gradient and the block expression available when the other item assignments are fixed, and describes the search used for each problem. At the end of the search budget, LEMON assigns each variable a state of maximum probability.

Verification. For the decoded assignment, LEMON evaluates the original energy, checks feasibility, and computes the energy change for every permitted flip. These checks give each certificate its meaning in the original problem. For MIS, independence and nonnegative one-flip gains certify a maximal independent set, which also yields a maximal clique in $\overline{G}$ and an inclusion-minimal vertex cover in G (Proposition C.1). For max-k-cut, the gains certify that no vertex can increase the cut by changing groups (Proposition C.3). For k-way NPP, certification requires nonempty groups and verifies that no reassignment preserving nonemptiness reduces the exact maximum set discrepancy (Proposition C.4).

Coloring requires one further test. A coloring with conflicts may have no improving flip, so we report a proper k-coloring only when its conflict energy is zero (Proposition C.2). Table 3 gives the certificate formulas, and Appendix C gives their proofs.

Corrections. When the decoded assignment passes the applicable checks, LEMON returns it without correction. Otherwise, it applies feasibility-restoring or strictly improving discrete moves and checks again. A fixed-k coloring that never reaches zero conflicts is reported as an unsuccessful run, not as evidence that the graph is not k-colorable. Thus, the final certificate depends on the original objective and permitted flips rather than on when continuous search stops.

6 EXPERIMENTS

We evaluate the solution quality and runtime of LEMON on various graph and number-partitioning problems. We present representative comparisons in Tables 1 and 2 and report the complete benchmark results, protocols, and ablations in the Appendix.

We implement LEMON in `Python` and run all experiments on the same system with 32 GB RAM and an AMD Ryzen 7 9700X 8-Core 64-bit processor. Reported solver times include optimization, any discrete corrections, and final verification, but exclude input loading and output display. Unless stated otherwise, we use ADAM (Kingma & Ba, 2015) with learning rate 0.08, first and second moment coefficients 0.9 and 0.999, and numerical stabilizer 10^{-8}. Projected search runs for $T \in \{500, 1000\}$ steps, depending on the experiment.

For integer-valued graph energies, we set the quadratic attractor coefficient to $\eta = 0.25$, giving an inward slope of $2\eta = 0.5$ within the sufficient range. For other exact lifted energies whose factor costs are integer multiples of a known $\delta > 0$, we use $\eta = \delta/2$ in accordance with Corollary A.6. These slope guarantees concern the exact lifted Hamiltonian. We initialize each graph probability vector by normalizing independent uniform samples. NPP starts from a greedy longest processing time (LPT) partition (Graham, 1969). Changes to these settings or budgets are stated with the corresponding results.

Figure 1 in the Appendix illustrates assignments found by LEMON for MIS, k-coloring, and Max-k-Cut on three benchmark graphs. Appendix E examines the sharp slope transition on examples from all four problem classes, including settings outside the theorem's conditions (Figure 4). It also compares the multi-state and one-hot binary landscapes, studies how the projected-search budget affects the need for discrete correction (Figure 5), and isolates the contribution of projected ADAM using baselines with matched correction procedures (Tables 4 and 5). Appendix F gives the complete results for public coloring and MIS benchmarks, random ER and BA MIS graphs, G-set Max-k-Cut, and k-way NPP.

Graph coloring, MIS, and max-cut. Table 1 presents representative public instances. For graph coloring, the reference chromatic numbers (χ) and the results of the GBS and W-GBS solvers proposed by Arora et al. (2026) are shown, with '$*$' denoting unknown χ value. LEMON attains the known chromatic number on most instances for which χ is known. In most MIS/clique instances, it matches or exceeds the best results from pCQO-MIS (Alkhouri et al., 2025) and ReduMIS (Lamm et al., 2017). On the G-set max-cut subset, LEMON improves on most reproduced D-Wave D-Wave (2023), FEM (Shen et al., 2025), MiP-CRIM (Banerjee et al., 2026) and Pi-GNN (Schuetz et al., 2022) values and matches several best-known cuts reported in the literature (Ma & Hao, 2017). Complete coloring and MIS results are in Appendix F.1 and F.2. Additional MIS comparisons on random Erdős–Rényi (ER) and Barabási–Albert (BA) graphs appear in Appendix F.3.

Max-k-cut and k-way NPP. For max-k-cut, LEMON closely matches the public PEAF reference values from Shao & Yang (2026). It attains the edge-count upper bound on G48 and G49 for $k \in \{2, 3, 4, 5\}$, on G50 for $k \in \{3, 4, 5\}$, and on G70 for $k \in \{4, 5\}$. For k-way NPP, LEMON achieves the lowest MSD among the compared methods (e.g. Swap from Zhang (2024)) in all six tested settings. Table 2 shows selected results. Complete comparisons on max-k-cut and k-way NPP are in Appendix F.4 and F.5 respectively.

Discussion. LEMON combines continuous search with exact discrete verification. For objectives covered by our theorem, the relaxation has no fractional local minima and preserves exactly the discrete one-flip local minima. The ablations study in Appendix E support this mechanism directly: exhaustive enumeration recovers the predicted sharp slope boundary, with the relaxed and discrete landscapes agreeing throughout $\rho \leqslant \Gamma(\mathcal{E})$ and spurious relaxed minima appearing immediately above it (Figure 4). The projected variant of ADAM supplies the same continuous update rule for binary and categorical variables, using problem-specific objectives and gradients. After decoding, feasibility

Table 1: *Representative results for graph coloring, MIS/maximum clique, and G-set max-cut.* Arrows indicate the preferred objective direction. For coloring, χ gives the known chromatic number while '$*$' indicates value is unknown. For MIS/maximum clique, 'Opt' denotes the largest reference value from Marino et al. (2024). Bold and underlining mark the best and second-best distinct solver values, respectively. The columns n and $|E|$ give the numbers of vertices and edges. Runtimes are for LEMON and include final verification.

Graph coloring (↓)

Graph	n	$\|E\|$	χ	GBS	W-GBS	DSATUR	LEMON	Runtime (s)
le450_5a	450	5714	5	8	7	10	**5**	0.3557
queen8_12	96	1368	12	13	13	14	**12**	0.3114
miles750	128	2113	31	32	32	**31**	**31**	0.3936
school1	385	19095	$*$	19	18	17	**14**	0.9813
flat300_20_0	300	21375	20	21	21	42	**20**	1.3679

MIS / maximum clique (↑)

Graph	n	$\|E\|$	Opt	pCQO-MIS	ReduMIS	LEMON	Runtime (s)
brock400_1	400	20077	**27**	25	25	**27**	1.3091
brock200_2	200	10024	**12**	11	**12**	**12**	0.5251
C2000.5	2000	999164	**16**	15	**16**	**16**	9.6043
gen200_p0.9_44	200	1990	**44**	42	**44**	**44**	1.1982

G-set max-cut (↑)

Graph	n	$\|E\|$	D-Wave	FEM	PI-GNN	MiP-CRIM	LEMON	Runtime (s)
G14	800	4694	3056	3024	3026	3056	**3057**	1.2811
G15	800	4661	3046	3003	2990	3037	**3048**	1.2807
G49	3000	6000	**6000**	5564	5918	**6000**	**6000**	0.2691
G50	3000	6000	5852	5590	5820	5856	**5880**	0.5382
G55	5000	12498	**10259**	9855	10138	10173	10219	1.3597

Table 2: *Results for max-k-cut and k-way NPP.* Max-k-cut reference values are from PEAF (Shao & Yang, 2026). For NPP, Swap (Zhang, 2024) achieves the lowest MSD among the compared baselines. Arrows indicate the preferred objective direction, and bold marks the best value in each row.

Problem	Instance	n	k	LEMON	Strongest baseline
Max-k-cut (↑)	G48	3000	3	**6000**	**6000** (PEAF)
Max-k-cut (↑)	G49	3000	4	**6000**	**6000** (PEAF)
Max-k-cut (↑)	G50	3000	5	**6000**	**6000** (PEAF)
Max-k-cut (↑)	G70	10000	5	**9999**	**9999** (PEAF)
k-way NPP (↓)	$\mathcal{U}$[32-bit]	1000	5	$\mathbf{1.16\times10^4}$	3.43×10^4 (Swap)
k-way NPP (↓)	$\mathcal{U}$[32-bit]	1000	10	$\mathbf{1.17\times10^4}$	7.22×10^4 (Swap)

checks and exact one-flip gains determine whether discrete corrections are needed. Experiments across MIS, coloring, max-k-cut, and multi-way number partitioning demonstrate the effectiveness of this common framework. It competes with specialized solvers on public graph benchmarks, achieves strong MIS results across random-graph densities and preferential-attachment degrees, and runs efficiently on the common CPU platform.

7 CONCLUSION

We introduced LEMON, a common solver for finite-state combinatorial problems that lifts each variable to a probability simplex and adds an admissible attractor. We prove that the resulting Hamiltonian has exactly the embedded one-flip local minima of the discrete objective when the shared inward slope satisfies a sharp bound. Our analysis also shows that global minimizers are preserved without this slope restriction.

LEMON uses a projected variant of ADAM for continuous search, then decodes the result and checks feasibility and exact one-flip energy changes. It applies discrete corrections when needed. These checks certify returned solutions in the original discrete neighborhood without assuming convergence of the continuous search. Across MIS, coloring, Max-k-Cut, and k-way NPP, the solver produces competitive solutions with verified local optimality.

AI USE STATEMENT

We used generative AI tools to polish phrasing, improve clarity and flow, and correct grammar and LaTeX formatting. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The appendix provides the necessary assumptions, proofs, optimization settings, benchmark definitions, search budgets, and complete results. All the experiments were run on the same machine with 32 GB RAM and an AMD Ryzen 7 9700X 8-core 64-bit processor. The source code is included in supplementary material and will be made public after acceptance.

REFERENCES

Sungsoo Ahn, Younggyo Seo, and Jinwoo Shin. Learning what to defer for maximum independent sets. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 134–144. PMLR, 2020.

Ismail Alkhouri, Cedric Le Denmat, Yingjie Li, Cunxi Yu, Jia Liu, Rongrong Wang, and Alvaro Velasquez. Differentiable quadratic optimization for the maximum independent set problem. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 1102–1127, 2025.

Urvashi Arora, Prisha Jain, Ashutosh Rai, and Sandeep Kumar. Graph coloring via learning and metric-guided independent set extraction. *Transactions on Machine Learning Research*, 2026.

Debraj Banerjee, Santanu Mahapatra, and Kunal N. Chaudhury. Local-minima-preserving polynomial relaxation of ising problems. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286 (5439):509–512, 1999. doi: 10.1126/science.286.5439.509.

Una Benlic and Jin-Kao Hao. Breakout local search for the max-cut problem. *Engineering Applications of Artificial Intelligence*, 26(3):1162–1173, 2013. doi: 10.1016/j.engappai.2012.09.001.

Daniel Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4): 251–256, 1979. doi: 10.1145/359094.359101.

B. Chen, C. Luo, J. Ding, Q. Zhang, Z. Su, and Z. L
"u. An adaptive configuration-aware simulated annealing for the maximally diverse grouping problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 36802–36810, 2026.

D-Wave. D-wave ocean software documentation, 2023. URL `https://docs.ocean.dwavesys.com/`.

Paul Erdős and Alfréd Rényi. On random graphs. i. *Publicationes Mathematicae Debrecen*, 6: 290–297, 1959.

Alan Frieze and Mark Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, 18:67–81, 1997.

Philippe Galinier and Jin-Kao Hao. Hybrid evolutionary algorithms for graph coloring. *Journal of Combinatorial Optimization*, 3:379–397, 1999. doi: 10.1023/A:1009823419804.

Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995. doi: 10.1145/227683.227684.

Hayato Goto, Kento Endo, Masaru Suzuki, Yoshisato Sakai, Taro Kanao, Yohei Hamakawa, Ryo Hidaka, Masaya Yamasaki, and Kosuke Tatsumura. High-performance combinatorial optimization based on classical mechanics. *Science Advances*, 7:eabe7953, 2021.

Ronald L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

Gurobi Optimization, LLC. Gurobi optimization, 2024. URL `https://www.gurobi.com/`.

Alain Hertz and Dominique de Werra. Using tabu search techniques for graph coloring. *Computing*, 39:345–351, 1987. doi: 10.1007/BF02239976.

Cheng-Han Huang, Yongliang Sun, Chaoyan Huang, Ismail Alkhouri, and Rongrong Wang. Local minima in quadratic-penalty relaxations of binary linear programs. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Richard E. Korf. Multi-way number partitioning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pp. 538–543, 2009.

Richard E. Korf. Objective functions for multi-way number partitioning. In *Proceedings of the Third Annual Symposium on Combinatorial Search (SoCS 2010)*, pp. 71–72, Stone Mountain, Georgia, 2010. AAAI Press.

Richard E. Korf. A hybrid recursive multi-way number partitioning algorithm. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pp. 591–596, 2011.

Sebastian Lamm, Peter Sanders, Christian Schulz, Darren Strash, and Renato F. Werneck. Finding near-optimal independent sets at scale. *Journal of Heuristics*, 23:207–229, 2017. doi: 10.1007/s10732-017-9337-x.

Khuê Lê-Hũu and Nikos Paragios. Continuous relaxation of MAP inference: A nonconvex perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Eitan Levin, Joe Kileel, and Nicolas Boumal. The effect of smooth parametrizations on nonconvex optimization landscapes. *Mathematical Programming*, 209:63–111, 2025.

Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2:5, 2014.

F. Ma and Jin-Kao Hao. A multiple search operator heuristic for the max-k-cut problem. *Annals of Operations Research*, 248(1):365–403, 2017. doi: 10.1007/s10479-016-2234-0.

Raffaele Marino, Lorenzo Buffoni, and Bogdan Zavalnij. A short review on the maximum clique problem algorithms with classical, ai, and quantum methods. *arXiv preprint arXiv:2403.09742*, 2024.

Laurent Perron and Frédéric Didier. CP-SAT, 2025. URL `https://developers.google.com/optimization/cp/cp_solver/`.

Renfrey B. Potts. Some generalized order-disorder transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48(1):106–109, 1952. doi: 10.1017/S0305004100027419.

Sebastian Sanokowski, Sepp Hochreiter, and Sebastian Lehner. A diffusion model framework for unsupervised neural combinatorial optimization. In *Proceedings of the 41st International Conference on Machine Learning*. PMLR, 2024.

Martin J. A. Schuetz, J. Kyle Brubaker, and Helmut G. Katzgraber. Combinatorial optimization with physics-inspired graph neural networks. *Nature Machine Intelligence*, 4(4):367–377, 2022.

Sihong Shao and Chuan Yang. A parallel evolutionary algorithm framework for graph k-CUT problems, 2026.

Z.-S. Shen, F. Pan, Y. Wang, Y.-D. Men, W.-B. Xu, M.-H. Yung, and P. Zhang. Free-energy machine for combinatorial optimization. *Nature Computational Science*, 5:322–332, 2025.

Haoran Sun, Katayoon Goshvadi, Azade Nova, Dale Schuurmans, and Hanjun Dai. Revisiting sampling for combinatorial optimization. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 32859–32874. PMLR, 2023.

Zhiqing Sun and Yiming Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. *Advances in Neural Information Processing Systems*, 36:3706–3731, 2023.

Yikai Wu, Haoyu Zhao, and Sanjeev Arora. Unrealized expectations: Comparing AI methods vs classical algorithms for maximum independent set. *Transactions on Machine Learning Research*, 2026.

B. Zhang. A new swap-based algorithm for solving multidimensional multi-way number partitioning problem. In *2024 21st International Computer Conference on Wavelet Active Media Technology and Information Processing*, pp. 1–4. IEEE, 2024.

Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron C. Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial problems with GFlowNets. *Advances in Neural Information Processing Systems*, 36:11952–11969, 2023.

Appendix

The appendices contain the supporting theory and proofs, implementation details, and extended empirical evidence. Appendix A contains the proofs of the landscape-equivalence results, while Appendix B develops the normalized-curvature and label-resolved attractor constructions. Appendices C–D collect the problem-specific certificates and implementation identities. Appendix E isolates the contributions of landscape equivalence, projected ADAM, and the discrete correction stage through controlled ablations, and Appendix F reports the complete benchmark protocols and results.

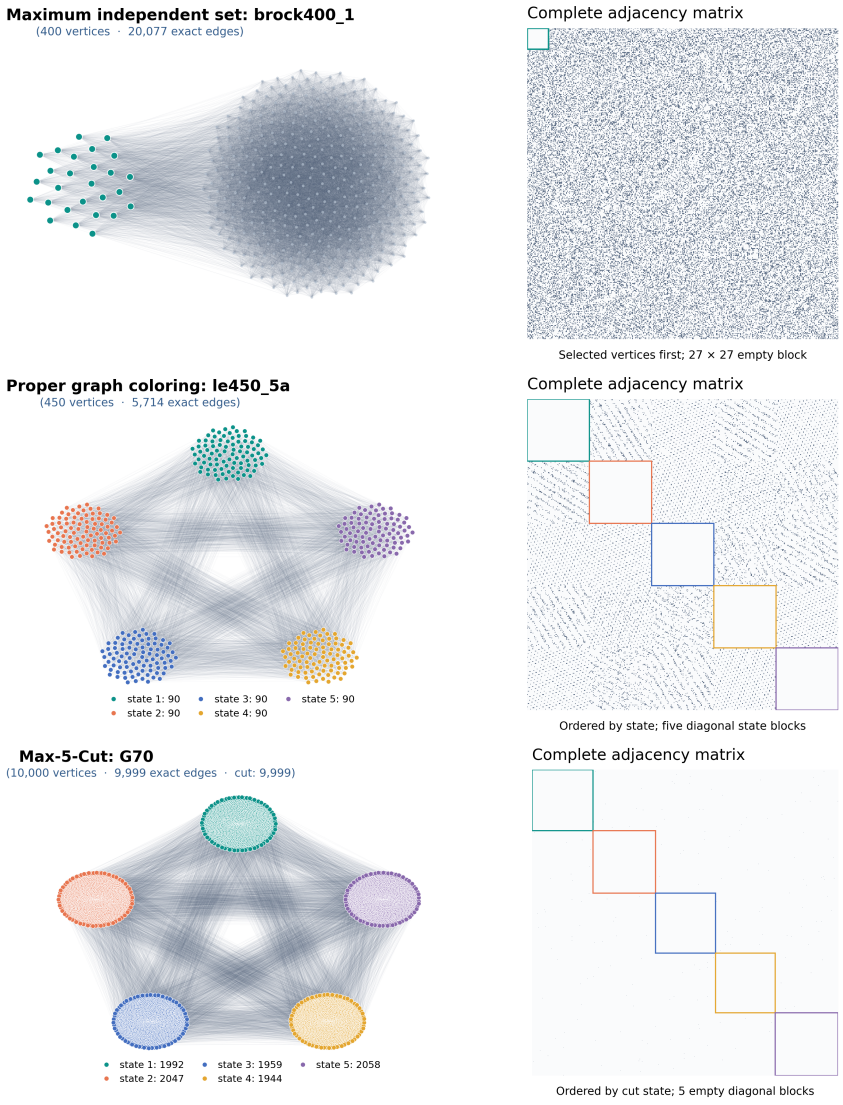


Figure 1: *Illustration of graph witnesses obtained by* LEMON *from 3 different benchmarks.* **Top:** the MIS graph derived from `brock400_1`, with the selected independent set highlighted and the corresponding empty adjacency block (LEMON *gives the exact MIS value of* 27 *for the graph*). **Middle:** `le450_5a` graph, with the five color classes and five empty diagonal blocks after reordering (LEMON *gives the exact colouring value of* $\chi = 5$ *for the graph*). **Bottom:** `G70` graph, with a five-state partition and five empty diagonal blocks after reordering (LEMON *cuts every unit-weight edge and attains the graph's total edge weight*).

A LANDSCAPE PROOFS

This appendix proves the landscape results stated in Section 4. Throughout, a factor scope contains distinct variables. If an interaction contains repeated occurrences of the same variable, those occurrences must first be identified; equivalently, the interaction is restricted to the diagonal corresponding to equal copies, and the result rewritten as a function of distinct variables. Otherwise the product-expectation lift would incorrectly treat the repeated occurrences as independent variables.

A.1 THE PRODUCT-EXPECTATION LIFT

Lemma A.1. *The function in* (9) *is the unique function on $\mathbb{Q}_{\boldsymbol{m}}$ that is affine in each block p_i when the other blocks are fixed and that agrees with $\mathcal{E}$ at every embedded assignment $v_{\boldsymbol{s}}$.*

Proof. Let $F : \mathbb{Q}_{\boldsymbol{m}} \to \mathbb{R}$ be affine in each block separately and satisfy $F(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s})$ for every $\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}$. Write $p_{-1} := (p_2, \dots, p_n)$ and, for $\boldsymbol{a} = (a_1, \dots, a_n) \in \mathbb{H}_{\boldsymbol{m}}$, let

$$v_{\boldsymbol{a}} := (\boldsymbol{e}_{a_1}, \dots, \boldsymbol{e}_{a_n}).$$

Because $p_1 = \sum_{a_1=1}^{m_1} p_{1,a_1} \boldsymbol{e}_{a_1}$ and F is affine in its first block,

$$F(p_1, p_{-1}) = \sum_{a_1=1}^{m_1} p_{1,a_1} F(\boldsymbol{e}_{a_1}, p_{-1}).$$

Applying the same identity successively to blocks $2, \dots, n$ gives

$$F(p) = \sum_{\boldsymbol{a} \in \mathbb{H}_{\boldsymbol{m}}} \left(\prod_{i=1}^{n} p_{i,a_i} \right) F(v_{\boldsymbol{a}}) = \sum_{\boldsymbol{a} \in \mathbb{H}_{\boldsymbol{m}}} \left(\prod_{i=1}^{n} p_{i,a_i} \right) \mathcal{E}(\boldsymbol{a}). \tag{15}$$

Thus any separately affine extension is uniquely determined by its values at the embedded discrete assignments.

Now let $S = (S_1, \dots, S_n)$ have independent coordinates with $\Pr(S_i = a) = p_{i,a}$. The last expression in (15) is $\mathbb{E}[\mathcal{E}(S)]$. Using the factorization (7),

$$\mathbb{E}[\mathcal{E}(S)] = \sum_{f \in \mathcal{F}} \sum_{a_f} \psi_f(a_f) \Pr(S_f = a_f) = \sum_{f \in \mathcal{F}} \sum_{a_f} \psi_f(a_f) \prod_{i \in f} p_{i,a_i},$$

which is exactly (9). Hence (9) exists, is affine in each block separately, agrees with $\mathcal{E}$ at every embedded assignment, and is unique. □

Lemma A.2. *Fix $\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}$, a variable i, and an alternative state $b \neq s_i$. For $0 \leqslant t \leqslant 1$,*

$$\mathcal{E}_{\mathbb{Q}}\big(v_{\boldsymbol{s}} + t(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i})\big) = \mathcal{E}(\boldsymbol{s}) + t\Delta_i^b(\boldsymbol{s}).$$

Here $\boldsymbol{e}_{ia}$ denotes the product-space vector whose ith block is $\boldsymbol{e}_a$ and whose other blocks are zero.

Proof. Set

$$p(t) = v_{\boldsymbol{s}} + t(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i}).$$

Only block i varies with t. By Lemma A.1, $t \mapsto \mathcal{E}_{\mathbb{Q}}(p(t))$ is affine. Its endpoint values are

$$\mathcal{E}_{\mathbb{Q}}(p(0)) = \mathcal{E}(\boldsymbol{s}), \qquad \mathcal{E}_{\mathbb{Q}}(p(1)) = \mathcal{E}\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big).$$

Linear interpolation therefore gives

$$\begin{aligned} \mathcal{E}_{\mathbb{Q}}(p(t)) &= (1-t)\mathcal{E}(\boldsymbol{s}) + t\,\mathcal{E}\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big) \\ &= \mathcal{E}(\boldsymbol{s}) + t\Delta_i^b(\boldsymbol{s}). \end{aligned}$$

□

A.2 CORNER LOCALIZATION AND THE EXACT VERTEX CRITERION

Lemma A.3. *Every local minimum of $\mathcal{H} = \mathcal{E}_{\mathbb{Q}} + \mathcal{A}$ on $\mathbb{Q}_{\boldsymbol{m}}$ is a vertex $v_{\boldsymbol{s}}$.*

Proof. Suppose, for contradiction, that a local minimum $p \in \mathbb{Q}_{\boldsymbol{m}}$ has a fractional block p_i. Then at least two coordinates of that block are positive; choose $a \neq b$ with $p_{i,a} > 0$ and $p_{i,b} > 0$. Define the product-space direction d by

$$d_i = \boldsymbol{e}_a - \boldsymbol{e}_b, \qquad d_j = 0 \quad (j \neq i).$$

For any $0 < \varepsilon < \min\{p_{i,a}, p_{i,b}\}$, both $p + \varepsilon d$ and $p - \varepsilon d$ remain in $\mathbb{Q}_{\boldsymbol{m}}$.

Along the line $p + td$, the extension $\mathcal{E}_{\mathbb{Q}}$ is affine in t because only block i varies. The terms $\alpha_j(p_j)$ with $j \neq i$ are constant, whereas $\alpha_i(p_i + td_i)$ is strictly concave. Hence $t \mapsto \mathcal{H}(p + td)$ is strictly concave near $t = 0$, and in particular

$$\mathcal{H}(p) > \frac{1}{2}\mathcal{H}(p + \varepsilon d) + \frac{1}{2}\mathcal{H}(p - \varepsilon d).$$

At least one of the two feasible neighboring points therefore has objective value strictly below $\mathcal{H}(p)$, contradicting local minimality. Thus every local minimum is a vertex. □

Lemma A.4. *For $\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}$, the vertex $v_{\boldsymbol{s}}$ is a local minimum of $\mathcal{H}$ if and only if*

$$\Delta_i^b(\boldsymbol{s}) + \rho_i > 0 \qquad \textit{for every } i \in [n] \textit{ and } b \in [m_i] \setminus \{s_i\}. \tag{16}$$

Whenever (16) *holds, $v_{\boldsymbol{s}}$ is a strict local minimum relative to $\mathbb{Q}_{\boldsymbol{m}}$.*

Proof. Fix $i \in [n]$ and $b \neq s_i$, and let $d_{ib} = \boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i}$. By Lemma A.2 and the definition of an admissible attractor, the one-sided derivative along the incident simplex edge is

$$\frac{d}{dt}\mathcal{H}(v_{\boldsymbol{s}} + td_{ib})\bigg|_{t=0^+} = \Delta_i^b(\boldsymbol{s}) + \rho_i.$$

If $\Delta_i^b(\boldsymbol{s}) + \rho_i < 0$, sufficiently small positive t gives descent, so $v_{\boldsymbol{s}}$ cannot be a local minimum. And if $\Delta_i^b(\boldsymbol{s}) + \rho_i = 0$, the restriction of $\mathcal{H}$ to that edge is the sum of an affine function and the strictly concave restriction of α_i, hence is strictly concave. For a differentiable strictly concave function h on an interval,

$$h(t) < h(0) + th'(0^+)$$

for every sufficiently small $t > 0$. Therefore

$$\mathcal{H}(v_{\boldsymbol{s}} + td_{ib}) < \mathcal{H}(v_{\boldsymbol{s}}) + t\big(\Delta_i^b(\boldsymbol{s}) + \rho_i\big) = \mathcal{H}(v_{\boldsymbol{s}}),$$

so the zero-derivative case is also incompatible with local minimality. This proves necessity of the strict inequalities.

Conversely, assume (16). Since there are finitely many incident edges,

$$c_0 := \min_{\substack{i \in [n] \\ b \neq s_i}} \big\{\Delta_i^b(\boldsymbol{s}) + \rho_i\big\} > 0.$$

Let $y \in \mathbb{Q}_{\boldsymbol{m}}$ be close to $v_{\boldsymbol{s}}$ and put $d = y - v_{\boldsymbol{s}}$. Feasibility of each simplex block gives the unique representation

$$d = \sum_{i=1}^{n} \sum_{b \neq s_i} \theta_{ib}(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i}), \qquad \theta_{ib} = y_{i,b} \geqslant 0.$$

For each block, $d_{i,s_i} = -\sum_{b \neq s_i} \theta_{ib}$, so

$$\|d\|_1 = 2\sum_{i=1}^{n} \sum_{b \neq s_i} \theta_{ib}.$$

Because $\mathcal{H}$ is continuously differentiable on a neighborhood of the product simplex,

$$\mathcal{H}(v_{\boldsymbol{s}} + d) - \mathcal{H}(v_{\boldsymbol{s}}) = \nabla\mathcal{H}(v_{\boldsymbol{s}})^{\top} d + r(d), \qquad r(d) = o(\|d\|_1).$$

Linearity of the directional derivative and (16) give

$$\nabla\mathcal{H}(v_{\boldsymbol{s}})^\top d = \sum_{i,b\neq s_i} \theta_{ib}\big(\Delta_i^b(\boldsymbol{s}) + \rho_i\big)$$
$$\geqslant c_0 \sum_{i,b\neq s_i} \theta_{ib} = \frac{c_0}{2}\|d\|_1.$$

Choose a neighborhood small enough that $|r(d)| \leqslant (c_0/4)\|d\|_1$. Then every nonzero feasible d in that neighborhood satisfies

$$\mathcal{H}(v_{\boldsymbol{s}} + d) - \mathcal{H}(v_{\boldsymbol{s}}) \geqslant \frac{c_0}{4}\|d\|_1 > 0.$$

Hence $v_{\boldsymbol{s}}$ is a strict local minimum. □

*Proof of **Theorem 4.1**.* By Lemma A.3, every continuous local minimum is a vertex. First let $\boldsymbol{s} \in \mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$. By Definition 3.1, every gain satisfies $\Delta_i^b(\boldsymbol{s}) \geqslant 0$, and therefore

$$\Delta_i^b(\boldsymbol{s}) + \rho_i \geqslant \rho_i > 0.$$

The vertex criterion makes $v_{\boldsymbol{s}}$ a strict local minimum of $\mathcal{H}$.

Now let $\boldsymbol{s} \notin \mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$. Then at least one flip has $\Delta_i^b(\boldsymbol{s}) < 0$. By the definition of $\gamma(\mathcal{E})$,

$$-\Delta_i^b(\boldsymbol{s}) \geqslant \gamma(\mathcal{E})$$

for every improving flip, equivalently $\Delta_i^b(\boldsymbol{s}) \leqslant -\gamma(\mathcal{E})$. Since $\rho_i \leqslant \gamma(\mathcal{E})$,

$$\Delta_i^b(\boldsymbol{s}) + \rho_i \leqslant 0$$

for such an improving flip. By Lemma A.4, $v_{\boldsymbol{s}}$ is not a local minimum. Thus the continuous local minima are exactly the embedded one-flip local minima, and all of them are strict. □

*Proof of **Theorem 4.2**.* The retention of one-flip local minima is automatic for every choice $\rho_i > 0$: if $\boldsymbol{s}$ is a one-flip local minimum, then $\Delta_i^b(\boldsymbol{s}) + \rho_i > 0$ for every incident edge. Hence only vertices corresponding to states that are not one-flip local minima need to be excluded.

For arbitrary positive slopes (ρ_i), Lemma A.4 shows that a nonminimum vertex $v_{\boldsymbol{s}}$ is excluded exactly when at least one incident edge satisfies

$$\Delta_i^b(\boldsymbol{s}) + \rho_i \leqslant 0.$$

Together with corner localization, this proves the stated vector-slope condition.

Now suppose that all blocks use the same slope $\rho > 0$. For each $\boldsymbol{s} \notin \mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$ define

$$M(\boldsymbol{s}) := \max_{\substack{i\in[n]\\ b\neq s_i}} \{-\Delta_i^b(\boldsymbol{s})\}.$$

Because $\boldsymbol{s}$ is not a one-flip local minimum, it has at least one improving flip, so $M(\boldsymbol{s}) > 0$. The vertex $v_{\boldsymbol{s}}$ is excluded if and only if some flip obeys $-\Delta_i^b(\boldsymbol{s}) \geqslant \rho$, which is equivalent to $M(\boldsymbol{s}) \geqslant \rho$. Therefore all nonminimum vertices are excluded if and only if

$$\rho \leqslant \min_{\boldsymbol{s}\notin \mathrm{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})} M(\boldsymbol{s}) = \Gamma(\mathcal{E}).$$

If the set of nonminima is empty, the outer minimum is $+\infty$ by convention and the conclusion is immediate. Otherwise the state space is finite, so a state $\boldsymbol{s}_\star$ attaining the minimum exists. If $\rho > \Gamma(\mathcal{E})$, then

$$-\Delta_i^b(\boldsymbol{s}_\star) < \rho$$

for every incident flip, hence every edge derivative at $v_{\boldsymbol{s}_\star}$ is strictly positive. By Lemma A.4, $v_{\boldsymbol{s}_\star}$ becomes a spurious strict continuous local minimum. At the endpoint $\rho = \Gamma(\mathcal{E})$, at least one incident derivative is zero at every minimizing nonminimum state, and the strict-concavity argument in Lemma A.4 still gives descent. Thus the sharp common-slope range is exactly $0 < \rho \leqslant \Gamma(\mathcal{E})$. □

*Proof of **Proposition 4.3**.* If every state in $\mathbb{H}_{\boldsymbol{m}}$ is a one-flip local minimum, then no negative gain exists. Both defining minima are over empty sets, so

$$\gamma(\mathcal{E}) = \Gamma(\mathcal{E}) = +\infty.$$

Assume now that at least one nonminimum state exists. Define the set of improving flips at $\boldsymbol{s}$ by

$$\mathcal{I}(\boldsymbol{s}) := \{(i, b) : b \neq s_i,\ \Delta_i^b(\boldsymbol{s}) < 0\}.$$

For every $\boldsymbol{s} \notin \operatorname{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})$, the set $\mathcal{I}(\boldsymbol{s})$ is nonempty. Since $\gamma(\mathcal{E})$ is the minimum improvement magnitude over all states and improving flips,

$$\gamma(\mathcal{E}) \leqslant -\Delta_i^b(\boldsymbol{s}) \qquad \text{for every } (i, b) \in \mathcal{I}(\boldsymbol{s}).$$

Hence, for every nonminimum $\boldsymbol{s}$,

$$\gamma(\mathcal{E}) \leqslant \max_{(i,b)\in\mathcal{I}(\boldsymbol{s})} \{-\Delta_i^b(\boldsymbol{s})\} = \max_{\substack{i\in[n]\\ b\neq s_i}} \{-\Delta_i^b(\boldsymbol{s})\}.$$

The equality holds because at a nonminimum state at least one value $-\Delta_i^b(\boldsymbol{s})$ is positive, whereas nonimproving flips contribute only nonpositive values. Taking the minimum over all nonminimum states gives

$$\gamma(\mathcal{E}) \leqslant \min_{\boldsymbol{s}\notin \operatorname{loc}_{\mathbb{H}_{\boldsymbol{m}}}(\mathcal{E})} \max_{\substack{i\in[n]\\ b\neq s_i}} \{-\Delta_i^b(\boldsymbol{s})\} = \Gamma(\mathcal{E}).$$

□

Corollary A.5 (Global minima)**.** *For every collection of admissible attractors with positive slopes, even when the slopes exceed the local-equivalence threshold,* (13) *holds.*

Proof. The product simplex $\mathbb{Q}_{\boldsymbol{m}}$ is compact and $\mathcal{H}$ is continuous, so $\mathcal{H}$ has a global minimizer. Every global minimizer is, in particular, a local minimizer relative to $\mathbb{Q}_{\boldsymbol{m}}$; by Lemma A.3, it must therefore be a vertex $v_{\boldsymbol{s}}$. At every vertex all block attractors vanish, so

$$\mathcal{H}(v_{\boldsymbol{s}}) = \mathcal{E}_{\mathbb{Q}}(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s}).$$

Consequently, the minimum value of $\mathcal{H}$ over $\mathbb{Q}_{\boldsymbol{m}}$ equals the minimum value of $\mathcal{E}$ over $\mathbb{H}_{\boldsymbol{m}}$, and a vertex minimizes $\mathcal{H}$ exactly when its state minimizes $\mathcal{E}$. This proves both inclusions in (13). □

Corollary A.6 (Lattice certificate)**.** *Suppose $\delta > 0$ and every factor-table value belongs to $\delta\mathbb{Z}$. Then every nonzero gain belongs to $\delta\mathbb{Z}$, and any slopes satisfying $0 < \rho_i \leqslant \delta$ obey the sufficient condition of Theorem 4.1.*

More generally, when all factor-table values are rational, one may clear denominators and obtain a variable-specific lattice spacing from the greatest common divisor of the incident factor differences generated by changing that variable. Any positive slope no larger than this spacing is safe for that variable. If all such differences vanish, the objective is independent of that variable and that block imposes no upper slope restriction.

Proof. For a flip of variable i from $a = s_i$ to b, all factors not containing i cancel, and

$$\Delta_i^b(\boldsymbol{s}) = \sum_{f\ni i} \left[\psi_f\Big(\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big)_f\Big) - \psi_f(\boldsymbol{s}_f)\right].$$

If every factor-table value lies in $\delta\mathbb{Z}$, every bracketed difference, and hence every gain, lies in $\delta\mathbb{Z}$. Thus any negative gain is at most $-\delta$, so either no improving flip exists or $\gamma(\mathcal{E}) \geqslant \delta$. The first claim follows from Theorem 4.1.

For the variable-specific statement, let q be a common positive integer such that every rescaled factor-table value $\widehat{\psi}_f := q\psi_f$ is integral. For each variable i, define

$$g_i := \gcd\Big\{\widehat{\psi}_f(b, a_{f\setminus\{i\}}) - \widehat{\psi}_f(a, a_{f\setminus\{i\}}) : f \ni i\Big\},$$

where the set ranges over all $a, b \in [m_i]$ and all assignments $a_{f\setminus\{i\}}$ to the remaining variables in f, with $g_i = 0$ when all such differences vanish. If $g_i > 0$, put $\delta_i = g_i/q$.

Every gain obtained by changing variable i is an integer linear combination of these factor differences and is therefore a multiple of δ_i. Hence every nonzero such gain has magnitude at least δ_i. Choosing

$$0 < \rho_i \leqslant \delta_i$$

for each i with $g_i > 0$ ensures that every strict improving flip of that variable retains a nonpositive continuous edge derivative. If $g_i = 0$, all incident factor tables are independent of the state of variable i, so all its gains vanish and no upper slope restriction is needed for that block. The vector-slope condition in Theorem 4.2 therefore gives landscape equivalence. □

B Constructing admissible attractors

Normalized-curvature family. Choose a continuous $k_i : [0,1] \to (0,\infty)$ with $\int_0^1 k_i(u)\,du = 1$ and define

$$\phi_i(t) = \rho_i \left[t \int_0^1 (1-u)k_i(u)\,du - \int_0^t (t-u)k_i(u)\,du \right]. \tag{17}$$

Set $\alpha_i(p_i) = \sum_a \phi_i(p_{i,a})$.

Theorem B.1 (Normalized-curvature construction)**.** *Let $k_i : [0,1] \to (0,\infty)$ be continuous and satisfy $\int_0^1 k_i(u)\,du = 1$. Define ϕ_i by* (17) *and $\alpha_i(p_i) = \sum_{a=1}^{m_i} \phi_i(p_{i,a})$. Then α_i is an admissible block attractor of inward slope ρ_i.*

Proof. Because k_i is continuous and strictly positive on the compact interval $[0,1]$, it has a positive minimum. It can therefore be extended continuously and positively to a small open interval containing $[0,1]$ (for example, by keeping it constant just beyond the two endpoints). We use this extension only to satisfy the neighborhood requirement in the definition of admissibility; the values on the simplex remain unchanged.

Let

$$A_i := \int_0^1 (1-u)k_i(u)\,du.$$

Then (17) can be written

$$\phi_i(t) = \rho_i \left[tA_i - \int_0^t (t-u)k_i(u)\,du \right].$$

Directly, $\phi_i(0) = 0$, and

$$\phi_i(1) = \rho_i \left[A_i - \int_0^1 (1-u)k_i(u)\,du \right] = 0.$$

Leibniz' rule gives

$$\phi_i'(t) = \rho_i \left[A_i - \int_0^t k_i(u)\,du \right], \qquad \phi_i''(t) = -\rho_i k_i(t) < 0.$$

Hence ϕ_i is strictly concave on a neighborhood of $[0,1]$, and the separable sum α_i is strictly concave on a neighborhood of the simplex. Also,

$$\phi_i'(0) - \phi_i'(1) = \rho_i \int_0^1 k_i(u)\,du = \rho_i.$$

At every simplex vertex $\boldsymbol{e}_a$, all coordinates are 0 or 1, so $\alpha_i(\boldsymbol{e}_a) = 0$. Finally, along the edge $(1-t)\boldsymbol{e}_a + t\boldsymbol{e}_b$,

$$\frac{d}{dt}\alpha_i\big((1-t)\boldsymbol{e}_a + t\boldsymbol{e}_b\big)\Big|_{t=0^+} = -\phi_i'(1) + \phi_i'(0) = \rho_i.$$

Thus all three requirements of an admissible block attractor are satisfied. □

Label-resolved family. For a lattice energy $\mathcal{E} : \mathbb{H}_{\boldsymbol{m}} \to \delta\mathbb{Z}$, write $\mathbb{H}_{-i} = \prod_{j\neq i}[m_j]$. Choose context tables $r_{i,a} : \mathbb{H}_{-i} \to [\varepsilon, \delta]$ with $0 < \varepsilon \leqslant \delta$, and extend them block-affinely to $\mathbb{Q}_{\boldsymbol{m}}$ by

$$\rho_{i,a}(p_{-i}) = \sum_{\boldsymbol{s}_{-i}\in\mathbb{H}_{-i}} r_{i,a}(\boldsymbol{s}_{-i}) \prod_{j\neq i} p_{j,s_j}.$$

Define

$$\mathcal{A}_{\mathrm{LR}}(p) = \frac{1}{2}\sum_{i=1}^{n}\sum_{a=1}^{m_i} \rho_{i,a}(p_{-i})\, p_{i,a}(1-p_{i,a}).$$

This broader construction is theoretical; in LEMON we use the quadratic specialization (14).

Theorem B.2 (Label-resolved landscape equivalence)**.** *Under the label-resolved construction above, the Hamiltonian*

$$\mathcal{H}_{\mathrm{LR}}(p) = \mathcal{E}_{\mathbb{Q}}(p) + \mathcal{A}_{\mathrm{LR}}(p)$$

has exactly the embedded one-flip local minima of $\mathcal{E}$ as its local minima, all strict. Its global minimizers are exactly the embedded global minimizers of $\mathcal{E}$.

Proof. Fix all blocks except p_i. The extension $\mathcal{E}_{\mathbb{Q}}$ is affine in p_i. In the ith attractor summand, every $\rho_{i,a}(p_{-i})$ is constant with respect to p_i, and the Hessian in p_i is diagonal with entries $-\rho_{i,a}(p_{-i}) \leqslant -\varepsilon$. For $j \neq i$, the factors $p_{j,a}(1-p_{j,a})$ are fixed, while $\rho_{j,a}(p_{-j})$ is affine in p_i. Hence $\mathcal{H}_{\mathrm{LR}}$ is strictly concave in each block, and the same corner-localization argument as in Lemma A.3 shows that every local minimum is a vertex.

Fix $\boldsymbol{s} \in \mathbb{H}_{\boldsymbol{m}}$ and vary variable i from $a = s_i$ to $b \neq s_i$. Along this edge, all attractor terms associated with blocks $j \neq i$ vanish because those blocks remain at simplex vertices. Let

$$\bar{r}_{ib}(\boldsymbol{s}) := \frac{r_{i,s_i}(\boldsymbol{s}_{-i}) + r_{i,b}(\boldsymbol{s}_{-i})}{2}.$$

Then

$$\mathcal{H}_{\mathrm{LR}}\big(v_{\boldsymbol{s}} + t(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i})\big) - \mathcal{H}_{\mathrm{LR}}(v_{\boldsymbol{s}}) = t\Delta_i^b(\boldsymbol{s}) + \bar{r}_{ib}(\boldsymbol{s})\, t(1-t).$$

If $\Delta_i^b(\boldsymbol{s}) \geqslant 0$, the inward derivative is

$$\Delta_i^b(\boldsymbol{s}) + \bar{r}_{ib}(\boldsymbol{s}) \geqslant \varepsilon > 0.$$

Thus every embedded one-flip local minimum has strictly positive incident edge derivatives and is a strict continuous local minimum.

If $\Delta_i^b(\boldsymbol{s}) < 0$, the lattice assumption gives $\Delta_i^b(\boldsymbol{s}) \leqslant -\delta$, while $\bar{r}_{ib}(\boldsymbol{s}) \leqslant \delta$. If $\bar{r}_{ib}(\boldsymbol{s}) < \delta$, the inward derivative is negative. In the endpoint case $\Delta_i^b(\boldsymbol{s}) = -\delta$ and $\bar{r}_{ib}(\boldsymbol{s}) = \delta$,

$$\mathcal{H}_{\mathrm{LR}}\big(v_{\boldsymbol{s}} + t(\boldsymbol{e}_{ib} - \boldsymbol{e}_{is_i})\big) - \mathcal{H}_{\mathrm{LR}}(v_{\boldsymbol{s}}) = -\delta t^2 < 0 \qquad (t > 0).$$

Hence every state admitting an improving flip is excluded from the continuous local minima. Together with corner localization and the same edge-cone argument as in Lemma A.4, this proves the local statement and strictness.

Finally, $\mathbb{Q}_{\boldsymbol{m}}$ is compact, $\mathcal{H}_{\mathrm{LR}}$ is continuous, and every global minimizer is therefore a local minimizer and hence a vertex. At every vertex, $\mathcal{A}_{\mathrm{LR}}(v_{\boldsymbol{s}}) = 0$ and $\mathcal{E}_{\mathbb{Q}}(v_{\boldsymbol{s}}) = \mathcal{E}(\boldsymbol{s})$. Thus the global minimizers of $\mathcal{H}_{\mathrm{LR}}$ are exactly the embedded global minimizers of $\mathcal{E}$. □

C PROBLEM-SPECIFIC CERTIFICATES

Proposition C.1. *The one-flip local minima of* (4) *are exactly the maximal independent sets. Each such set is a maximal clique in $\overline{G}$, and its complement is an inclusion-minimal vertex cover of G. For admissible attractors with common slope $0 < \rho \leqslant 1$, the embedded maximal independent sets are exactly the local minima of the relaxed Hamiltonian, and all these continuous minima are strict.*

Proof. Let $I = \{i : z_i = 1\}$ be the selected vertex set and, as in the main text, let $r_i = \sum_{j\in N(i)} z_j$ be the number of selected neighbors of vertex i.

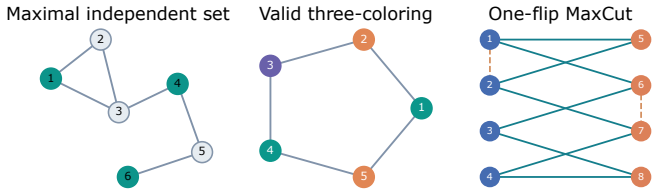


Figure 2: *Native one-flip neighborhoods.* Left: maximal independent set and its inclusion-minimal complementary cover. Middle: proper three-coloring. Right: one-flip max-cut partition with variables drawn on opposing sides.

Suppose first that z is a one-flip local minimum of (4). If $i \in I$, changing z_i from 1 to 0 has gain

$$\Delta_i^{1\to 0} = 1 - 2r_i.$$

If $r_i \geqslant 1$, this gain is negative, contradicting local minimality. Hence every selected vertex has $r_i = 0$, so I is independent. If $i \notin I$ and $r_i = 0$, then

$$\Delta_i^{0\to 1} = -1 + 2r_i = -1 < 0,$$

again contradicting local minimality. Therefore every unselected vertex has a neighbor in I, so the independent set I is maximal.

Conversely, suppose that I is maximal and independent. For $i \in I$, independence gives $r_i = 0$ and therefore $\Delta_i^{1\to 0} = 1 > 0$. For $i \notin I$, maximality gives $r_i \geqslant 1$ and hence $\Delta_i^{0\to 1} = -1 + 2r_i \geqslant 1$. Thus every one-flip gain is strictly positive: z is a strict one-flip local minimum. Because the objective is integer-valued, its nonzero gains have magnitude at least one. Therefore, for a common admissible slope $0 < \rho \leqslant 1$, Theorem 4.1 makes exactly these embedded states strict local minima of the relaxed Hamiltonian.

The graph transformations are witness identities. A set I is independent in G if and only if it is a clique in the complement graph $\overline{G}$, and maximality is preserved under this equivalence. Also, $C \subseteq V$ is a vertex cover of G if and only if $V \setminus C$ is independent. If $I = V \setminus C$ is maximal independent, then every $v \in C$ has a neighbor in I; removing v from C would leave that edge uncovered, so C is inclusion-minimal. Conversely, if C is inclusion-minimal and $I = V \setminus C$ were not maximal independent, some $v \in C$ could be added to I, which means v has no neighbor in I; then removing v from C would still cover every edge, a contradiction. Hence the three statements describe the same verified witness under the standard graph transformations. □

Proposition C.2. *If G is k-colorable, the global minima of the relaxed Hamiltonian associated with* (3) *are exactly the embedded proper k-colorings. For every positive admissible attractor slope, each embedded proper coloring is a strict continuous local minimum. In addition, for a common slope $0 < \rho \leqslant 1$, the entire continuous local-minimum set is exactly the embedded one-flip local-minimum set of the integer conflict energy.*

Proof. The conflict energy $\mathcal{E}_{\mathrm{col},k}$ is a nonnegative integer and equals zero exactly for proper k-colorings. By Corollary A.5, the global minimizers of the relaxed Hamiltonian are therefore exactly the embedded proper colorings, independently of the size of the positive admissible slope.

Now let $\boldsymbol{s}$ be a proper coloring and suppose vertex i currently has color $a = s_i$. Properness gives $d_i(a) = 0$. Recoloring i to any $b \neq a$ has gain

$$\Delta_i^b(\boldsymbol{s}) = d_i(b) - d_i(a) = d_i(b) \geqslant 0.$$

Hence every inward edge derivative at $v_{\boldsymbol{s}}$ equals $\Delta_i^b(\boldsymbol{s}) + \rho_i > 0$, and Lemma A.4 makes $v_{\boldsymbol{s}}$ a strict local minimum for every positive admissible slope. Finally, because the conflict energy is integer-valued, every nonzero gain has magnitude at least one; the full local-landscape correspondence for $0 < \rho \leqslant 1$ follows from Theorem 4.1 (or Corollary A.6). □

Proposition C.3. *For the equality energy $\mathcal{E}_{k\text{-cut}}$ in* (2)*, a labeling $\boldsymbol{s}$ is a one-flip local minimum if and only if*

$$\sum_{j:s_j=b} w_{ij} - \sum_{j:s_j=s_i} w_{ij} \geqslant 0 \qquad \textit{for every } i \in [n] \textit{ and } b \neq s_i.$$

If the edge weights are integers, every nonzero gain is an integer, so any common theorem slope $0 < \rho \leqslant 1$ is safe. Moreover, if

$$W := \sum_{\{i,j\}\in E} w_{ij}$$

is attained by a cut, that cut is globally optimal.

Proof. Fix a vertex i with current label $a = s_i$ and reassign it to $b \neq a$. Only incident edges can change their equality status. An edge from i to a neighbor currently in class a ceases to contribute its weight to the equality energy, while an edge from i to a neighbor in class b begins to contribute. All other incident edges are unchanged. Therefore

$$\mathcal{E}_{k\text{-cut}}\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big) - \mathcal{E}_{k\text{-cut}}(\boldsymbol{s}) = \sum_{j:s_j=b} w_{ij} - \sum_{j:s_j=a} w_{ij},$$

which is (8). By Definition 3.1, the labeling is a one-flip local minimum exactly when every such gain is nonnegative.

If all w_{ij} are integers, each gain is an integer sum and hence every nonzero gain has magnitude at least one; the lattice certificate gives the stated slope range. Finally, because the manuscript assumes $w_{ij} \geqslant 0$, every edge contributes at most its weight to the cut, so

$$\mathcal{C}(\boldsymbol{s}) \leqslant W.$$

A witness attaining W therefore attains a valid global upper bound and is globally optimal. □

Proposition C.4. *Let*

$$\mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) = D(\boldsymbol{s}) + \Lambda N_0(\boldsymbol{s}), \qquad \Lambda > B := \sum_{i=1}^{n} |a_i|,$$

with $n \geqslant k$. A configuration $\boldsymbol{s} \in [k]^n$ is a one-flip local minimum of $\mathcal{E}_{k\text{-NPP}}$ if and only if every group is nonempty and

$$D\big(\boldsymbol{s} + (q - s_i)\boldsymbol{e}_i\big) - D(\boldsymbol{s}) \geqslant 0 \tag{18}$$

for every $i \in [n]$ and $q \in [k] \setminus \{s_i\}$ such that the source group contains at least two items. Moreover, the global minimizers of $\mathcal{E}_{k\text{-NPP}}$ are exactly the globally optimal nonempty k-way partitions under MSD.

Proof. For every assignment,

$$0 \leqslant D(\boldsymbol{s}) \leqslant B.$$

If $\boldsymbol{s}$ contains an empty group, then because $n \geqslant k$, some occupied group contains at least two items. Moving one of these items into an empty group decreases N_0 by one without creating a new empty group. Since the MSD can increase by at most B,

$$\mathcal{E}_{k\text{-NPP}}\big(\boldsymbol{s} + (q - s_i)\boldsymbol{e}_i\big) - \mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) \leqslant B - \Lambda < 0.$$

Hence no configuration containing an empty group is a one-flip local minimum.

Now suppose every group is nonempty. If the source group of a move contains at least two items, both assignments are valid and the penalty remains zero. Therefore

$$\mathcal{E}_{k\text{-NPP}}\big(\boldsymbol{s} + (q - s_i)\boldsymbol{e}_i\big) - \mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) = D\big(\boldsymbol{s} + (q - s_i)\boldsymbol{e}_i\big) - D(\boldsymbol{s}).$$

If the source group is a singleton, the move creates an empty group. Since the MSD can decrease by at most B, its penalized energy change is at least

$$\Lambda - B > 0.$$

Such a move therefore cannot be improving. This proves the one-flip local-minimum characterization.

Finally, every invalid configuration has $\mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) \geqslant \Lambda > B$, whereas every valid configuration has $\mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) = D(\boldsymbol{s}) \leqslant B$. Hence every global minimizer is valid, and minimizing $\mathcal{E}_{k\text{-NPP}}$ over $[k]^n$ is equivalent to minimizing MSD over nonempty k-way partitions. □

Corollary C.5 (k-way NPP certificate)**.** *Assume $a_i > 0$ for every i. If every group is nonempty and*

$$\min_i a_i > D(\boldsymbol{s}),$$

then every feasible one-flip move strictly increases the MSD; hence $\boldsymbol{s}$ is a one-flip local minimum.

Proof. Let i be moved from its current group $p = s_i$ to another group q, and write $a = a_i$. Since $D(\boldsymbol{s}) < a$, every pair of current group sums differs by less than a. In particular,

$$S_q + a > \max_r S_r(\boldsymbol{s}), \qquad S_p - a < \min_r S_r(\boldsymbol{s}).$$

After the move, the destination is therefore above the previous maximum and the source is below the previous minimum. Consequently,

$$\begin{aligned} D\big(\boldsymbol{s} + (q - s_i)\boldsymbol{e}_i\big) &\geqslant (S_q + a) - (S_p - a) \\ &= 2a - (S_p - S_q) > a > D(\boldsymbol{s}). \end{aligned}$$

Thus every feasible one-flip move strictly increases the MSD. □

D Algorithmic identities and implementation

For an assignment $\boldsymbol{s}$, a coordinate $i \in [n]$, and a candidate state $b \in [m_i]$, define the incident conditional field

$$F_i(b; \boldsymbol{s}_{-i}) := \sum_{\substack{f \in \mathcal{F} \\ i \in f}} \psi_f\big(b, \boldsymbol{s}_{f \setminus \{i\}}\big),$$

where $\boldsymbol{s}_{-i}$ denotes all coordinates except i. The arguments of each interaction function follow the fixed variable order for that factor. Factors not containing i are unchanged when s_i is replaced by b and therefore cancel in the energy difference. Hence

$$\Delta_i^b(\boldsymbol{s}) = F_i(b; \boldsymbol{s}_{-i}) - F_i(s_i; \boldsymbol{s}_{-i}).$$

Exact energy changes can therefore be evaluated from incident factors alone. Problem-specific quantities, such as color-neighbour counts, edge weights to each group, and maintained subset sums, make these evaluations efficient.

Problem-specific objectives and gradients. For MIS, let $p_{i,1}$ denote the probability that vertex i is selected and $p_{i,0}$ the probability that it is excluded. The relaxed energy and Hamiltonian are

$$\begin{aligned} \mathcal{E}_{\mathbb{Q}_{\mathrm{MIS}}}(p) &= -\sum_i p_{i,1} + 2 \sum_{\{i,j\} \in E} p_{i,1} p_{j,1}, \\ \mathcal{H}_{\mathrm{MIS}}(p) &= \mathcal{E}_{\mathbb{Q}_{\mathrm{MIS}}}(p) + \mathcal{A}_\eta(p). \end{aligned}$$

Its gradient components are

$$\frac{\partial \mathcal{H}_{\mathrm{MIS}}}{\partial p_{i,0}} = -2\eta p_{i,0}, \qquad \frac{\partial \mathcal{H}_{\mathrm{MIS}}}{\partial p_{i,1}} = -1 + 2 \sum_{j \in N(i)} p_{j,1} - 2\eta p_{i,1}.$$

For fixed-k coloring,

$$\begin{aligned} \mathcal{E}_{\mathbb{Q}_{\mathrm{col},k}}(p) &= \sum_{\{i,j\} \in E} \sum_{c=1}^{k} p_{i,c} p_{j,c}, \\ \mathcal{H}_{\mathrm{col},k}(p) &= \mathcal{E}_{\mathbb{Q}_{\mathrm{col},k}}(p) + \mathcal{A}_\eta(p). \end{aligned}$$

Its gradient is

$$\frac{\partial \mathcal{H}_{\mathrm{col},k}}{\partial p_{i,c}} = \sum_{j \in N(i)} p_{j,c} - 2\eta p_{i,c}.$$

For max-k-cut, edge weights give

$$\mathcal{E}_{\mathbb{Q}_{k\text{-cut}}}(p) = \sum_{\{i,j\}\in E} w_{ij} \sum_{c=1}^{k} p_{i,c} p_{j,c},$$
$$\mathcal{H}_{k\text{-cut}}(p) = \mathcal{E}_{\mathbb{Q}_{k\text{-cut}}}(p) + \mathcal{A}_\eta(p).$$

Its gradient is

$$\frac{\partial \mathcal{H}_{k\text{-cut}}}{\partial p_{i,c}} = \sum_{j\in N(i)} w_{ij} p_{j,c} - 2\eta p_{i,c}.$$

For k-way NPP, the exact extension of (6) is

$$\mathcal{E}_{\mathbb{Q}_{k\text{-NPP}}}(p) = \sum_{\boldsymbol{s}\in[k]^n} \mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) \prod_{j=1}^{n} p_{j,s_j},$$
$$\mathcal{H}_{k\text{-NPP}}(p) = \mathcal{E}_{\mathbb{Q}_{k\text{-NPP}}}(p) + \mathcal{A}_\eta(p). \tag{19}$$

Its exact gradient is

$$\frac{\partial \mathcal{H}_{k\text{-NPP}}}{\partial p_{i,r}} = \sum_{\substack{\boldsymbol{s}\in[k]^n \\ s_i=r}} \mathcal{E}_{k\text{-NPP}}(\boldsymbol{s}) \prod_{j\neq i} p_{j,s_j} - 2\eta p_{i,r}. \tag{20}$$

This is the same product-expectation extension as in (9), with the NPP energy treated as a factor involving all n variables, and is therefore covered by Theorem 4.1 under its attractor-slope condition.

When all blocks other than i are fixed at the embedded states $\boldsymbol{e}_{s_j}$, block affinity gives the exact restriction

$$\mathcal{E}_{\mathbb{Q}_{k\text{-NPP}}}\big(p_i, (\boldsymbol{e}_{s_j})_{j\neq i}\big) = \sum_{r=1}^{k} p_{i,r}\, \mathcal{E}_{k\text{-NPP}}\big(\boldsymbol{s} + (r - s_i)\boldsymbol{e}_i\big). \tag{21}$$

Thus evaluating this block restriction and its gradient requires only the k endpoint energies.

Problem-specific adapter summary. For the table below, let $r_i = \sum_{j\in N(i)} z_j$ denote the number of selected neighbours of vertex i in MIS, $d_i(c) = |\{j \in N(i) : s_j = c\}|$ the number of neighbours of color c, and $n_r(\boldsymbol{s}) = |\{i : s_i = r\}|$ the size of NPP group r.

Table 3: *Problem-specific quantities used by* LEMON. The search stage uses the exact extension of each listed energy together with the quadratic attractor. Certificates use exact changes in the original discrete neighborhood.

Problem	Energy	Exact energy change	Verification
Max-k-cut	(2)	$\sum_{j\in N(i)} w_{ij}\big(\mathbb{1}\{s_j = b\} - \mathbb{1}\{s_j = s_i\}\big)$	All labels are feasible; every one-flip change is nonnegative.
Coloring	(3)	$d_i(b) - d_i(s_i)$	Check all one-flip changes; accept a proper coloring only if $\mathcal{E}_{\mathrm{col},k} = 0$.
MIS	(4)	$2r_i - 1$ when adding vertex i; $1 - 2r_i$ when removing it	Verify independence and nonnegative changes for every flip.
k-way NPP	(6)	$D\big(\boldsymbol{s} + (b - s_i)\boldsymbol{e}_i\big) - D(\boldsymbol{s})$ for moves preserving nonemptiness	Every group is nonempty and every feasible one-flip change is nonnegative.

Projected and block-coordinate search. For MIS, coloring, and max-k-cut, LEMON maintains one simplex block per variable, evaluates the exact gradients above, applies ADAM, and projects every updated block by Euclidean simplex projection. The initial graph blocks are obtained by normalizing the independent and uniform $\mathcal{U}[0,1]$ samples. For k-way NPP, the code first constructs a longest-processing-time (LPT) partition and embeds its simplex vertices, then it applies projected ADAM updates using (21) (as the other blocks remain at vertices).

Search over the number of colors. Let ϑ be the maximum degree of the graph. We initialize the search using the bound $\chi(G) \leqslant \vartheta + 1$ and use a halving and bisection schedule over candidate values of k. This schedule requires $O(\log(\vartheta + 1))$ fixed-k LEMON calls. Each candidate is accepted only after an independent zero-conflict check, and we report the smallest successful coloring encountered.

Algorithm 1. LEMON (Landscape Equivalent Multi-State Optimization)

Inputs. State sets $[m_i]$, discrete objective $\mathcal{E}$, exact one-flip energy changes $\Delta_i^b(\boldsymbol{s})$, a feasibility predicate, search gradient $g(p) = \nabla\mathcal{H}(p)$, and iteration budget T.

Outputs. A feasible assignment, its discrete objective value, and a one-flip local optimality certificate.

1. **Initialize.** Choose a random point $p^{(0)} \in \mathbb{Q}_{\boldsymbol{m}}$ and initialize the ADAM moments.
2. **Run projected ADAM.** For $t = 0, \ldots, T-1$, evaluate $g^{(t)} = g(p^{(t)})$ and take an ADAM step to obtain $y^{(t+1)}$. Project each block onto its simplex,
$$p_i^{(t+1)} = \Pi_{\Delta_{m_i-1}}\big(y_i^{(t+1)}\big), \qquad i \in [n].$$
3. **Decode and verify.** Choose
$$\hat{s}_i \in \arg\max_{a\in[m_i]} p_{i,a}^{(T)}.$$
Then evaluate $\mathcal{E}(\hat{\boldsymbol{s}})$ and check feasibility and the exact one-flip energy changes. If the assignment is feasible and no flip decreases the energy, return it with its certificate.
4. **Correct if needed.** Apply improving or feasibility-restoring discrete updates. Recompute the objective and repeat the checks until feasibility and one-flip local optimality are verified.

Figure 3: *Algorithmic description of the* LEMON *Solver.* LEMON combines continuous search with exact discrete verification. Discrete corrections are applied only when the decoded assignment fails the feasibility or local optimality check.

A failed heuristic run does not establish non-k-colorability, so the schedule does not certify that the reported number of colors is minimal.

Verification and correction. After decoding, the solver checks feasibility and the exact one-flip energy changes. If these checks certify the assignment, no correction is applied. When verification is combined with descent, the routine scans for a negative energy change. Finding none completes the local optimality check, while finding one triggers an improving move. For feasible assignments, the correction routine accepts strictly improving feasible flips and repeats the checks. Any required feasibility restoration is performed before returning a certified solution. For the exact landscape-equivalent objectives, Theorem 4.1 ensures that the continuous and discrete formulations have corresponding local minima. The theorem does not imply finite-step convergence of the numerical search.

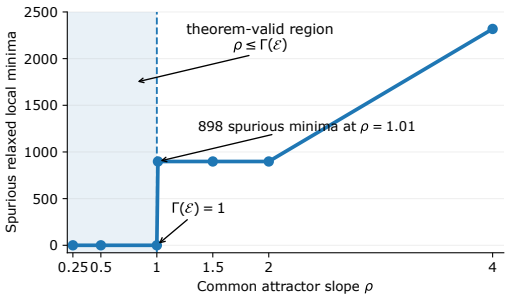
Optional discrete corrections. The implementation also includes optional discrete corrections after the primary search. MIS can use exchange search and maximal completion. Coloring can use a fixed-k conflict search. For k-way NPP, exact-MSD $1 \leftrightarrow 1$ exchanges are followed by exact one-flip (item) descent. max-k-cut uses one-vertex improvements followed by an exhaustive check of all one-vertex moves. After any correction, all reported solutions undergo the corresponding feasibility and exact local optimality checks.

E ABLATION STUDY

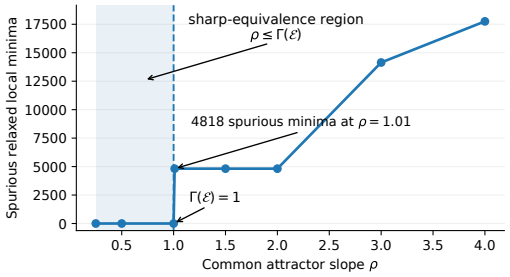
We isolate three aspects of the continuous stage: the sharp common-slope threshold, preservation of the multi-state geometry relative to a one-hot binary reduction, and whether projected ADAM improves the final discrete objective when the subsequent discrete correction moves are held fixed. We use the ADAM settings of Section 6.

Sharp slope threshold. We test the sharp threshold by exact enumeration on four small instances spanning all problem classes. For MIS, a unit-weight $\mathrm{ER}(12, 0.25)$ graph gives $\gamma(\mathcal{E}) = \Gamma(\mathcal{E}) = 1$ and 20 discrete local minima; the relaxed landscape agrees exactly for every tested $\rho \leqslant 1$, while $\rho = 1.01$ creates 898 spurious minima. The same transition appears for 3-coloring on an $\mathrm{ER}(9, 0.40)$ graph, where $\gamma(\mathcal{E}) = \Gamma(\mathcal{E}) = 1$ and crossing to $\rho = 1.01$ creates 4818 spurious minima. More significantly, max-4-cut on an integer-weight $\mathrm{ER}(8, 0.45)$ graph has $\gamma(\mathcal{E}) = 1 < \Gamma(\mathcal{E}) = 2$: equivalence persists throughout $\rho \leqslant 2$, but 6480 spurious minima appear at $\rho = 2.01$. For 3-way

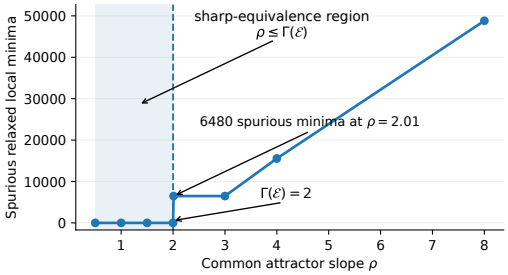
NPP under the exact-MSD lift, the separation is larger, with $\gamma(\mathcal{E}) = 1 < \Gamma(\mathcal{E}) = 4$ and 120 spurious minima appearing immediately at $\rho = 4.01$. Thus, across all four problems, the empirical transition occurs precisely at the sharp boundary predicted by Theorem 4.2; the latter two cases additionally demonstrate that the sharp range can be strictly larger than the sufficient $\gamma(\mathcal{E})$ range.



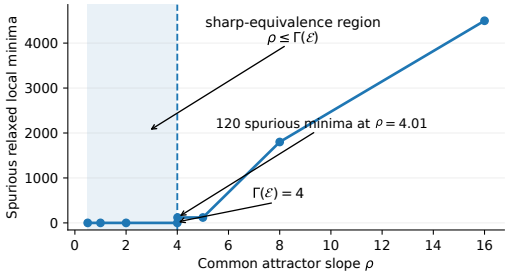
(a) MIS: $\Gamma(\mathcal{E}) = 1$, 898 spurious at $\rho = 1.01$.



(b) 3-coloring: $\Gamma(\mathcal{E}) = 1$, 4818 spurious at $\rho = 1.01$.



(c) Max-4-cut: $\Gamma(\mathcal{E}) = 2$, 6480 spurious at $\rho = 2.01$.



(d) 3-way NPP: $\Gamma(\mathcal{E}) = 4$, 120 spurious at $\rho = 4.01$.

Figure 4: *Exact verification of the sharp common-slope threshold across all four problem classes.* Each panel exhaustively enumerates the discrete state space and applies the exact vertex criterion of Lemma A.4. No spurious relaxed local minimum occurs throughout $\rho \leqslant \Gamma(\mathcal{E})$, whereas crossing $\Gamma(\mathcal{E})$ immediately introduces spurious minima. In (c) and (d), $\Gamma(\mathcal{E}) > \gamma(\mathcal{E}) = 1$, directly illustrating that the sharp equivalence range can be strictly larger than the sufficient range. Panel (d) uses the exact-MSD lift for NPP in (19).

Multi-state versus one-hot binary geometry. We compare the product-simplex relaxation with a value-preserving one-hot binary reduction on `myciel6` ($n = 95$, $k = 7$) and a unit-weight $\mathrm{ER}(100, 0.30)$ max-4-cut graph. LEMON uses $\eta = 0.25$ ($\rho = 0.5$). The binary control represents each k-state variable by k bits with

$$\lambda\left(1 - \sum_{c=1}^{k} z_{ic} + 2\sum_{c<d} z_{ic}z_{id}\right), \qquad \lambda = 1 + \max_i d_i,$$

so feasible one-hot states preserve the discrete objective, whereas a native label reassignment generally requires multiple bit flips.

On `myciel6`, LEMON requires no correction at $T \in \{500, 1000\}$ and all $10/10$ decoded states are already certified, whereas the one-hot control still requires 40.8 ± 2.5 recolorings. For max-4-cut, increasing T from 100 to 1000 reduces the LEMON correction count from 49.6 ± 4.2 to 3.3 ± 2.1, while its decoded cut increases from 1054.6 ± 18.5 to 1239.8 ± 5.8.

Projected ADAM improves solution quality. We isolate the effect of projected ADAM by comparing the full LEMON pipeline with the same initialization followed directly by the same discrete correction moves. Table 5 report the average gain over 100 paired runs for each problem.

Projected ADAM improves all four objectives on average: MIS increases from 26.03 to 49.80 ($+91.32\%$); 4-coloring conflicts reduce from 5.78 to just 0.40 (-93.08%); max-5-cut increases from 895.82 to 1372.40 ($+53.2\%$) on ER-graph ensemble; and three-way NPP MSD decreases from 2.937×10^{-4} to 4.082×10^{-5} (-86.1%) on numbers sampled uniformly from $[0, 1]$. These gains

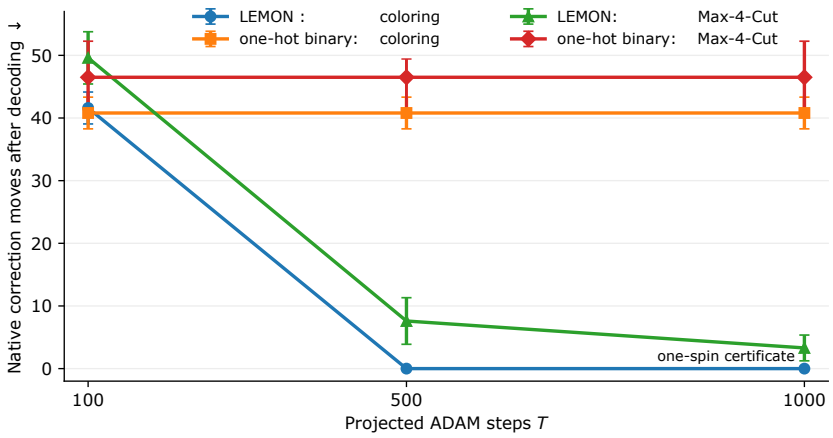


Figure 5: *One-flip certificate proximity versus projected budget.* Mean correction moves after decoding ($\pm$ one standard deviation over ten paired seeds). The simplex relaxation approaches the native certificate as T increases, while the one-hot binary control remains substantially farther away despite reaching one-hot feasibility.

Table 4: *Multi-state versus one-hot binary relaxation at* $T = 1000$. Mean $\pm$ standard deviation over ten paired seeds. "Certificate" is the fraction already satisfying the exact certificate, "Corrections" is the subsequent correction count, and "Final objective" is the objective value after correction steps.

Problem	Relaxation	Decoded	Certificate. (↑)	Correction. (↓)	Final objective
Coloring	One-hot binary	100.1 ± 10.2	0%	40.8 ± 2.5	1.2 ± 1.0
Coloring	LEMON	1.1 ± 0.6	100%	0.0 ± 0.0	1.1 ± 0.6
Max-4-cut	One-hot binary	1071.7 ± 14.0	0%	46.5 ± 5.8	1226.7 ± 6.8
Max-4-cut	LEMON	1239.8 ± 5.8	0%	3.3 ± 2.1	1244.0 ± 5.6

show that projected ADAM does more than reduce the number of subsequent correction moves: it guides the search toward better-quality discrete solutions before the exact correction stage is applied.

Table 5: *Solution-quality gain from projected ADAM.* For each problem, we compare LEMON against the same initialization followed directly by the discrete correction procedure, with projected ADAM omitted. Both variants therefore use identical initialization and post-processing, so the reported difference isolates the contribution of the projected stage. "No ADAM" denotes direct application of the discrete correction moves, while "Projected ADAM" applies the projected ADAM iterations before the same corrections. Entries report mean objective values and the corresponding relative gain, averaged over 100 paired runs.

Problem	Setting	Objective (No ADAM)	Objective (Projected ADAM)	Gain
MIS (↑)	$\mathrm{ER}(100, 0.03)$	26.03	**49.80**	91.32%
Coloring (↓)	$k = 4,\ \mathrm{ER}(100, 0.05)$	5.78	**0.40**	93.08%
Max-k-cut (↑)	$k = 5,\ \mathrm{ER}(100, 0.30)$	895.82	**1372.40**	53.20%
NPP–MSD (↓)	$k = 3,\ n = 100,\ \mathcal{U}[0, 1]$	2.937×10^{-4}	$\mathbf{4.082\times10^{-5}}$	86.10%

Together, these ablations test both the theorem and the optimization mechanism: the sharp slope boundary is recovered exactly, the native multi-state representation approaches the intended one-flip landscape substantially more closely than a one-hot reduction, and projected ADAM steps sufficiently improves the solution quality.

F EXPERIMENTAL PROTOCOL AND BENCHMARKING

All experiments use one 32 GB CPU with an AMD Ryzen 7 9700X 8-Core 64-bit processor. Official solver implementations are used whenever available; otherwise we use the implementations that

preserve the published objective. Projected runs use the random initialization, and ADAM parameters stated in Section 6.

Table 6: *Benchmark protocol used throughout the experiments.*

Study	Evaluation set	Independent verification
Public graphs	exact DIMACS and G-set files used by the corresponding source studies	graph identity, objective, feasibility, and all one-flip flip gains
Random MIS	100 ER and BA graphs for every parameter setting	independence, maximality, and MIS size
Max-k-cut	G-set graphs	k-Cut recomputation and the one-flip certificate
k-way NPP	100 NPP instances per distribution	MSD recomputation and the one-flip certificate

Every reported graph output is reread by a verifier separate from the search routine. The verifier recomputes the objective and the original one-flip changes from the returned assignment. The random-graph and NPP ensemble summaries are computed directly from the per-run result files.

F.1 Graph Coloring Benchmark

The χ column is the chromatic number reported in the coloring study of Arora et al. (2026); '$*$' denotes the chromatic number is unknown for the corresponding graph. The GBS/W-GBS column reports the best between the paper's GBS and W-GBS results.

Table 7: *Full graph-coloring comparison.* Results on 45 DIMACS benchmark instances. Bold denotes the best value, and underlined denotes the 2nd best value. The column χ gives the known chromatic number when available, while '$*$' indicates no known value. LEMON reaches the known χ on $27/30$ instances for which it is available and matches or improves GBS/W-GBS on $43/45$ instances overall, including several strict improvements. The n and $|E|$ columns list the exact numbers of vertices and edges of the graphs, and the runtime is the recorded LEMON solver time.

Instance	n	$\lvert E\rvert$	χ	GBS/W-GBS	LEMON	Runtime (s)
jean	80	254	**10**	**10**	**10**	0.3179
games120	120	638	**9**	**9**	**9**	0.2883
miles250	128	387	**8**	**8**	**8**	0.2900
miles500	128	1170	**20**	**20**	**20**	0.3446
miles750	128	2113	**31**	32	**31**	0.3936
fpsol2.i.3	425	8688	**30**	**30**	**30**	0.7942
fpsol2.i.2	451	8691	**30**	**30**	**30**	0.7705
fpsol2.i.1	496	11654	**65**	**65**	**65**	1.7977
le450_5a	450	5714	**5**	7	**5**	0.3557
le450_5b	450	5734	**5**	8	6	0.3499
le450_25a	450	8260	**25**	27	**25**	0.7037
le450_25b	450	8263	**25**	27	**25**	0.7744
le450_5d	450	9757	**5**	**5**	**5**	0.4495
le450_5c	450	9803	**5**	**5**	**5**	0.5394
homer	561	1628	**13**	**13**	**13**	0.3226
queen5_5	25	160	**5**	**5**	**5**	0.2895
queen6_6	36	290	**7**	9	**7**	0.2688
queen7_7	49	476	**7**	**7**	**7**	0.2994
queen8_8	64	728	**9**	**9**	**9**	0.2814
queen8_12	96	1368	**12**	13	**12**	0.3114
queen9_9	81	1056	**10**	**10**	**10**	0.2689
queen10_10	100	1470	$*$	**11**	**11**	0.3765
queen11_11	121	1980	**11**	13	13	0.3177
queen12_12	144	2596	$*$	**14**	**14**	0.3235
queen13_13	169	3328	**13**	15	15	0.3674
queen14_14	196	4186	$*$	**16**	**16**	0.4261
queen15_15	225	5180	$*$	**17**	**17**	0.4884
queen16_16	256	6320	$*$	**18**	**18**	0.5216
DSJC125.1	125	736	$*$	**5**	**5**	0.2825
DSJC125.5	125	3891	$*$	20	**18**	0.4384
DSJC125.9	125	6961	$*$	46	**44**	0.8784
DSJC250.1	250	3218	$*$	10	**9**	0.3221
DSJC250.5	250	15668	$*$	31	**30**	1.3873
DSJC250.9	250	27897	$*$	76	**74**	4.5778

Table 7 continued

Instance	n	$\lvert E\rvert$	χ	GBS/W-GBS	LEMON	Runtime (s)
DSJC500.5	500	62624	$*$	**53**	54	7.1151
DSJC1000.1	1000	49629	$*$	24	**23**	2.8854
DSJC1000.5	1000	249826	$*$	**94**	99	49.8344
mulsol.i.3	184	3916	**31**	**31**	**31**	0.4977
myciel5	47	236	**6**	**6**	**6**	0.3026
myciel6	95	755	**7**	**7**	**7**	0.2771
myciel7	191	2360	**8**	**8**	**8**	0.3227
zeroin.i.2	211	3541	**30**	**30**	**30**	0.4708
school1	385	19095	$*$	18	**14**	0.9813
anna	138	493	**11**	**11**	**11**	0.2584
flat300_20_0	300	21375	**20**	21	**20**	1.3679

F.2 MIS/CLIQUE BENCHMARK

For the MIS instances, we use the complement of the DIMACS clique graphs, so the independent-set size equals the source-graph clique size (Proposition C.1). pCQO-MIS and ReduMIS values are taken from Alkhouri et al. (2025). Vertex Cover size can be calculated as $n - |\text{MIS}|$ for the MIS graph.

Table 8: *Full MIS/maximum-clique comparison.* Results on 61 DIMACS benchmark instances. Bold denotes the best value, and underlined denotes the 2nd best value. LEMON achives the best results in $53/61$ cases; beats pCQO-MIS in all cases and only differ by 1 from ReduMIS in the remaining 8 cases. The n and $|E|$ columns list the exact numbers of vertices and edges, respectively, and the runtime is the recorded LEMON solver time.

Instance	n	$\lvert E\rvert$	pCQO-MIS	ReduMIS	LEMON	Runtime (s)
C1000.9	1000	49421	65	**67**	66	7.1865
C125.9	125	787	**34**	**34**	**34**	0.7546
C2000.5	2000	999164	15	**16**	**16**	9.6043
C250.9	250	3141	**44**	**44**	**44**	1.4773
C500.9	500	12418	56	**57**	56	3.0517
DSJC1000_5	1000	249674	**15**	**15**	**15**	2.2914
DSJC500_5	500	62126	**13**	**13**	**13**	0.9230
MANN_a9	45	72	**16**	**16**	**16**	0.7338
brock200_1	200	5066	**21**	**21**	**21**	0.8717
brock200_2	200	10024	11	**12**	**12**	0.5251
brock200_3	200	7852	14	**15**	**15**	0.5669
brock200_4	200	6811	16	**17**	**17**	0.6076
brock400_1	400	20077	25	25	**27**	1.3091
brock800_1	800	112095	20	**21**	**21**	2.1440
c-fat200-1	200	18366	**12**	**12**	**12**	0.3531
c-fat200-2	200	16665	**24**	**24**	**24**	0.3928
c-fat200-5	200	11427	**58**	**58**	**58**	0.6297
c-fat500-1	500	120291	**14**	**14**	**14**	1.2778
c-fat500-10	500	78123	**126**	**126**	**126**	4.8999
c-fat500-2	500	115611	**26**	**26**	**26**	1.7365
c-fat500-5	500	101559	**64**	**64**	**64**	2.9693
gen200_p0.9_44	200	1990	42	**44**	**44**	1.1982
gen200_p0.9_55	200	1990	**55**	**55**	**55**	1.0927
gen400_p0.9_55	400	7980	52	**55**	**55**	2.4728
gen400_p0.9_65	400	7980	**65**	**65**	**65**	2.6271
gen400_p0.9_75	400	7980	**75**	**75**	**75**	2.6690
hamming10-2	1024	5120	**512**	**512**	**512**	9.3490
hamming10-4	1024	89600	**40**	**40**	**40**	4.6387
hamming6-2	64	192	**32**	**32**	**32**	0.3537
hamming6-4	64	1312	**4**	**4**	**4**	0.2825
hamming8-2	256	1024	**128**	**128**	**128**	1.0878
hamming8-4	256	11776	**16**	**16**	**16**	0.3996
johnson16-2-4	120	1680	**8**	**8**	**8**	0.3059
johnson32-2-4	496	14880	**16**	**16**	**16**	0.5388
johnson8-2-4	28	168	**4**	**4**	**4**	0.2969
johnson8-4-4	70	560	**14**	**14**	**14**	0.3085
keller4	171	5100	**11**	**11**	**11**	0.9330
p_hat1000-1	1000	377247	**10**	**10**	**10**	2.5412
p_hat1000-2	1000	254701	**46**	**46**	**46**	6.4951
p_hat1000-3	1000	127754	67	**68**	65	5.8801
p_hat1500-1	1500	839327	11	**12**	11	4.8620
p_hat1500-2	1500	555290	**65**	**65**	64	12.0403

Table 8 continued

Instance	n	$\lvert E\rvert$	pCQO-MIS	ReduMIS	LEMON	Runtime (s)
p_hat1500-3	1500	277006	93	**94**	93	12.9821
p_hat300-1	300	33917	**8**	**8**	**8**	0.5313
p_hat300-2	300	22922	**25**	**25**	**25**	1.2727
p_hat300-3	300	11460	**36**	**36**	**36**	1.4136
p_hat500-1	500	93181	**9**	**9**	**9**	0.9484
p_hat500-2	500	61804	**36**	**36**	**36**	2.4419
p_hat500-3	500	30950	**50**	**50**	49	2.4367
p_hat700-1	700	183651	**11**	**11**	**11**	1.3483
p_hat700-2	700	122922	**44**	**44**	**44**	3.7463
p_hat700-3	700	61640	**62**	**62**	61	3.5742
san200_0.7_1	200	5970	**30**	**30**	**30**	1.9335
san200_0.9_1	200	1990	**70**	**70**	**70**	1.2498
san200_0.9_2	200	1990	**60**	**60**	**60**	1.3675
san200_0.9_3	200	1990	**44**	**44**	**44**	1.2860
san400_0.9_1	400	7980	**100**	**100**	**100**	3.1335
sanr200_0.7	200	6032	**18**	**18**	**18**	0.6714
sanr200_0.9	200	2037	**42**	**42**	**42**	1.1429
sanr400_0.5	400	39816	**13**	**13**	**13**	0.6610
sanr400_0.7	400	23931	**21**	**21**	**21**	1.0818

F.3 RANDOM ER AND BA MIS BENCHMARKS

We use two standard random-graph families. ER graphs (Erdős & Rényi, 1959) with n nodes and probability p; and BA graphs (Barabási & Albert, 1999), with n nodes and q edges attached by each new node. We run pCQO-MIS (with 30 seconds runtime limit) from Alkhouri et al. (2025), Gurobi, and LEMON on 100 randomly generated ER graphs for each (n, p) pair and BA graphs for each (n, q) pair, then report the average MIS value.

Table 9: *MIS on Erdős–Rényi (ER) graphs.* Average independent-set size over 100 independently generated $ER(n, p)$ graphs with $n = 3000$, shown across increasing edge densities p. Higher values are better. Across all seven densities, LEMON obtains the largest average MIS size among the compared methods. Runtime is averaged over 100 graphs for LEMON.

p	Gurobi	pCQO-MIS	LEMON	runtime (s)
0.1	56.3	77.2	**79.4**	20.2789
0.2	31.7	38.3	**42.9**	21.1170
0.3	22.2	25.7	**28.9**	18.7755
0.4	15.1	19.2	**21.4**	13.27765
0.5	11.7	14.8	**16.8**	11.26595
0.6	9.5	10.8	**13.1**	11.87632
0.7	6.9	8.7	**10.9**	12.22299

LEMON achieves the largest average MIS across all ER rows while remaining well below the 30 second source time limit used for the pCQO-MIS. This is particularly useful on the denser graphs, where the set of edges grows rapidly and the ILP search becomes more expensive.

LEMON exceeds the best learning-based value on seven BA settings, ties it on one, and trails it only at $(n, q) = (1000, 5)$. It also matches or exceeds ReduMIS (Lamm et al., 2017) on six of the nine settings. The learning-based benchmark uses repeated sampling and substantial training budgets (Wu et al., 2026); LEMON solves each graph directly on the CPU and needs no training set.

F.4 MAX-k-CUT ON G-SET BENCHMARK

"Public best" values are reported from the best max-k-cut values in Tables 18–21 of Shao & Yang (2026) and Table 4 of Banerjee et al. (2026). All benchmark runs for LEMON passed the one-flip certificate. In particular, LEMON attains all 6000 edges of G48 and G49 for $k = 2, 3, 4, 5$, all 6000 edges of G50 for $k = 3, 4, 5$, and all 9999 edges of G70 for $k = 4, 5$. Since these are unit-weight instances, these values certify global optimality.

Table 10: *MIS on Barabási–Albert (BA) graphs.* Average independent-set size over 100 independently generated $BA(n, q)$ graphs with different n, q parameters. "Best learning" denotes the strongest value reported among LwD, LTFT, DIFUSCO, and DiffUCO by Wu et al. (2026). LEMON obtains the best result on 7 and tied-best result on 1 out of the 9 settings, spanning both small and large graphs, while remaining competitive on the remaining case. Bold denotes the best value, and underlined denotes the 2nd best value. Runtime is averaged over 100 graphs for LEMON.

n	q	ReduMIS	pCQO-MIS	Best learning	LEMON	runtime (s)
100	5	<u>38.70</u>	<u>39.33</u>	<u>39.50</u>	**39.67**	0.6730
100	15	<u>21.33</u>	<u>21.63</u>	21.62	**21.66**	0.7278
300	5	<u>121.66</u>	<u>123.22</u>	123.12	**123.58**	1.8374
300	15	<u>70.66</u>	<u>71.38</u>	70.75	**71.43**	1.58357
300	50	**50.00**	**50.00**	**50.00**	**50.00**	1.43431
1000	5	**417.88**	416.625	<u>417.12</u>	416.00	9.1235
1000	15	<u>241.00</u>	241.875	243.12	**243.75**	8.0013
1000	50	<u>115.33</u>	110.75	113.00	**115.66**	8.14452
1000	150	**150.00**	**150.00**	**150.00**	**150.00**	6.4030

Table 11: *Max-k-cut on the G-set benchmarks.* Results span binary max-cut ($k = 2$) and max-k-cut ($k = 3, 4, 5$). "Public best" denotes the reference value reported for each graph and k, and the relative gap is $100(\text{Public best} - \text{LEMON})/\text{Public best}$. Across the 144 problem settings with 51 different G-set graphs, LEMON matches the public-best value on 26 and is within 1% on all but three; notably, it attains the best reference value for all $k = 2, \ldots, 5$ on G48 and G49, for all listed k on G50, and for $k = 4, 5$ on G70. The n and $|E|$ columns give the graph dimensions, and the runtime is the recorded LEMON solver time.

Graph	n	$\lvert E\rvert$	k	LEMON	Runtime (s)	Public best	Relative Gap (%)
G1	800	19176	2	11597	1.3465	11624	0.232
G1	800	19176	3	15164	1.3097	15165	0.007
G1	800	19176	4	16802	1.3169	16804	0.012
G1	800	19176	5	17695	1.3239	17705	0.056
G2	800	19176	2	11620	1.3538	11620	0.000
G2	800	19176	3	15165	1.3010	15172	0.046
G2	800	19176	4	16806	1.3087	16810	0.024
G2	800	19176	5	17693	1.3184	17707	0.079
G3	800	19176	2	11622	1.3916	11622	0.000
G3	800	19176	3	15172	1.3000	15173	0.007
G3	800	19176	4	16798	1.3041	16806	0.048
G3	800	19176	5	17689	1.3096	17703	0.079
G4	800	19176	2	11638	1.3477	11646	0.069
G4	800	19176	3	15167	1.3283	15184	0.112
G4	800	19176	4	16795	1.3682	16814	0.113
G4	800	19176	5	17695	1.3617	17709	0.079
G10	800	19176	2	1999	2.2153	2000	0.050
G11	800	1600	2	564	1.3085	564	0.000
G12	800	1600	2	556	1.3069	556	0.000
G13	800	1600	2	582	1.2859	582	0.000
G14	800	4694	2	3057	1.2811	3064	0.228
G14	800	4694	3	4009	1.3036	4012	0.075
G14	800	4694	4	4428	1.3152	4440	0.270
G14	800	4694	5	4630	1.3160	4642	0.259
G15	800	4661	2	3048	1.2807	3050	0.066
G15	800	4661	3	3980	1.3015	3984	0.100
G15	800	4661	4	4395	1.3239	4407	0.272
G15	800	4661	5	4594	1.3128	4607	0.282
G16	800	4672	2	3052	1.3328	3052	0.000
G16	800	4672	3	3985	1.3531	3992	0.175
G16	800	4672	4	4399	1.3073	4418	0.430
G16	800	4672	5	4606	1.3261	4616	0.217
G17	800	4667	2	3044	1.2839	3047	0.098
G17	800	4667	3	3974	1.3069	3986	0.301
G17	800	4667	4	4398	1.3227	4415	0.385
G17	800	4667	5	4602	1.3803	4611	0.195
G18	800	4694	2	988	1.3270	992	0.403
G19	800	4661	2	906	1.3149	906	0.000
G20	800	4672	2	936	1.2869	941	0.531
G21	800	4667	2	927	1.2809	931	0.430
G22	2000	19990	2	13358	1.3340	13359	0.007

Table 11 continued

Graph	n	$\|E\|$	k	LEMON	Runtime (s)	Public best	Relative Gap (%)
G22	2000	19990	3	17156	1.3207	17169	0.076
G22	2000	19990	4	18733	1.3337	18776	0.229
G22	2000	19990	5	19529	1.4066	19554	0.128
G23	2000	19990	2	13340	1.3713	13344	0.030
G23	2000	19990	3	17147	1.3212	17172	0.146
G23	2000	19990	4	18755	1.3417	18779	0.128
G23	2000	19990	5	19507	1.3420	19549	0.215
G24	2000	19990	2	13335	1.3887	13337	0.015
G24	2000	19990	3	17149	1.4034	17163	0.082
G24	2000	19990	4	18748	1.4053	18775	0.144
G24	2000	19990	5	19501	1.3284	19545	0.225
G25	2000	19990	2	13329	1.3267	13340	0.082
G25	2000	19990	3	17150	1.3331	17166	0.093
G25	2000	19990	4	18758	1.3196	18779	0.112
G25	2000	19990	5	19521	1.3291	19557	0.184
G26	2000	19990	2	13320	1.3221	13328	0.060
G26	2000	19990	3	17137	1.3079	17161	0.140
G26	2000	19990	4	18738	1.3556	18772	0.181
G26	2000	19990	5	19508	1.3221	19539	0.159
G27	2000	19990	2	3336	1.3111	3341	0.150
G28	2000	19990	2	3292	1.2990	3298	0.182
G29	2000	19990	2	3389	1.2933	3405	0.470
G30	2000	19990	2	3412	1.3024	3413	0.029
G31	2000	19990	2	3307	1.3280	3310	0.091
G32	2000	4000	2	1402	1.3049	1410	0.567
G33	2000	4000	2	1372	1.2880	1382	0.724
G34	2000	4000	2	1376	1.2894	1384	0.578
G35	2000	11778	2	7661	1.3161	7687	0.338
G35	2000	11778	3	10017	1.3331	10051	0.338
G35	2000	11778	4	11086	1.3869	11118	0.288
G35	2000	11778	5	11594	1.3303	11626	0.275
G36	2000	11766	2	7664	1.3048	7680	0.208
G36	2000	11766	3	10014	1.4016	10044	0.299
G36	2000	11766	4	11090	1.3120	11115	0.225
G36	2000	11766	5	11590	1.3175	11621	0.267
G37	2000	11785	2	7665	1.3092	7691	0.338
G37	2000	11785	3	10032	1.3078	10062	0.298
G37	2000	11785	4	11103	1.3086	11124	0.189
G37	2000	11785	5	11618	1.3196	11633	0.129
G38	2000	11779	2	7664	1.3051	7688	0.312
G38	2000	11779	3	10008	1.3122	10051	0.428
G38	2000	11779	4	11086	1.3391	11117	0.279
G38	2000	11779	5	11608	1.3138	11627	0.163
G39	2000	11778	2	2389	1.3262	2408	0.789
G40	2000	11766	2	2376	1.3413	2400	1.000
G41	2000	11785	2	2394	1.3030	2405	0.457
G42	2000	11779	2	2461	1.3108	2481	0.806
G43	1000	9990	2	6660	1.2958	6660	0.000
G43	1000	9990	3	8572	1.3157	8573	0.012
G43	1000	9990	4	9366	1.3474	9377	0.117
G43	1000	9990	5	9749	1.3252	9770	0.215
G44	1000	9990	2	6650	1.3215	6650	0.000
G44	1000	9990	3	8558	1.3577	8571	0.152
G44	1000	9990	4	9360	1.4419	9379	0.203
G44	1000	9990	5	9748	1.4021	9766	0.184
G45	1000	9990	2	6654	1.3388	6654	0.000
G45	1000	9990	3	8563	1.3438	8566	0.035
G45	1000	9990	4	9366	1.3077	9377	0.117
G45	1000	9990	5	9750	1.3466	9771	0.215
G46	1000	9990	2	6649	1.2980	6649	0.000
G46	1000	9990	3	8557	1.3993	8568	0.128
G46	1000	9990	4	9365	1.3233	9379	0.149
G46	1000	9990	5	9759	1.4229	9770	0.113
G47	1000	9990	2	6657	1.3048	6657	0.000
G47	1000	9990	3	8555	1.3039	8572	0.198
G47	1000	9990	4	9377	1.3204	9382	0.053
G47	1000	9990	5	9759	1.3480	9769	0.102
G48	3000	6000	2	6000	0.3115	6000	0.000
G48	3000	6000	3	6000	0.2992	6000	0.000
G48	3000	6000	4	6000	0.2923	6000	0.000
G48	3000	6000	5	6000	0.2962	6000	0.000
G49	3000	6000	2	6000	0.2691	6000	0.000
G49	3000	6000	3	6000	0.3300	6000	0.000
G49	3000	6000	4	6000	0.3263	6000	0.000
G49	3000	6000	5	6000	0.3008	6000	0.000
G50	3000	6000	2	5880	0.5382	5880	0.000
G50	3000	6000	3	6000	0.3145	6000	0.000
G50	3000	6000	4	6000	0.3324	6000	0.000

Table 11 continued

Graph	n	$\lvert E\rvert$	k	LEMON	Runtime (s)	Public best	Relative Gap (%)
G50	3000	6000	5	6000	0.3871	6000	0.000
G51	1000	5909	2	3846	1.3307	3848	0.052
G51	1000	5909	3	5027	1.3575	5039	0.238
G51	1000	5909	4	5564	1.3507	5576	0.215
G51	1000	5909	5	5819	1.3432	5838	0.325
G52	1000	5916	2	3848	1.4850	3851	0.078
G52	1000	5916	3	5037	1.3419	5045	0.159
G52	1000	5916	4	5568	1.3683	5588	0.358
G52	1000	5916	5	5822	1.3321	5842	0.342
G53	1000	5914	2	3847	1.4858	3850	0.078
G53	1000	5914	3	5021	1.3448	5045	0.476
G53	1000	5914	4	5570	1.3374	5582	0.215
G53	1000	5914	5	5821	1.3116	5838	0.291
G54	1000	5916	2	3843	1.3626	3852	0.234
G54	1000	5916	3	5030	1.3096	5040	0.198
G54	1000	5916	4	5566	1.3162	5584	0.322
G54	1000	5916	5	5822	1.3138	5839	0.291
G55	5000	12498	2	10219	1.3597	10299	0.777
G55	5000	12498	3	11730	1.3548	12429	5.624
G55	5000	12498	4	12351	1.3412	12498	1.176
G55	5000	12498	5	12487	1.3480	12498	0.088
G70	10000	9999	2	9266	1.4603	9591	3.389
G70	10000	9999	3	9968	1.4228	9999	0.310
G70	10000	9999	4	9999	0.4440	9999	0.000
G70	10000	9999	5	9999	0.5763	9999	0.000

F.5 k-WAY NPP BENCHMARK

Each row in Table 12 samples 100 independently generated $n = 1000$ elements multisets for the k-way NPP from a particular distribution. $\mathcal{N}[0,1]$ and $\mathcal{U}$ denote the standard normal and uniform distributions respectively. IRNP is the improved recursive number partitioning method (Korf, 2009; 2011); ACSA* denotes the insertion-and-swap trajectory of Chen et al. (2026) adapted for unconstrained k-way NPP; Swap is the swap-based method of Zhang (2024). They (along with LEMON) are run on the common Ryzen 7 9700X 8-Core CPU and 32 GB RAM with the same instance sets and runtime budget of ~ 2 seconds. LEMON gives the best (and mean) MSD on all of the six (five out of six) settings. Every one of the LEMON outputs satisfies $\min_i a_i > D$, which by the Proposition C.4, certifies the one-flip(item) local optimality. IRNP is deterministic and it minimizes the makespan (maximum subset-sum) (Korf, 2010) to find the best assignment while the others use MSD; so for IRNP we run it once for each problem, obtain the output assignment then calculate the MSD for an even comparison.

Table 12: *k-way number partitioning under maximum set discrepancy (MSD).* Best MSD and mean MSD are reported over 100 runs for $n = 1000$, across three input distributions and $k \in \{5, 10\}$; lower is better for all reported metrics. Compared with IRNP, ACSA*, and Swap, LEMON attains the lowest best MSD in all 6/6 settings and the lowest mean MSD in 5/6 settings. Bold denotes the best value for each metric, and underlined denotes the 2nd best value.

		LEMON		IRNP	ACSA*		Swap	
Distribution	k	best	mean	best	best	mean	best	mean
$\lvert\mathcal{N}(0,1)\rvert$	5	$\mathbf{5.1\times10^{-6}}$	$\mathbf{6.55\times10^{-6}}$	3.86×10^{-3}	6.09×10^{-4}	6.49×10^{-4}	$\underline{1.16\times10^{-5}}$	$\underline{1.16\times10^{-5}}$
$\mathcal{N}(0,1)\rvert$	10	$\mathbf{1.52\times10^{-6}}$	$\mathbf{7.96\times10^{-6}}$	3.08×10^{-3}	1.17×10^{-3}	1.51×10^{-3}	$\underline{2.8\times10^{-5}}$	$\underline{2.8\times10^{-5}}$
$\mathcal{U}[0,1]$	5	$\mathbf{1.46\times10^{-6}}$	$\underline{2.46\times10^{-6}}$	1.7×10^{-3}	4.42×10^{-4}	4.89×10^{-4}	$\underline{2.32\times10^{-6}}$	$\mathbf{2.32\times10^{-6}}$
$\mathcal{U}[0,1]$	10	$\mathbf{1.06\times10^{-6}}$	$\mathbf{3.84\times10^{-6}}$	3.32×10^{-3}	5.82×10^{-4}	8.06×10^{-4}	$\underline{1.85\times10^{-5}}$	$\underline{1.96\times10^{-5}}$
$\mathcal{U}$[32-bit]	5	$\mathbf{1.16\times10^{4}}$	$\mathbf{1.17\times10^{4}}$	2.41×10^{6}	2.52×10^{5}	3.03×10^{5}	$\underline{3.43\times10^{4}}$	$\underline{3.43\times10^{4}}$
$\mathcal{U}$[32-bit]	10	$\mathbf{1.17\times10^{4}}$	$\mathbf{4.19\times10^{4}}$	7.56×10^{6}	5.52×10^{5}	8.93×10^{5}	$\underline{7.22\times10^{4}}$	$\underline{7.22\times10^{4}}$