# AUGMENTED LAGRANGIAN OPTIMIZATION NEURAL NETWORK FOR MINIMUM VERTEX COVER

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Unsupervised neural solvers for minimum vertex cover (MVC) use continuous penalties to learn small feasible covers. We identify a geometric mismatch in the spherical MVC relaxation: rotating a selected vertex from a discrete cover decreases the objective at second order, while a squared edge penalty responds only at fourth order. Linear feedback acts at second order and can reverse this directional curvature. Guided by this distinction, we introduce ALON, an Augmented Lagrangian Optimization Neural Network combining gradient-based message passing, projected node feedback, and an edge-wise penalty. Node states reduce multiplier storage from edges to vertices. We quantify the information lost by this compression, bound edge violations as embeddings become discrete, and derive conditional feasibility and first-order residual bounds for projected updates. Across 19 datasets, the +AL configurations improve cover quality in 70 of 76 backbone–dataset pairs and tie three more. ALON attains the best printed neural mean on 16 of the 19 datasets, and amortized inference is about three orders of magnitude faster than a direct solve of the same relaxation on the WS$[100, 200]$ split.

## 1 INTRODUCTION

Minimum vertex cover (MVC) seeks the smallest set of vertices covering every edge, a basic NP-hard graph optimization problem (Karp, 1972). Graph neural networks offer amortized solvers that reuse training across instances (Bengio et al., 2021; Cappart et al., 2023). Unsupervised methods learn without optimal solution labels by minimizing continuous surrogates (Schuetz et al., 2022; Yau et al., 2024). Their loss must balance reducing the cover size with satisfying every edge constraint. This makes the form of constraint feedback central to the learned solver.

Quadratic penalties, used in the SDP-based OptGNN relaxation (Yau et al., 2024), control violations through a scalar coefficient. Beyond the usual trade-off between objective reduction and constraint accuracy (Nocedal & Wright, 2006), the spherical MVC relaxation has a specific geometric obstruction. At every nonempty discrete feasible cover, including an optimum, rotating a selected vertex decreases the objective at second order, while the squared edge penalty contributes only at fourth order. No finite increase in the penalty coefficient changes this order mismatch. On edges whose coverage depends on that vertex, multiplier feedback acts at second order and can reverse the directional curvature. The exact identity in Theorem 4.1 makes this distinction explicit and links feedback strength to the edges for which that vertex is responsible.

Representing this feedback efficiently raises a second question. Edge multipliers require one state per constraint, whereas node multipliers share feedback across incident edges. This compression introduces an information loss: signed edge violations can cancel when summed at a node. We combine node feedback with an edge-wise penalty that retains the canceled components. Our compression bound quantifies the resulting trade-off: aggregated residuals control true edge violations up to an error that vanishes as the embeddings become discrete (Theorem 4.2). This gives a mathematical basis for separating feedback storage from constraint evaluation.

These insights lead to ALON (Augmented Lagrangian Optimization Neural Network). Its message-passing layers use normalized objective/penalty and dual-gradient features, while an analytic outer loop updates node multipliers by projected ascent. The design brings augmented Lagrangian feedback (Hestenes, 1969) into a learned spherical relaxation with $O(n)$ dual-state storage. Related unrolled architectures use coupled networks to learn primal and dual updates (Hadou & Ribeiro,



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.

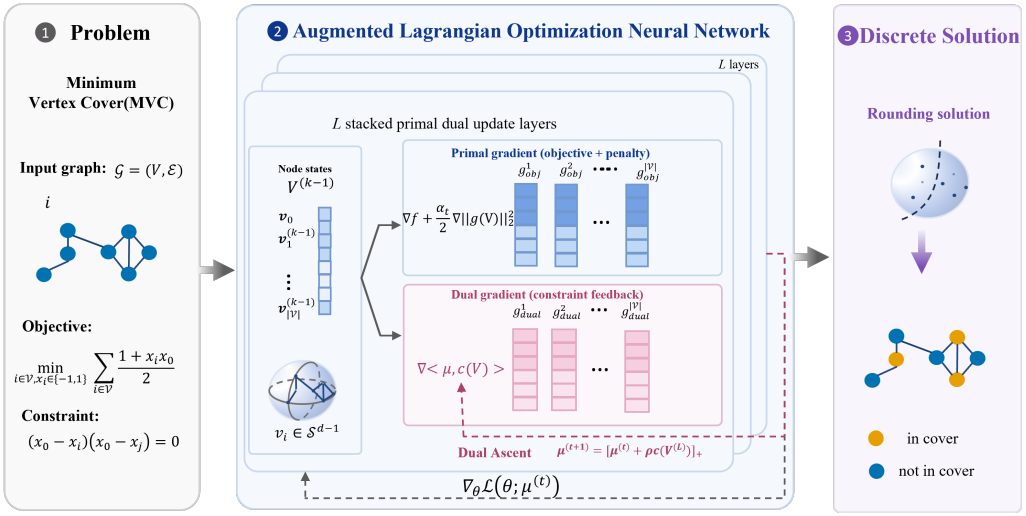


Figure 1: ALON embeds augmented Lagrangian dynamics into GNN message passing, combining node-level dual ascent with an edge-wise quadratic penalty. Minimum vertex cover is relaxed by the Burer–Monteiro factorization onto a product of spheres. Each of $L$ learned layers combines normalized objective/penalty and feedback gradient features with the previous state. Feature and training-loss coefficients are specified separately. Dual multipliers $\mu$ are updated in the outer loop (pink dashed) while $\theta$ is trained by backpropagation through the unrolled dynamics. Rounding with greedy repair gives a feasible cover.

2025; 2026); ALON uses explicit node feedback and analyzes its effect on local geometry and edge feasibility. Figure 1 summarizes the architecture. Our contributions are as follows.

- We characterize the distinct local roles of penalties and dual feedback in the spherical MVC relaxation. An exact rotation identity exposes the second-order objective decrease and fourth-order penalty response, and gives a feedback threshold that reverses the directional curvature. It also distinguishes vertices essential to the current cover from redundant ones that can be removed along a feasible path.
- We develop ALON, combining projected node feedback with gradient-based message passing and an edge-wise penalty. The design stores $O(n)$ multipliers in place of $O(m)$ edge multipliers while preserving evaluation of every edge constraint. The same augmented loss and update can equip standard GNN backbones.
- We give a quantitative analysis of compressed feedback, including a bound on edge violations, realizable cancellation patterns, and a distinction between residual visibility and stabilizing multiplier capacity. For the stated projected recurrence, we derive joint feasibility and first-order residual bounds with separate feedback step sizes and loss coefficients. Conditional KKT accumulation results and BM/SDP output certificates connect this analysis to stationarity and solution quality.
- Across 19 datasets and four GNN backbones, we evaluate the augmented configurations, distribution shift, weighted MVC, and node-versus-edge feedback. A matched WS ablation isolates the feedback intervention in that setting. Direct relaxation solves and classical heuristics provide distinct quality and runtime comparisons.

## 2 Related Work

### 2.1 Deep Learning for Combinatorial Optimization

Neural combinatorial optimization learns to construct or improve discrete solutions (Bengio et al., 2021; Cappart et al., 2023). Methods include supervised prediction (Li et al., 2018; Gasse et al., 2019), reinforcement learning (Kool et al., 2018; Barrett et al., 2020), and unsupervised optimization of continuous surrogates (Schuetz et al., 2022; Yau et al., 2024). Learned construction and improvement

offer complementary ways to amortize search (Joshi et al., 2019; Chen & Tian, 2019). ALON studies constraint feedback in an unsupervised relaxation-based solver.

### 2.2 Learning under constraints

Constraint handling includes penalty losses (Karalias & Loukas, 2020; Schuetz et al., 2022), differentiable optimization layers (Amos & Kolter, 2017; Agrawal et al., 2019), feasibility mappings (Donti et al., 2021; Tordesillas et al., 2023), and explicit multiplier updates (Fioretto et al., 2020; Park & Van Hentenryck, 2023; Kotary & Fioretto, 2024). Learned primal–dual architectures (Hadou & Ribeiro, 2025; 2026), differentiable dual optimization (Abbas & Swoboda, 2024), and multiplier prediction (Demelas et al., 2024) provide related approaches. ALON uses analytic node feedback and tracks edge residuals despite compressing multiplier storage; OptGNN (Yau et al., 2024) is its closest architectural antecedent.

### 2.3 From SDP Relaxation to the Burer–Monteiro Factorization

Semidefinite relaxations provide bounds for graph optimization problems (Vandenberghe & Boyd, 1996; Goemans & Williamson, 1995; Kleinberg & Goemans, 1998). The Burer–Monteiro factorization replaces a positive semidefinite matrix by a low-rank Gram representation (Burer & Monteiro, 2003; 2005). Its landscape guarantees depend on regularity and the cost family (Boumal et al., 2016; 2020). SATNet uses a differentiable low-rank SDP layer for logical reasoning (Wang et al., 2019). OptGNN is the closest architectural antecedent: it builds message passing from relaxation gradients and develops SDP certificates (Yau et al., 2024). ALON studies how compressed dual feedback changes the geometry of this relaxation and how aggregation affects edge feasibility. Its analysis connects these structural properties to the projected update and specializes SDP duality to the resulting effective multipliers.

## 3 Methodology

We relax MVC to an SDP, parameterize its positive semidefinite matrix by unit embeddings, and train a message-passing network using an augmented loss with projected node feedback.

### 3.1 Continuous Relaxation under Manifold Constraints

**Discrete formulation.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with $n$ vertices, $m$ unordered edges, and neighbor sets $\mathcal{N}(i)$. A vertex cover contains at least one endpoint of every edge. Encode membership by $x_i \in \{-1, +1\}$, with $+1$ denoting selection, and fix an auxiliary anchor $x_0 = 1$. The indicator $(1 + x_i x_0)/2$ gives the discrete formulation

$$\min_{\boldsymbol{x} \in \{-1,1\}^n} \quad \sum_{i \in \mathcal{V}} \tfrac{1+x_i x_0}{2} \tag{1}$$

$$\text{s.t.} \quad (x_0 - x_i)(x_0 - x_j) = 0, \quad \forall (i,j) \in \mathcal{E}, \tag{2}$$

which enforces that every edge has at least one endpoint in the cover.

**SDP relaxation.** We stack the anchor spin and the vertex spins into $\boldsymbol{x} = (x_0, x_1, \ldots, x_n)^\top \in \mathbb{R}^{n+1}$ and lift it by $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^\top$, which rewrites every quadratic term $x_p x_q$ as the matrix entry $\boldsymbol{X}_{pq}$. Relaxing the nonconvex rank one constraint $\operatorname{rank}(\boldsymbol{X}) = 1$ to $\boldsymbol{X} \succeq 0$ yields the Kleinberg and Goemans SDP (Kleinberg & Goemans, 1998).

$$\min_{\boldsymbol{X} \succeq 0} \quad \sum_{i \in \mathcal{V}} \tfrac{1+\boldsymbol{X}_{i,0}}{2} \tag{3}$$

$$\text{s.t.} \quad 1 - \boldsymbol{X}_{i,0} - \boldsymbol{X}_{j,0} + \boldsymbol{X}_{i,j} = 0, \quad \forall (i,j) \in \mathcal{E}, \tag{4}$$

$$\boldsymbol{X}_{ii} = 1, \quad \forall i \in \mathcal{V} \cup \{0\}. \tag{5}$$

The objective equation 3 counts the selected vertices, the edge constraint equation 4 expands the discrete condition using $x_0^2 = 1$, and the diagonal constraints equation 5 encode $x_k^2 = 1$. Since the feasible set contains the lifted discrete solutions, the SDP value is a valid lower bound on the discrete

optimum. Interior-point SDP solvers scale polynomially and become intractable at scale, motivating the factorization below.

**Burer–Monteiro factorization.** Factor $\boldsymbol{X} = YY^T$ with $Y = [\boldsymbol{e}_1; \boldsymbol{V}] \in \mathbb{R}^{(n+1)\times d}$, fixing the anchor by a common orthogonal rotation. The free variable $\boldsymbol{V}$ contains the $n$ vertex rows, so $\mathcal{M} = (\mathcal{S}^{d-1})^n$. The diagonal constraints hold exactly, and the continuous problem is

$$\begin{aligned} \min_{\boldsymbol{V}\in\mathcal{M}} \quad & f(\boldsymbol{V}) = \sum_{i\in\mathcal{V}} \tfrac{1+\langle \boldsymbol{v}_i, \boldsymbol{e}_1\rangle}{2}, \\ \text{s.t.} \quad & g_{ij}(\boldsymbol{V}) = \langle \boldsymbol{e}_1 - \boldsymbol{v}_i, \boldsymbol{e}_1 - \boldsymbol{v}_j\rangle = 0, \quad (i,j) \in \mathcal{E}. \end{aligned} \tag{6}$$

The residual equals $1 - X_{i0} - X_{j0} + X_{ij}$ and can take either sign on $\mathcal{M}$. This is the SDP with the additional constraint $\operatorname{rank}(X) \le d$. Its global value lies between the SDP and discrete MVC optima, with equality to the SDP whenever an SDP optimum has rank at most $d$ (Proposition 6).

### 3.2 Augmented Lagrangian Method

With loss coefficient $\beta_t > 0$, the residual in equation 6 defines the edge augmented Lagrangian

$$\mathcal{L}_{\mathrm{AL}}^{\mathrm{edge}}(\boldsymbol{V}, \boldsymbol{\lambda}) = f(\boldsymbol{V}) + \sum_{(i,j)\in\mathcal{E}} [\lambda_{ij} g_{ij}(\boldsymbol{V}) + \tfrac{\beta_t}{2} g_{ij}^2(\boldsymbol{V})], \tag{7}$$

where equality multipliers $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{E}|}$ are unrestricted. On discrete embeddings, $g_{ij}$ is zero for a covered edge and four otherwise. On continuous embeddings it is signed, and the squared penalty acts on both signs.

ALON uses a nonnegative node feedback state, defined below, while retaining this edge-wise penalty. The linear feedback and squared penalty have different roles: near discrete covers, feedback changes the second-order response while the squared penalty enters at fourth order (Section 4.1). Figure 2 illustrates the regimes on a single WS$[100, 200]$ instance, where node-multiplier feedback removes the redundant vertices left by the fixed penalty.

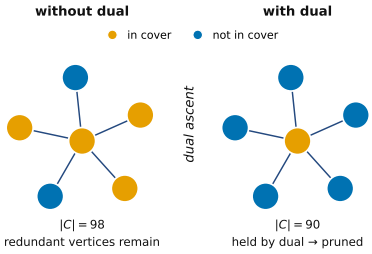


Figure 2: Dual feedback on a WS-$[100, 200]$ instance (test graph 1955). The model trained with node multipliers produces a smaller cover. Theorem 4.1 characterizes feedback near discrete covers; Figure 4 (Appendix A) gives the annotated comparison.

### 3.3 Node Level Dual Reparameterization

Restricting $\lambda_{ij} = \mu_i + \mu_j$ stores $n$ node multipliers in place of $m$ edge multipliers. For the unsigned incidence matrix $B$, define $\boldsymbol{c} = B\boldsymbol{g}$, or $c_i = \sum_{j\in\mathcal{N}(i)} g_{ij}$. Then

$$\sum_{(i,j)\in\mathcal{E}} (\mu_i + \mu_j) g_{ij} = \sum_i \mu_i c_i = \boldsymbol{\mu}^T B \boldsymbol{g}. \tag{8}$$

The loss coefficient $\beta_t$, feature coefficient $\alpha_t$, and feedback step size $\eta_t$ serve separate roles. Equations sum each undirected edge once and use paper-scaled multipliers. The loss and per-graph projected update are

$$\mathcal{L}_{\mathrm{AL}}^{\mathrm{node}}(\boldsymbol{V}, \boldsymbol{\mu}) = f(\boldsymbol{V}) + \boldsymbol{\mu}^T B \boldsymbol{g}(\boldsymbol{V}) + \tfrac{\beta_t}{2} \|\boldsymbol{g}(\boldsymbol{V})\|_2^2, \tag{9}$$

$$\boldsymbol{\mu}^{t+1} = [\boldsymbol{\mu}^t + \eta_t \boldsymbol{c}(\boldsymbol{V}_\theta)]_+. \tag{10}$$

The projected rule stores feedback at nodes and evaluates constraints on edges: $B^T\boldsymbol{\mu}$ restricts the available edge feedback, whereas the penalty still evaluates every edge. Node and edge feasibility coincide on discrete embeddings, but signed continuous residuals may cancel under aggregation. Section 4.2 quantifies this distinction.

### 3.4 ALON: Augmented Lagrangian Update Layers

Rather than optimize the relaxation separately for each graph, ALON amortizes the optimization across instances. A GNN with parameters $\theta = \{\boldsymbol{W}^{(k)}\}_{k=1}^L$ maps the input graph $\mathcal{G}$ to a parameterized variable $\boldsymbol{V}_\theta(\mathcal{G})$, and we minimize the resulting loss

$$\tilde{\mathcal{L}}^{\text{node}}_{\text{AL}}(\theta;\boldsymbol{\mu}) \triangleq f(\boldsymbol{V}_\theta(\mathcal{G})) + \sum_{i\in\mathcal{V}} \mu_i c_i(\boldsymbol{V}_\theta(\mathcal{G})) + \sum_{(i,j)\in\mathcal{E}} \frac{\beta_t}{2} g_{ij}^2(\boldsymbol{V}_\theta(\mathcal{G})). \tag{11}$$

The forward pass constructs an embedding update from gradient features at each layer. At layer $k$ we construct two gradient features at the current iterate $\boldsymbol{V}^{(k-1)}$ to obtain two decoupled vector fields, the objective gradient $\boldsymbol{g}^{(k)}_{\text{obj}} = \nabla_{\boldsymbol{V}}\big(f + \frac{\alpha_t}{2}\sum_{(i,j)\in\mathcal{E}} g_{ij}^2\big)$ and the constraint gradient $\boldsymbol{g}^{(k)}_{\text{dual}} = \nabla_{\boldsymbol{V}}\langle\boldsymbol{\mu},\boldsymbol{c}\rangle$. Both have closed forms whose block at node $i$ depends only on $\boldsymbol{v}_i$ and its neighbors, so evaluating them is exactly the neighborhood aggregation of a Message Passing Neural Network (MPNN).

The decoupled lifting layer treats these gradients as directional features. The shared weight $\boldsymbol{W}^{(k)}$ acts as the combine phase of an MPNN, fusing the messages with the current state,

$$\boldsymbol{h}_i^{(k)} = \boldsymbol{W}^{(k)}\Big[\,\boldsymbol{v}_i^{(k-1)}\;;\;\frac{\boldsymbol{g}^{(k)}_{\text{obj},i}}{\|\boldsymbol{g}^{(k)}_{\text{obj},i}\|_2}\;;\;\gamma\,\frac{\boldsymbol{g}^{(k)}_{\text{dual},i}}{\|\boldsymbol{g}^{(k)}_{\text{dual},i}\|_2}\Big], \tag{12}$$

where $[\cdot;\cdot]$ denotes concatenation and $\gamma \in (0,1)$ scales the dual contribution. Normalization supplies directional features, with a zero gradient represented by zero. The learned map combines these features to determine the embedding update. Finally, the intermediate output is retracted onto the sphere by $L_2$ normalization,

$$\boldsymbol{v}_i^{(k)} = \Pi_{\mathcal{S}^{d-1}}(\boldsymbol{h}_i^{(k)}) = \boldsymbol{h}_i^{(k)}/\|\boldsymbol{h}_i^{(k)}\|_2, \tag{13}$$

For nonzero projected inputs, every output row lies on the unit sphere. The differential statements in Appendix B.8 apply at differentiable network states.

Before normalization, the two features sum to the training objective gradient when $\alpha_t = \beta_t$. Distinct coefficients define related directional features. The learned combination and spherical projection do not require a literal gradient-descent step. Inference uses $\boldsymbol{\mu} = 0$ and no outer feedback iterations.

Algorithm 1 (Appendix A) summarizes the training dynamics. The forward pass constructs $\boldsymbol{V}_\theta(\mathcal{G})$ layer by layer, the backward pass minimizes $\tilde{\mathcal{L}}^{\text{node}}_{\text{AL}}$, and the outer loop updates $\boldsymbol{\mu}$ by dual ascent.

## 4 Theoretical Analysis

We study the geometry of the loss and the information retained by node feedback. Fix $\boldsymbol{v}_0 = \boldsymbol{e}_1$, let $B \in \mathbb{R}^{n\times m}$ be the unsigned incidence matrix, and write $\boldsymbol{c} = B\boldsymbol{g}$. The analysis also covers positive vertex weights, with $f_w = \sum_i w_i(1 + \langle \boldsymbol{v}_i, \boldsymbol{e}_1\rangle)/2$; ordinary MVC has $w_i = 1$. Set $F_{\rho,\boldsymbol{\mu}} = f_w + \boldsymbol{\mu}^T B\boldsymbol{g} + \rho\|\boldsymbol{g}\|^2/2$.

Here $\rho$ denotes a generic positive loss coefficient; at training step $t$, it is $\beta_t$. The feedback step $\eta_t$ and feature coefficient $\alpha_t$ remain distinct.

### 4.1 Local Geometry of Dual Feedback

A discrete cover $C$ has $\boldsymbol{v}_i = \boldsymbol{e}_1$ for $i \in C$ and $\boldsymbol{v}_i = -\boldsymbol{e}_1$ otherwise. At such a point, both $\boldsymbol{g}$ and its first tangential derivative vanish. Consequently, the quadratic penalty has zero Riemannian Hessian, and

$$\operatorname{Hess}_{\mathcal{M}} F_{\rho,\boldsymbol{\mu}} = \operatorname{Hess}_{\mathcal{M}}(f_w + \boldsymbol{\mu}^T B\boldsymbol{g}). \tag{14}$$

Proposition 1 proves this identity as a refinement of penalty persistence at feasible points. The following exact calculation identifies the contribution of dual feedback.

**Theorem 4.1** (Local feedback identity)**.** *Fix a discrete feasible cover $C$ and $i \in C$. Rotate only $\boldsymbol{v}_i$ to $\cos t\,\boldsymbol{e}_1 + \sin t\,\boldsymbol{u}$, where $\boldsymbol{u} \perp \boldsymbol{e}_1$ is unit length. Let $a = 1 - \cos t$, $k_i = |\mathcal{N}(i) \setminus C|$, and $D_i = \sum_{j\in\mathcal{N}(i)\setminus C}(\mu_i + \mu_j)$. Then*

$$F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}(t)) - F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}(0)) = (2D_i - w_i/2)a + 2\rho k_i a^2. \tag{15}$$

The objective and dual term act at order $t^2$, while the squared penalty acts at order $t^4$. Thus every nonempty discrete cover, including an optimum, has a negative-curvature direction under a pure finite quadratic penalty. Dual feedback supplies positive curvature along this rotation when $D_i > w_i/4$. A redundant vertex has $k_i = D_i = 0$ and can be removed along an edge-feasible path. Appendix B.10 characterizes the additional restrictions from coupled rotations.

### 4.2 What Node Compression Preserves

Define the polarization defect $\mathcal{P}(\boldsymbol{V}) = \sum_i \deg(i)(1 - \langle \boldsymbol{v}_i, \boldsymbol{e}_1 \rangle^2)$. It vanishes when all nonisolated vertices are discrete.

**Theorem 4.2** (Compression error bound)**.** *Every unit-row embedding satisfies*

$$\|\boldsymbol{g}\|_1 \leq \tfrac{1}{2}\mathbf{1}^T\boldsymbol{c} + \mathcal{P}(\boldsymbol{V}) \leq \tfrac{1}{2}\|\boldsymbol{c}\|_1 + \mathcal{P}(\boldsymbol{V}). \tag{16}$$

Node residuals therefore control edge feasibility near discrete outputs. Away from that regime, signed residuals can cancel in $B\boldsymbol{g}$. Appendix B.4 characterizes the invisible space $\ker B$ and constructs a realizable four-cycle example. The edge-wise penalty retains these residuals even when node feedback is zero.

### 4.3 First-Order Guarantees for Projected Feedback

Index the graph outputs used at projected feedback updates as $\boldsymbol{\mu}^{t+1} = [\boldsymbol{\mu}^t + \eta_t \boldsymbol{c}(\boldsymbol{V}^t)]_+$, with $\boldsymbol{\mu}^0 = 0$. For $S_T = \sum_{t<T} \eta_t$, define

$$H_T = \frac{\|\boldsymbol{\mu}^T\|_1}{2S_T} + \frac{\sum_{t<T} \eta_t \mathcal{P}(\boldsymbol{V}^t)}{S_T}. \tag{17}$$

The projected update and Theorem 4.2 give $\sum_{t<T} \eta_t \|\boldsymbol{g}(\boldsymbol{V}^t)\|_1 / S_T \leq H_T$ (Theorem B.1). This holds for arbitrary primal outputs and positive feedback step sizes.

For a separate positive loss coefficient $\beta_t$, set $\widehat{\boldsymbol{\lambda}}^t = B^T\boldsymbol{\mu}^t + \beta_t \boldsymbol{g}(\boldsymbol{V}^t)$ and $\epsilon_t = \|\operatorname{grad}_{\mathcal{M}} F_{\beta_t, \boldsymbol{\mu}^t}(\boldsymbol{V}^t)\|_F$. The effective multiplier gives the exact identity $\operatorname{grad} F = \operatorname{grad} f_w + D_{\mathcal{M}}\boldsymbol{g}^*\widehat{\boldsymbol{\lambda}}$. Let $\mathcal{R}$ be the sum of the edge-feasibility and Lagrangian stationarity residuals in equation 40, and let $A_T = \sum_{t<T} \eta_t \epsilon_t^2 / S_T$. Then

$$\min_{t<T} \mathcal{R}(\boldsymbol{V}^t, \widehat{\boldsymbol{\lambda}}^t) \leq \frac{\sum_{t<T} \eta_t \mathcal{R}(\boldsymbol{V}^t, \widehat{\boldsymbol{\lambda}}^t)}{S_T} \leq \sqrt{A_T} + H_T. \tag{18}$$

The bound separates embedding stationarity, multiplier growth, and proximity to discrete configurations. Appendices B.7–B.8 give the conditions for KKT accumulation and for transferring parameter-gradient accuracy to the embedding space.

These are output-dependent bounds. They do not establish that training drives polarization and embedding stationarity errors to zero. Parameter-gradient norms alone do not verify those conditions.

### 4.4 BM Representation and Output Certificates

The factorization restricts the SDP rank: $p_{\mathrm{SDP}} \leq p_{d+1} \leq p_d \leq p_{\mathrm{MVC}}$. Appendix B.9 derives an SDP lower bound $\ell(X, \boldsymbol{\lambda})$ from any output $X = YY^T$ and edge multiplier, including $\widehat{\boldsymbol{\lambda}}$. A positive semidefinite dual slack together with edge feasibility certifies SDP optimality; a repaired cover $C_{\mathrm{out}}$ satisfies $0 \leq w(C_{\mathrm{out}}) - p_{\mathrm{MVC}} \leq w(C_{\mathrm{out}}) - \ell$. The same slack bounds suboptimality of the fixed augmented subproblem. The certificates apply to arbitrary network outputs and assess solution quality through feasibility and the spectrum of the dual slack.

## 5 Experiments

We evaluate ALON along three dimensions: solution quality and scalability, out-of-distribution generalization, and computational efficiency of the dual reparameterization; protocols are in Appendix C.1.

**Datasets.** We consider three graph categories. First, synthetic graphs (ER (Erdős & Rényi, 1960), BA (Barabási & Albert, 1999), WS (Watts & Strogatz, 1998), HK (Holme & Kim, 2002)) at scales $|\mathcal{V}| \in \{[50, 100], [100, 200], [400, 500]\}$. Second, five real world datasets from the TU benchmark collection (Yanardag & Vishwanathan, 2015) (MUTAG (Debnath et al., 1991), ENZYMES, PROTEINS (Borgwardt et al., 2005), IMDB-BIN, COLLAB). Third, hard Forced-RB instances (RB200, RB500) (Xu et al., 2007) near the phase transition.

**Baselines and protocol.** We compare ALON, which builds on OptGNN (Yau et al., 2024) and replaces its fixed penalty by an augmented Lagrangian (AL) mechanism, against four GNN backbones (GIN (Xu et al., 2018), GAT (Veličković et al., 2017), GCNN (Morris et al., 2019) and GatedGCNN (Li et al., 2015)) and OptGNN itself, each backbone also equipped with the mechanism (+AL) to probe generality. And eight classical MVC solvers, greedy and maximal-matching 2-approximations and LP rounding (Vazirani, 2001); the local-search heuristics NuMVC (Cai et al., 2013) and FastVC (Cai et al., 2017), with FastWVC (Cai et al., 2019) for weighted instances; Gurobi (Gurobi Optimization, LLC, 2024); and an exact branch-and-reduce solver, the PACE 2019 Vertex Cover track entry (Hespe et al., 2020). Continuous-output methods in the unweighted MVC comparison use randomized hyperplane rounding and greedy repair. Weighted MVC uses the dedicated protocol in Appendix C.6.

Table 1: Main results on Minimum Vertex Cover. We report the average objective value (OBJ, lower is better, ↓). '+AL' denotes the backbone equipped with our augmented Lagrangian objective and node level dual ascent; ALON is its realization on the dedicated architecture. Among the method columns, bold marks the best value (shared when several methods tie, e.g. on MUTAG) and † the unique second best. The Reference column is the best available reference from Table 8: the exact optimum where the branch-and-reduce solve closes within 10 s, otherwise the best classical value (Appendix C.3). Run-to-run standard deviations are reported in Table 6.

| Dataset | Reference | GIN | | GAT | | GCNN | | GatedGCNN | | OptGNN | ALON |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Raw | +AL | Raw | +AL | Raw | +AL | Raw | +AL | | |
| *Synthetic Datasets* | | | | | | | | | | | |
| ER [50,100] | 53.52 | 69.62 | 63.56 | 74.65 | 74.65 | 63.23 | 56.95 | 63.05 | 59.12 | 55.50† | **55.28** |
| BA [50,100] | 42.05 | 51.86 | 49.41 | 73.18 | 49.35 | 52.51 | 46.87 | 50.91 | 44.17 | 42.39† | **42.13** |
| WS [50,100] | 45.46 | 51.62 | 47.79 | 70.03 | 69.87 | 57.69 | 52.91 | 46.09 | 45.81† | 45.89 | **45.60** |
| HK [50,100] | 43.96 | 53.90 | 49.95 | 76.81 | 51.58 | 55.14 | 50.07 | 54.42 | 46.73 | 44.28† | **44.04** |
| ER [100,200] | 122.57 | 150.45 | 147.39 | 150.71 | 150.67 | 141.42 | 138.76 | 143.09 | 138.02† | 146.29 | **137.30** |
| BA [100,200] | 84.17 | 114.75 | 111.76 | 150.92 | 113.32 | 111.87 | 103.30 | 114.32 | 100.95 | 85.46† | **84.49** |
| WS [100,200] | 89.63 | 101.82 | 95.07 | 148.65 | 93.41 | 118.02 | 97.90 | 93.23 | 92.70 | 91.27† | **90.43** |
| HK [100,200] | 84.66 | 114.13 | 112.03 | 151.17 | 108.59 | 112.17 | 107.04 | 114.34 | 85.94 | 85.06† | **85.04** |
| ER [400,500] | 409.88 | 449.65 | 449.85 | 449.85 | 449.85 | 449.57 | 449.08 | **445.85** | 447.55† | 449.85 | 449.80 |
| BA [400,500] | 246.89 | 363.22 | 337.66 | 445.46 | 351.62 | 348.88 | 297.77 | 362.12 | 298.25 | 250.65† | **250.22** |
| WS [400,500] | 271.19 | 317.99 | 287.84 | 296.87 | 280.12 | 334.59 | 299.87 | 290.95 | 294.99 | 278.39† | **278.01** |
| HK [400,500] | 246.94 | 364.33 | 329.93 | 440.77 | 354.45 | 349.11 | 303.96 | 365.33 | 273.48 | 250.45† | **249.09** |
| *Real-world Datasets (TU-small)* | | | | | | | | | | | |
| MUTAG | 7.74 | 10.63 | 8.11 | 13.47 | 8.00 | 12.74 | 8.00 | 7.79† | **7.74** | **7.74** | **7.74** |
| ENZYMES | 20.00 | 24.90 | 22.48 | 28.30 | 24.85 | 26.97 | 22.38 | 20.33 | 20.05† | 20.12 | **20.04** |
| PROTEINS | 24.96 | 32.22 | 28.02 | 41.00 | 27.18 | 33.28 | 28.19 | 25.49 | 25.19 | 25.10† | **25.09** |
| IMDB-BIN | 16.76 | 19.28 | 19.26 | 20.02 | 20.01 | 19.26 | 19.15 | 19.13 | 19.12 | 19.11† | **16.76** |
| COLLAB | 67.46 | 77.19 | 73.82 | 77.28 | 77.21 | 73.68 | 73.47 | 74.71 | 72.39 | 67.79† | **67.55** |
| *Hard RB Datasets* | | | | | | | | | | | |
| RB200 | 145.79 | 154.56 | 154.09 | 155.60 | 155.60 | 154.31 | 150.90† | 154.17 | **150.10** | 154.96 | 154.77 |
| RB500 | 467.93 | 493.26 | 492.62 | 494.09 | 493.10 | 492.64 | 491.53† | 492.94 | **491.40** | 493.52 | 491.98 |

### 5.1 Main Results and Scalability

Table 1 compares raw and augmented GNN configurations, OptGNN, and ALON. The +AL configurations improve 70 of 76 backbone–dataset pairs and tie three more; the three regressions occur at the largest synthetic scale and remain within $1.4\%$. Among the neural methods, ALON has the best printed mean on 16 of 19 datasets and matches the certified optimum on MUTAG and IMDB-BIN. The +AL family leads on 18 datasets, while raw GatedGCNN leads on ER$[400, 500]$. These configuration comparisons complement the matched WS feedback ablation in Appendix A; the Reference column separately reports classical solver quality.

### 5.2 Generalization under Distribution Shift

We train on a synthetic family (ER or BA) or a real source (MUTAG or COLLAB) and test zero shot on five real datasets without retraining (Table 3). The reported comparisons favor ALON in

Table 2: Weighted vertex cover. The architecture and covering constraints are retained; the objective and decoding are adapted to vertex weights, as specified in Appendix C.6. Entries are mean cover cost on the test split (lower is better,↓), with the best available reference in the Reference column. Bold marks the smaller mean.

| Family | Reference | OptGNN | ALON |
|---|---|---|---|
| ER | 332.07 | $336.99 \pm 0.16$ | **336.22** $\pm 0.20$ |
| BA | 259.25 | $261.41 \pm 0.12$ | **260.67** $\pm 0.21$ |
| MUTAG | 43.26 | $43.58 \pm 0.07$ | **43.42** $\pm 0.03$ |
| COLLAB | 265.95 | $266.52 \pm 0.11$ | **266.05** $\pm 0.05$ |

several source–target settings. These experiments assess transfer of the trained solver; they do not isolate which component causes the difference. References computed on a smaller target subset are not optimality-gap denominators for the full-target evaluation.

Table 3: Out-of-distribution (OOD) performance. Models trained on ER/BA (left block) or MUTAG/COLLAB (right block) are evaluated zero-shot on unseen real datasets (average OBJ, lower is better, ↓). Each cell aggregates two independent training runs; ± denotes the sample standard deviation, and bold marks the smaller mean. The parenthetical optima are computed on the classical test split, a smaller subset than the full target datasets used for the OOD evaluation, so a cell can sit above or below that reference. Diagonal cells (source evaluated on itself) are in-distribution controls, not OOD.

| **Test Dataset (Subset ref.)** | **Method** | **Synthetic source** | | **Real source** | |
|---|---|---|---|---|---|
| | | ER | BA | MUTAG | COLLAB |
| MUTAG (7.74) | OptGNN | $13.21 \pm 1.19$ | $8.68 \pm 0.72$ | $7.74 \pm 0.00$ | $7.79 \pm 0.07$ |
| | ALON | **8.36** $\pm 0.09$ | **8.17** $\pm 0.00$ | $7.74 \pm 0.00$ | **7.76** $\pm 0.04$ |
| ENZYMES (20.00) | OptGNN | $28.67 \pm 2.12$ | $21.71 \pm 1.70$ | $21.53 \pm 0.11$ | $20.27 \pm 0.26$ |
| | ALON | **20.87** $\pm 0.25$ | **20.04** $\pm 0.01$ | **20.47** $\pm 0.24$ | **20.06** $\pm 0.01$ |
| PROTEINS (24.96) | OptGNN | $34.74 \pm 2.59$ | $26.26 \pm 2.08$ | $27.49 \pm 0.20$ | $25.70 \pm 0.24$ |
| | ALON | **25.68** $\pm 0.56$ | **25.05** $\pm 0.07$ | **25.89** $\pm 0.32$ | **25.16** $\pm 0.03$ |
| IMDB-BIN (16.76) | OptGNN | $18.99 \pm 0.03$ | $18.30 \pm 0.86$ | $17.63 \pm 0.13$ | $16.77 \pm 0.01$ |
| | ALON | **17.59** $\pm 0.89$ | **17.18** $\pm 0.23$ | **17.23** $\pm 0.58$ | **16.76** $\pm 0.00$ |
| COLLAB (67.46) | OptGNN | $71.89 \pm 1.07$ | $68.98 \pm 1.03$ | $71.59 \pm 0.00$ | $67.79 \pm 0.12$ |
| | ALON | **68.88** $\pm 1.07$ | **68.67** $\pm 0.20$ | **69.59** $\pm 0.68$ | **67.55** $\pm 0.00$ |

**Beyond MVC.** The same architecture also applies to weighted vertex cover, by changing the vertex-weighted objective (Table 2). On weighted MVC, whose vertex weights span an order of magnitude, ALON stays within $1.3\%$ of the reported reference and attains a smaller mean cover cost than the fixed-penalty baseline on every family ($336.22$ vs. $336.99$ on ER, $260.67$ vs. $261.41$ on BA, $43.42$ vs. $43.58$ on MUTAG, and $266.05$ vs. $266.52$ on COLLAB).

### 5.3 Training Trajectories and Computational Cost

**Observed training trajectories.** Figure 3 compares parameter-gradient trajectories and repaired covers on Forced-RB instances. The penalty arms show a larger gradient decrease than ALON. The local identity in Theorem 4.1 distinguishes penalty and feedback forces in embedding space; the measured parameter gradients additionally depend on loss scaling, the network Jacobian, and the training trajectory. Their magnitude does not by itself establish embedding stationarity or explain the cover-quality difference.

**Memory and time.** Node reparameterization reduces multiplier storage and the projected state update from $\mathcal{O}(m)$ to $\mathcal{O}(n)$ (Table 4); forming $\boldsymbol{c} = B\boldsymbol{g}$ still traverses the edges.

**Time Complexity.** One forward pass costs $\mathcal{O}(L(|\mathcal{E}|d + |\mathcal{V}|d^2))$, the same order as an $L$-layer GNN (Appendix C.7). On the WS test split in Appendix C.8, the recorded direct-solve time is $111.40$ s and ALON forward plus decoding takes $39.11$ ms. The direct timer excludes decoding. This comparison

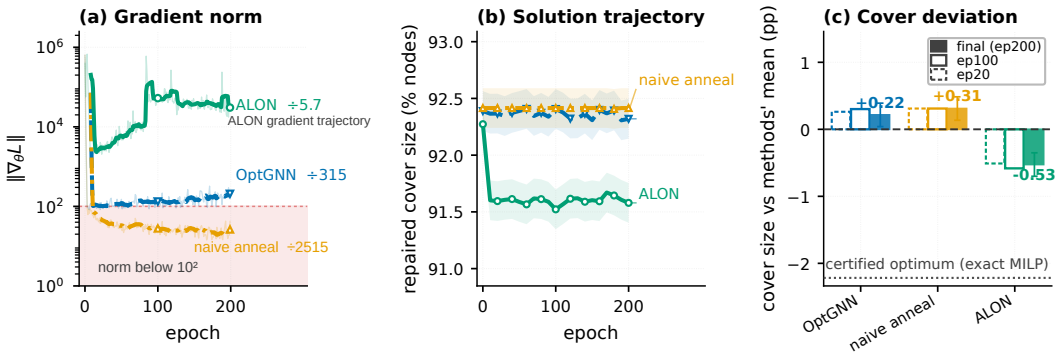


Figure 3: **Penalty dilemma analysis** on Forced-RB instances over the first 200 epochs. (a) Parameter gradient norm (log scale). Final/initial ten-epoch mean ratios are approximately $1/315$, $1/2515$, and $1/5.7$ for fixed penalty, annealed penalty, and ALON, respectively. (b) Repaired cover size (% of nodes). ALON reaches $91.6\%$ within 20 epochs; both penalty arms remain at $92.4\%$. (c) Deviation from the three-method final mean (solid: epoch 200; hollow: 20 and 100); the dotted line marks the certified optimum.

Table 4: Dual-state memory comparison on Forced-RB instances. Node level duals compress the dual state substantially, more on denser graphs. Totals aggregate over the full RB evaluation protocol (train+test splits as released). Total memory remains $\mathcal{O}(|\mathcal{E}|d)$-dominated by constraint gradients and no dual state is stored at inference.

| **Dataset** | Mean $\lvert\mathcal{V}\rvert$ | $\lvert\mathcal{E}\rvert/\lvert\mathcal{V}\rvert$ | **Dual Variables (total)** | | **Dual-state Memory** | |
|---|---|---|---|---|---|---|
| | | | Edge | Node | Edge | Node |
| RB200 | 126 | 59.4 | 9,360,574 | 157,335 | 35.7 MB | 0.6 MB |
| RB500 | 406 | 89.7 | 45,173,222 | 504,707 | 172.3 MB | 1.9 MB |

is specific to the direct relaxation solver; NuMVC and FastVC show no time-to-quality disadvantage at the tested sizes.

## 6 Conclusion and Limitations

ALON studies node feedback in learned optimization of the spherical MVC relaxation. The local analysis separates the roles of linear feedback and squared penalties; the compression analysis relates edge feasibility to polarization and distinguishes residual visibility from stabilizing capacity. Matched feedback experiments and broader solver comparisons assess these mechanisms at different levels. Empirically, the augmented configurations improve 70 of 76 backbone–dataset pairs, and ALON attains the best printed neural mean on 16 of 19 datasets. Transfer and weighted-cover evaluations extend the comparison beyond the training distribution and the unweighted objective. Together, these results support compressed feedback as a useful design for amortized neural MVC solvers. Inference uses zero node multipliers and no outer updates, so these gains reflect the solver learned during training and include its decoding procedure.

The projected-update bounds depend on trajectory quantities and do not prove convergence of network training or optimality of rounded covers. Node feedback restricts the multiplier space, and the smallest node–edge margins remain unresolved. The amortization comparison concerns direct relaxation solving; the tested classical local-search methods remain competitive or superior in time-to-quality. Empirical tightness and cost of the spectral certificates remain to be evaluated. The analysis motivates studying when observed polarization and residual structure justify additional edge feedback, and whether directional curvature thresholds predict learned trajectories. Such studies should keep the training objective and decoding budget fixed to distinguish optimization effects from decoding effects. Reduced multiplier storage still requires evaluating every edge constraint.

## 7 ETHICS STATEMENT

This work studies an unsupervised neural solver for a classical combinatorial optimization problem (minimum vertex cover) on synthetic graphs and on public benchmark datasets. It involves no human subjects, no personal data and no safety-critical deployment, and we see no direct dual-use concerns. The method is a research prototype for academic benchmarking. The reported experiments were run on a single-GPU-per-run basis. The energy footprint of any individual run is modest (the longest training run is a few GPU hours). The anonymous repository provides training and evaluation code, example commands, and a smoke-test checkpoint; the scope of the released artifacts is described in the reproducibility statement.

## 8 AI USE STATEMENT

Large language models assisted with language editing, organization of the theoretical analysis, derivation and checking of proof arguments, and manuscript consistency checks. They were also used for retrieval and discovery (e.g., finding related work). These tools were not used to generate the reported experimental measurements. The authors are responsible for the technical content and final manuscript.

## 9 REPRODUCIBILITY STATEMENT

All experiments specify seeds for graph generation, data splitting, training initialization and hyperplane rounding. Unweighted continuous-output methods use randomized hyperplane rounding of the continuous embeddings followed by greedy feasibility repair; the weighted decoder is specified in Appendix C.6. Every reported cover is checked for feasibility by enumerating the edges of each test graph. Reference values are the best classical solver result from Appendix C.3, computed locally with branch-and-reduce, Gurobi and local search. The per dataset hyperparameter configurations, the dataset generation protocols and the training infrastructure are given in Appendix C. Training and evaluation source, configuration definitions, and example commands are available at `https://anonymous.4open.science/r/alon-FBFD/`. The anonymous supplement includes per-arm reproduction commands for the WS$[50, 100]$ setting (200 epochs, 2000 graphs) and one smoke-test checkpoint. The repository README describes the provided artifacts and their reproduction commands.

## REFERENCES


Ahmed Abbas and Paul Swoboda. Doge-train: Discrete optimization on gpu with end-to-end training. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 20623–20631, 2024.

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019.

B. Amos and J. Z. Kolter. Optnet: Differentiable optimization as a layer in neural networks. *ICML*, 2017.

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

Thomas Barrett, William Clements, Jakob Foerster, and Alex Lvovsky. Exploratory combinatorial optimization with reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 3243–3250, 2020.

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1): i47–i56, 2005.

Nicolas Boumal, Vladislav Voroninski, and Afonso Bandeira. The non-convex burer-monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2765–2773, 2016.

Nicolas Boumal, Vladislav Voroninski, and Afonso S. Bandeira. Deterministic guarantees for Burer–Monteiro factorizations of smooth semidefinite programs. *Communications on Pure and Applied Mathematics*, 73(3):581–608, 2020.

Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical programming*, 95(2):329–357, 2003.

Samuel Burer and Renato DC Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical programming*, 103(3):427–444, 2005.

Shaowei Cai, Kaile Su, Chuan Luo, and Abdul Sattar. Numvc: An efficient local search algorithm for minimum vertex cover. *Journal of Artificial Intelligence Research*, 46:687–716, 2013. doi: 10.1613/jair.3907.

Shaowei Cai, Jinkun Lin, and Chuan Luo. Finding a small vertex cover in massive sparse graphs: Construct, local search, and preprocess. *Journal of Artificial Intelligence Research*, 59:463–494, 2017. doi: 10.1613/jair.5443.

Shaowei Cai, Yuanjie Li, Wenying Hou, and Haoran Wang. Towards faster local search for minimum weight vertex cover on massive graphs. *Information Sciences*, 471:64–79, 2019. doi: 10.1016/j.ins.2018.08.052.

Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*, 24(130):1–61, 2023.

Xinyun Chen and Yuandong Tian. Learning to perform local rewriting for combinatorial optimization. *Advances in neural information processing systems*, 32, 2019.

Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34 (2):786–797, 1991.

Francesco Demelas, Joseph Le Roux, Mathieu Lacroix, and Axel Parmentier. Predicting Lagrangian multipliers for mixed integer linear programs. In *International Conference on Machine Learning*, volume 235, pp. 10368–10384, 2024. URL https://proceedings.mlr.press/v235/demelas24a.html.

Priya L Donti, David Rolnick, and J Zico Kolter. Dc3: A learning method for optimization with hard constraints. *arXiv preprint arXiv:2104.12225*, 2021.

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5(1):17–61, 1960.

Ferdinando Fioretto, Pascal Van Hentenryck, Terrence WK Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 118–135. Springer, 2020.

Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. *Advances in neural information processing systems*, 32, 2019.

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.

Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2024.

Samar Hadou and Alejandro Ribeiro. Unrolled graph neural networks for constrained optimization. *arXiv preprint arXiv:2509.17156*, 2025.

Samar Hadou and Alejandro Ribeiro. Unrolled neural networks for constrained optimization. *arXiv preprint arXiv:2601.17274*, 2026.

Demian Hespe, Sebastian Lamm, Christian Schulz, and Darren Strash. Wegotyoucovered: The winning solver from the pace 2019 challenge, vertex cover track. In *2020 proceedings of the SIAM workshop on combinatorial scientific computing*, pp. 1–11. SIAM, 2020.

Magnus R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, 1969.

Petter Holme and Beom Jun Kim. Growing scale-free networks with tunable clustering. *Physical review E*, 65(2):026107, 2002.

Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint arXiv:1906.01227*, 2019.

Nikolaos Karalias and Andreas Loukas. Erdos goes neural: an unsupervised learning framework for combinatorial optimization on graphs. *Advances in Neural Information Processing Systems*, 33: 6659–6672, 2020.

Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger (eds.), *Complexity of Computer Computations*, pp. 85–103. Springer, 1972.

Jon Kleinberg and Michel X. Goemans. The Lovász theta function and a semidefinite programming relaxation of vertex cover. *SIAM Journal on Discrete Mathematics*, 11(2):196–204, 1998.

Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*, 2018.

James Kotary and Ferdinando Fioretto. Learning constrained optimization with deep augmented lagrangian methods. *arXiv preprint arXiv:2403.03454*, 2024.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Combinatorial optimization with graph convolutional networks and guided tree search. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4602–4609, 2019.

Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 2006.

Seonho Park and Pascal Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *AAAI Conference on Artificial Intelligence*, volume 37, pp. 4052–4060, 2023.

Mehmet Fatih Sahin, Armin Eftekhari, Ahmet Alacaoglu, Fabian Latorre, and Volkan Cevher. An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints. *Advances in Neural Information Processing Systems*, 32:13943–13955, 2019.

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

Jesus Tordesillas, Jonathan P How, and Marco Hutter. Rayen: Imposition of hard convex constraints on neural networks. *arXiv preprint arXiv:2307.08336*, 2023.

Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996.

Vijay V. Vazirani. *Approximation Algorithms*. Springer, Berlin, Heidelberg, 2001. ISBN 3-540-65367-8.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pp. 6545–6554. PMLR, 2019.

Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393 (6684):440–442, 1998.

Ke Xu, Frédéric Boussemart, Fred Hemery, and Christophe Lecoutre. Random constraint satisfaction: Easy generation of hard (satisfiable) instances. *Artificial intelligence*, 171(8-9):514–534, 2007.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Pinar Yanardag and Svn VN Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1365–1374, 2015.

Morris Yau, Nikolaos Karalias, Eric Lu, Jessica Xu, and Stefanie Jegelka. Are graph neural networks optimal approximation algorithms? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, 2024.

# A Algorithm and Mechanism Details

Algorithm 1 distinguishes loss, feature, and feedback coefficients. Residuals are cached from the forward outputs at which each batch is visited; these outputs need not share one frozen parameter vector. The analysis indexes each graph at its feedback updates and requires the stated projected recurrence and positive step sizes.

**Algorithm 1** ALON Training with Separate Coefficients

**Input:** training graph collection, parameters $\theta$, node states $\boldsymbol{\mu} = 0$, coefficients $\alpha_t, \beta_t$, optimizer and feedback schedules.

**for** epoch $t = 1, \ldots, T$ **do**
  Set feature coefficient $\alpha_t$ and loss coefficient $\beta_t$; clear $\boldsymbol{C}^{\mathrm{acc}}$.
  **for** training batch **do**
    Read its node states and initialize the spherical embeddings.
    **for** layer $k = 1, \ldots, L$ **do**
      Compute $\boldsymbol{g}$, $\boldsymbol{c} = B\boldsymbol{g}$, and the two gradient features using $\alpha_t$ and $\boldsymbol{\mu}$.
      Apply the learned combination in equation 12 and normalize each output row.
    **end for**
    Update $\theta$ with the declared optimizer and loss $F_{\beta_t, \boldsymbol{\mu}}(\boldsymbol{V})$.
    Accumulate $\boldsymbol{c}(\boldsymbol{V})$ from this forward output into $\boldsymbol{C}^{\mathrm{acc}}$.
  **end for**
  **if** a feedback update is due **then**
    Select $\eta_t$ by the declared schedule, including any batch normalization.
    $\boldsymbol{\mu} \leftarrow [\boldsymbol{\mu} + \eta_t \boldsymbol{C}^{\mathrm{acc}}]_+$.
  **end if**
**end for**

Theorem 4.1 gives a local restoring threshold for vertices with uncovered-side neighbors and an edge-feasible removal direction for redundant vertices. Figure 4 illustrates the learned outputs on one graph. In Figure 4 the final cover is drawn in orange and the remaining vertices in blue. The cover obtained without the dual retains redundant vertices that the dual removes. For vertex $v_{13}$, the alignment coordinate $x_{i0} = \langle v_i, e_1 \rangle$ moves from $-0.08$ (without dual) to $+0.51$ (with dual). The two final covers differ by eight vertices, 98 without the dual against 90 with it, and the latter is this graph's certified optimum. The same ordering holds in aggregate over all 200 test graphs $\times$ 3 seeds ($n$=600 paired observations): the two arms share the backbone, layer count, embedding dimension, schedule, epoch budget and decoding, and differ only in whether the dual feedback is applied during training. The mean repaired cover is $91.090 \pm 0.040$ without the dual against $90.428 \pm 0.162$ with it (paired difference $0.662$, lower is better), with $290/226/84$ per-graph wins/ties/losses and a pooled paired Wilcoxon $p = 3.12 \times 10^{-28}$. Graph–seed pairs reuse test graphs across seeds, so this pooled test is descriptive rather than evidence from 600 independent graphs. The raw (pre-repair) infeasible fraction falls from $89.5\%$ to $0.83\%$. A separate 200-graph evaluation gives $90.34$ with the dual and $91.09$ without, reproducing the same ordering. Both panels are decoded with the same rounding and the same feasibility repair. As a control for the penalty schedule itself, on the Forced-RB instances the ALON arm falls to $91.6\%$ of nodes within 20 epochs while the two dual-free penalty arms (fixed $\rho$=1 and annealed $\rho$:1$\rightarrow$100) remain at $92.4\%$; these two tested penalty schedules do not close the gap; they do not exhaust or match all effective penalty trajectories (Figure 3).

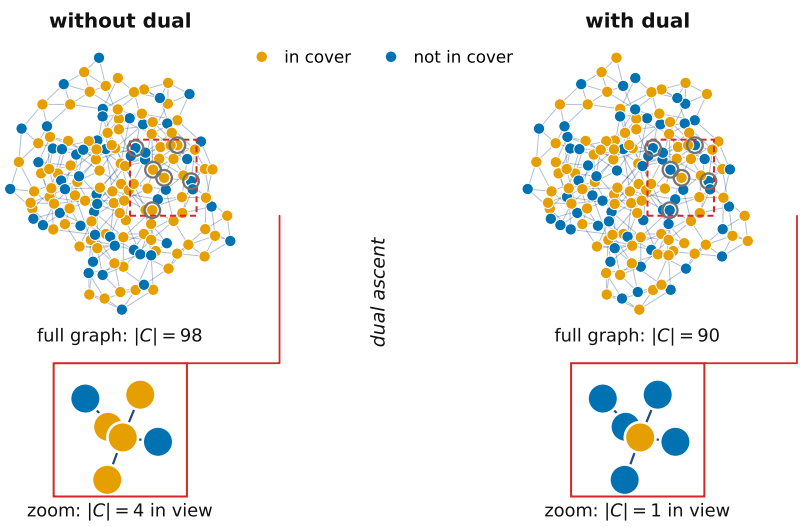


Figure 4: How the dual produces a smaller vertex cover on a real instance. The same test graph (idx=1955, $n = 151$, $m = 302$, WS[100,200] test split) is shown for the without-dual (left) and per epoch dual (right) model. Each panel draws the full graph, with orange vertices in the final cover and blue vertices outside it, and, below it, a magnified inset of the same six-vertex window (which contains 4 and 1 cover vertices respectively). The final cover sizes are 98 and 90, the latter being the certified optimum. All cover sizes refer to the full graph.

## B Theoretical Details and Proofs

We prove the local feedback and compression bounds, relate the projected updates to first-order KKT residuals, and derive BM and SDP optimality certificates. We distinguish conditional KKT accumulation from spectral certification, and residual information loss from restrictions on stabilizing multipliers.

### B.1 Setting and Discrete Degeneracy

Let $G = (\mathcal{V}, \mathcal{E})$ be a finite simple undirected graph with $n$ vertices and $m$ edges. The anchor $\boldsymbol{v}_0 = \boldsymbol{e}_1$ is fixed. The variable $\boldsymbol{V}$ comprises the $n$ remaining rows, so $\mathcal{M} = (\mathcal{S}^{d-1})^n$ has dimension $n(d-1)$. We assume $d \geq 2$. Define

$$f_w(\boldsymbol{V}) = \sum_i \frac{w_i}{2}(1 + \langle \boldsymbol{v}_i, \boldsymbol{e}_1 \rangle), \qquad g_{ij}(\boldsymbol{V}) = \langle \boldsymbol{e}_1 - \boldsymbol{v}_i, \boldsymbol{e}_1 - \boldsymbol{v}_j \rangle, \tag{19}$$

$$F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}) = f_w(\boldsymbol{V}) + \boldsymbol{\mu}^T B\boldsymbol{g}(\boldsymbol{V}) + \frac{\rho}{2}\|\boldsymbol{g}(\boldsymbol{V})\|_2^2, \qquad \boldsymbol{c}(\boldsymbol{V}) = B\boldsymbol{g}(\boldsymbol{V}). \tag{20}$$

Here $w_i > 0$, $w_i = 1$ for MVC, and each column of $B$ has ones at the two endpoints of its edge. Local statements fix $\rho$ and $\boldsymbol{\mu}$; the trajectory result allows both to vary according to the node update. The lifted matrix can be recovered as $X = \overline{V}\overline{V}^T$ by adjoining the anchor row to $\boldsymbol{V}$.

The constrained continuous problem studied below is

$$\min_{\boldsymbol{V} \in \mathcal{M}} f_w(\boldsymbol{V}) \quad \text{subject to} \quad \boldsymbol{g}(\boldsymbol{V}) = \boldsymbol{0}. \tag{21}$$

Its equality multipliers are unrestricted edge variables. They are distinct from the nonnegative node state used by the algorithm. For lifted certificates we write $Y = \overline{V}$ and $N = n + 1$, so the anchor is included in $X = YY^T$.

**Lemma 1** (Discrete feasibility and vanishing first derivatives)**.** *At an embedding $\boldsymbol{v}_i = s_i\boldsymbol{e}_1$, $s_i \in \{-1, 1\}$, the residual is $g_{ij} = (1 - s_i)(1 - s_j) \in \{0, 4\}$. Consequently, $\boldsymbol{c} = \boldsymbol{0}$ if and only if $\boldsymbol{g} = \boldsymbol{0}$. At every such embedding, feasible or infeasible,* $\operatorname{grad} f_w = 0$ *and* $\operatorname{grad} g_{ij} = 0$*. Thus* $\operatorname{grad} F_{\rho,\boldsymbol{\mu}} = 0$ *for every fixed $\rho$, $\boldsymbol{\mu}$.*

*Proof.* The residual formula follows by substitution. Every $c_i$ is a sum of nonnegative edge residuals, so these sums all vanish exactly when all edge residuals vanish. For the derivative statement, the tangent projection at $\boldsymbol{v}_i$ is $P_i = I - \boldsymbol{v}_i \boldsymbol{v}_i^T$. Direct differentiation gives

$$\operatorname{grad}_{\boldsymbol{v}_i} f_w = \frac{w_i}{2} P_i \boldsymbol{e}_1, \qquad \operatorname{grad}_{\boldsymbol{v}_i} g_{ij} = P_i(\boldsymbol{v}_j - \boldsymbol{e}_1). \tag{22}$$

At a discrete embedding, both projected vectors are zero. The chain rule proves the final statement. □

For any differentiable parameterization $\theta \mapsto \boldsymbol{V}_\theta \in \mathcal{M}$, the same calculation gives $D_\theta \boldsymbol{c} = 0$ at a discrete embedding. Both optimal covers and infeasible discrete assignments are therefore first-order stationary. The local curvature and projected-feedback analyses below distinguish properties that this stationarity condition leaves unresolved.

### B.2 Penalty Persistence and Discrete Curvature

For fixed $\boldsymbol{\mu}$, let $L_{\boldsymbol{\mu}} = f_w + \boldsymbol{\mu}^T B \boldsymbol{g}$ and $p = \|\boldsymbol{g}\|^2/2$, so $F_{\rho,\boldsymbol{\mu}} = L_{\boldsymbol{\mu}} + \rho p$.

**Proposition 1** (Persistence at edge-feasible points)**.** *At any $\boldsymbol{V}_*$ with $\boldsymbol{g}(\boldsymbol{V}_*) = 0$,*

$$\operatorname{grad} F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}_*) = \operatorname{grad} L_{\boldsymbol{\mu}}(\boldsymbol{V}_*), \tag{23}$$

$$\operatorname{Hess} F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}_*) = \operatorname{Hess} L_{\boldsymbol{\mu}}(\boldsymbol{V}_*) + \rho D_{\mathcal{M}} \boldsymbol{g}(\boldsymbol{V}_*)^* D_{\mathcal{M}} \boldsymbol{g}(\boldsymbol{V}_*). \tag{24}$$

*Hence an edge-feasible critical point remains critical, and an edge-feasible second-order stationary point remains second-order stationary, for every $\rho \geq 0$. At a discrete feasible cover the added Hessian is zero, proving equation 14.*

*If* $\operatorname{argmin}_{\mathcal{M}} L_{\boldsymbol{\mu}} \cap \{\boldsymbol{g} = 0\} \neq \varnothing$*, then* $\min_{\mathcal{M}} F_{\rho,\boldsymbol{\mu}} = \min_{\mathcal{M}} L_{\boldsymbol{\mu}}$ *for all* $\rho \geq 0$*, and*

$$\operatorname{argmin}_{\mathcal{M}} F_{\rho,\boldsymbol{\mu}} = \operatorname{argmin}_{\mathcal{M}} L_{\boldsymbol{\mu}} \cap \{\boldsymbol{g} = 0\} \quad \text{for } \rho > 0. \tag{25}$$

*Proof.* For any tangent vector $\boldsymbol{\xi}$, differentiation on $\mathcal{M}$ gives

$$\operatorname{grad} p = D_{\mathcal{M}} \boldsymbol{g}^* \boldsymbol{g}, \qquad \operatorname{Hess} p[\boldsymbol{\xi}, \boldsymbol{\xi}] = \|D_{\mathcal{M}} \boldsymbol{g}[\boldsymbol{\xi}]\|^2 + \sum_e g_e \operatorname{Hess} g_e[\boldsymbol{\xi}, \boldsymbol{\xi}]. \tag{26}$$

At an edge-feasible point, the second sum and the gradient vanish, yielding equation 23–equation 24. At a discrete point, Lemma 1 also gives $D_{\mathcal{M}} \boldsymbol{g} = 0$. Finally, $F_{\rho,\boldsymbol{\mu}} \geq L_{\boldsymbol{\mu}} \geq \min L_{\boldsymbol{\mu}}$, and any edge-feasible global minimizer of $L_{\boldsymbol{\mu}}$ attains both inequalities. For $\rho > 0$, equality requires both $L_{\boldsymbol{\mu}} = \min L_{\boldsymbol{\mu}}$ and $p = 0$, proving equation 25. □

The general feasible-point identity specializes here to equality of Hessians at discrete covers. At these points the edge penalty contributes no quadratic restoring force; changes in a singular Hessian's null directions are governed by higher-order terms. The global persistence statement applies when the fixed node state admits an edge-feasible global minimizer of $L_{\boldsymbol{\mu}}$.

### B.3 Proof and Consequences of the Local Feedback Identity

*Proof of Theorem 4.1.* Let $I = \mathcal{V} \setminus C$. Since $C$ is a cover, $I$ is independent, and every edge residual initially vanishes. Only edges incident to the rotated vertex $i$ can change. For $j \in C$, $\boldsymbol{e}_1 - \boldsymbol{v}_j = 0$, so $g_{ij}(\boldsymbol{V}(t)) = 0$. For $j \in I$, $\boldsymbol{e}_1 - \boldsymbol{v}_j = 2\boldsymbol{e}_1$, so

$$g_{ij}(\boldsymbol{V}(t)) = 2\langle \boldsymbol{e}_1 - \boldsymbol{v}_i(t), \boldsymbol{e}_1 \rangle = 2(1 - \cos t) = 2a. \tag{27}$$

The objective changes by $-w_i a/2$, the dual term by $2D_i a$, and the penalty by $2\rho k_i a^2$. Adding these changes proves equation 15. □

The second derivative at $t = 0$ is $2D_i - w_i/2$. With zero multipliers it is negative for every selected vertex, independent of the finite penalty parameter. Lemma 1 then implies that every nonempty discrete cover is a stationary point with a negative-curvature direction for the pure squared-penalty objective. This includes discrete minimum covers. The squared penalty contributes $\rho k_i t^4/2 + O(t^6)$ along the rotation.

When $k_i > 0$, minimizing along this path is equivalent to minimizing a scalar quadratic over $a \in [0, 2]$, giving

$$a_i^* = \mathrm{clip}_{[0,2]}\left(\frac{w_i/2 - 2D_i}{4\rho k_i}\right). \tag{28}$$

For zero multipliers and an interior minimizer, $a_i^* = w_i/(8\rho k_i)$; as $\rho \to \infty$, the corresponding angle satisfies $t_i^* \sim \sqrt{w_i/(4\rho k_i)}$. With $D_i \geq w_i/4$, $a_i^* = 0$ for any $\rho > 0$. Equality at the threshold leaves a positive fourth-order penalty along this path; a strict inequality gives a positive second derivative.

If $k_i = 0$, the vertex is redundant: all its neighbors already lie in $C$. Its rotation keeps every edge residual zero and decreases the objective by $w_i a/2$, including at $t = \pi$, which removes the vertex from the discrete cover. Thus the local response depends on coverage responsibility rather than membership in the current cover alone. These statements describe individual vertex rotations; Appendix B.10 examines the additional restrictions created by simultaneous rotations.

### B.4 Which Residuals Does Node Aggregation Preserve?

Set $Q = BB^T = D + A_G$, where $D$ and $A_G$ are the degree and adjacency matrices. Let $Q^\dagger$ be the Moore–Penrose inverse. The orthogonal projector onto $\mathrm{range}(B^T)$ is $P = B^T Q^\dagger B$.

**Proposition 2** (Visible and invisible edge residuals)**.** *For $\boldsymbol{c} = B\boldsymbol{g}$ and $\boldsymbol{g}_{\mathrm{hid}} = (I - P)\boldsymbol{g}$,*

$$B\boldsymbol{g}_{\mathrm{hid}} = 0, \qquad \|\boldsymbol{g}\|_2^2 = \boldsymbol{c}^T Q^\dagger \boldsymbol{c} + \|\boldsymbol{g}_{\mathrm{hid}}\|_2^2. \tag{29}$$

*If $b$ is the number of bipartite connected components, counting isolated vertices, then* $\dim\ker B = m - n + b$.

*Proof.* The singular value decomposition of $B$ gives the projector formula and $B(I - P) = 0$. Since $P\boldsymbol{g} = B^T Q^\dagger \boldsymbol{c}$ and $\boldsymbol{c} \in \mathrm{range}(Q)$,

$$\|P\boldsymbol{g}\|^2 = \boldsymbol{c}^T Q^\dagger Q Q^\dagger \boldsymbol{c} = \boldsymbol{c}^T Q^\dagger \boldsymbol{c}. \tag{30}$$

Orthogonality yields equation 29. To compute the rank, observe that $B^T\boldsymbol{x} = 0$ means $x_i = -x_j$ on every edge. On a connected bipartite component, these equations have a one-dimensional solution space. An odd cycle forces all entries on its connected component to be zero. Hence $\dim\ker B^T = b$, and rank–nullity gives the result. □

For a connected graph, $B$ is injective on the edge space exactly when the graph is a tree or has one cycle and that cycle is odd. On these graphs $\boldsymbol{c} = 0$ implies $\boldsymbol{g} = 0$ for every continuous embedding. On a general graph, node feedback cannot observe components in $\ker B$. The edge-wise penalty retains their squared norm in equation 29; replacing it by $\|\boldsymbol{c}\|^2$ would remove this term. The rank characterization is a standard incidence-matrix fact; its role here is to describe the information retained by the multiplier parameterization.

**A realizable cancellation example.** For the four-cycle with edges $(1, 2), (2, 3), (3, 4), (4, 1)$, take any $s \in (0, 1)$ and, in dimension three, set

$$\boldsymbol{v}_i = s\boldsymbol{e}_1 + (1 - s)b_i\boldsymbol{e}_2 + \sqrt{2s(1 - s)}\, t_i\boldsymbol{e}_3, \quad \boldsymbol{b} = (1, -1, 1, -1), \quad \boldsymbol{t} = (1, 1, -1, -1). \tag{31}$$

Each row has squared norm $s^2 + (1 - s)^2 + 2s(1 - s) = 1$. Consecutive entries of $\boldsymbol{b}$ have opposite signs, whereas the products of consecutive entries of $\boldsymbol{t}$ alternate. Substitution gives

$$\boldsymbol{g} = 2s(1 - s)(1, -1, 1, -1)^T, \qquad B\boldsymbol{g} = \boldsymbol{0}. \tag{32}$$

Thus exact node feasibility need not imply edge feasibility even for realizable unit-row embeddings. As $s \to 1$, these embeddings approach the all-selected discrete configuration, and the nonzero residuals vanish with the squared angular deviation. This motivates the quantitative neighborhood bound proved next.

### B.5 Proof of the Compression Error Bound

*Proof of Theorem 4.2.* Write $a_i = \langle \boldsymbol{v}_i, \boldsymbol{e}_1 \rangle$, $\boldsymbol{z}_i = \boldsymbol{v}_i - a_i \boldsymbol{e}_1$, and $\eta_i = \|\boldsymbol{z}_i\| = \sqrt{1 - a_i^2}$ and $R(\boldsymbol{V}) = \sum_{(i,j)\in\mathcal{E}} \eta_i \eta_j$. The orthogonal decomposition of each row gives

$$g_{ij} = (1 - a_i)(1 - a_j) + \boldsymbol{z}_i^T \boldsymbol{z}_j. \tag{33}$$

Because $a_i \leq 1$, the first term is nonnegative. Cauchy–Schwarz therefore implies $[-g_{ij}]_+ \leq \eta_i \eta_j$. Using $\mathbf{1}^T \boldsymbol{c} = 2 \sum_{(i,j)} g_{ij}$, we obtain the sharper bound

$$\|\boldsymbol{g}\|_1 = \sum_{(i,j)} g_{ij} + 2 \sum_{(i,j)} [-g_{ij}]_+ \leq \frac{1}{2} \mathbf{1}^T \boldsymbol{c} + 2R(\boldsymbol{V}). \tag{34}$$

Finally, $2\eta_i \eta_j \leq \eta_i^2 + \eta_j^2$ gives

$$R(\boldsymbol{V}) \leq \frac{1}{2} \sum_i \deg(i) \eta_i^2 = \frac{1}{2} \mathcal{P}(\boldsymbol{V}). \tag{35}$$

Since $\mathbf{1}^T \boldsymbol{c} \leq \|\boldsymbol{c}\|_1$, both inequalities in equation 16 follow. □

At a discrete embedding $\mathcal{P} = 0$, so this bound recovers the equivalence in Lemma 1. More generally, with $\delta_i = 1 - |a_i|$, the inequality $1 - a_i^2 \leq 2\delta_i$ yields

$$\|\boldsymbol{g}\|_1 \leq \frac{1}{2} \|\boldsymbol{c}\|_1 + 2 \sum_i \deg(i) \delta_i. \tag{36}$$

For $m > 0$ and $\delta_i \leq \delta$ at every vertex, this becomes $\|\boldsymbol{g}\|_1 / m \leq \|\boldsymbol{c}\|_1 / (2m) + 4\delta$. These are output-dependent bounds: they quantify the effect of polarization without assuming that a particular optimizer produces it.

### B.6 Proof of the Finite-Step Projected Feedback Bound

**Theorem B.1** (Average edge feasibility)**.** *For arbitrary $\boldsymbol{V}^t \in \mathcal{M}$, $\eta_t > 0$, and the update $\boldsymbol{\mu}^{t+1} = [\boldsymbol{\mu}^t + \eta_t \boldsymbol{c}(\boldsymbol{V}^t)]_+$ with $\boldsymbol{\mu}^0 = 0$,*

$$\frac{\sum_{t<T} \eta_t \|\boldsymbol{g}(\boldsymbol{V}^t)\|_1}{S_T} \leq \frac{\|\boldsymbol{\mu}^T\|_1}{2S_T} + \frac{\sum_{t<T} \eta_t \mathcal{P}(\boldsymbol{V}^t)}{S_T} = H_T, \qquad S_T = \sum_{t<T} \eta_t. \tag{37}$$

*Proof of Theorem B.1.* For every coordinate, projection onto the nonnegative half-line satisfies

$$\boldsymbol{\mu}^{t+1} = [\boldsymbol{\mu}^t + \eta_t \boldsymbol{c}(\boldsymbol{V}^t)]_+ \geq \boldsymbol{\mu}^t + \eta_t \boldsymbol{c}(\boldsymbol{V}^t). \tag{38}$$

Summing over iterations and coordinates, and using $\boldsymbol{\mu}^0 = 0$ and $\boldsymbol{\mu}^T \geq 0$, gives

$$\sum_{t<T} \eta_t \mathbf{1}^T \boldsymbol{c}(\boldsymbol{V}^t) \leq \mathbf{1}^T \boldsymbol{\mu}^T = \|\boldsymbol{\mu}^T\|_1. \tag{39}$$

Multiply the first inequality of equation 16 by $\eta_t$, sum over $t$, apply equation 39, and divide by $S_T$. This proves equation 37. Using equation 34 instead gives the sharper version with $2 \sum_t \eta_t R(\boldsymbol{V}^t) / S_T$ in place of the polarization term. □

If $S_T \to \infty$, $\|\boldsymbol{\mu}^T\|_1 = o(S_T)$ and $\sum_{t<T} \eta_t \mathcal{P}(\boldsymbol{V}^t) = o(S_T)$, the weighted average of absolute edge residuals tends to zero. These are conditions on multiplier growth and the primal trajectory. The conclusion concerns weighted-average feasibility; KKT accumulation additionally uses the first-order accuracy conditions below.

For a fixed training collection with separate node states, the argument applies to each graph indexed by its update times, or to their disjoint union with full residual evaluation. A graph omitted from an update receives weight zero in the telescoping sum. This result uses exact within-graph residuals and accommodates arbitrary primal outputs.

### B.7 First-Order Certificates and Conditional KKT Accumulation

The projected node update preserves the edge-wise penalty in equation 20. We analyze this update directly. General inexact ALM results require specified primal accuracy and geometric conditions (Sahin et al., 2019); the results below instead expose the primal residual of each learned output.

Let $D_{\mathcal{M}}\boldsymbol{g}(\boldsymbol{V}) : T_{\boldsymbol{V}}\mathcal{M} \to \mathbb{R}^m$ denote the differential of the edge residual, and let $D_{\mathcal{M}}\boldsymbol{g}(\boldsymbol{V})^*$ be its adjoint under the Frobenius and Euclidean inner products. Define

$$\mathcal{R}(\boldsymbol{V}, \boldsymbol{\lambda}) = \|\boldsymbol{g}(\boldsymbol{V})\|_1 + \|\operatorname{grad}_{\mathcal{M}} f_w(\boldsymbol{V}) + D_{\mathcal{M}}\boldsymbol{g}(\boldsymbol{V})^*\boldsymbol{\lambda}\|_F. \tag{40}$$

The norm constraints are satisfied exactly on $\mathcal{M}$. Since the edge constraints are equalities, their multipliers $\boldsymbol{\lambda} \in \mathbb{R}^m$ have no sign restriction; $\mathcal{R} = 0$ is precisely their first-order KKT condition together with feasibility.

**Proposition 3** (Effective multipliers and first-order residuals)**.** *For any $\boldsymbol{V} \in \mathcal{M}$, $\rho > 0$, and $\boldsymbol{\mu}$, set*

$$\widehat{\boldsymbol{\lambda}} = B^T\boldsymbol{\mu} + \rho\boldsymbol{g}(\boldsymbol{V}). \tag{41}$$

*Then*

$$\operatorname{grad}_{\mathcal{M}} F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V}) = \operatorname{grad}_{\mathcal{M}} f_w(\boldsymbol{V}) + D_{\mathcal{M}}\boldsymbol{g}(\boldsymbol{V})^*\widehat{\boldsymbol{\lambda}}, \tag{42}$$

*and*

$$\mathcal{R}(\boldsymbol{V}, \widehat{\boldsymbol{\lambda}}) \le \|\operatorname{grad}_{\mathcal{M}} F_{\rho,\boldsymbol{\mu}}(\boldsymbol{V})\|_F + \tfrac{1}{2}\|\boldsymbol{c}(\boldsymbol{V})\|_1 + \mathcal{P}(\boldsymbol{V}). \tag{43}$$

*Proof.* Holding $\rho$ and $\boldsymbol{\mu}$ fixed, differentiation of equation 20 gives the two constraint contributions $D_{\mathcal{M}}\boldsymbol{g}^*B^T\boldsymbol{\mu}$ and $\rho D_{\mathcal{M}}\boldsymbol{g}^*\boldsymbol{g}$. This proves equation 42. Substitute it into equation 40 and apply Theorem 4.2 to the feasibility term. □

The effective multiplier combines the stored node feedback $B^T\boldsymbol{\mu}$ with the signed edge correction $\rho\boldsymbol{g}$. This correction can lie outside the cone representable by nonnegative node multipliers, making $\widehat{\boldsymbol{\lambda}}$ an edge-space quantity for the residual and spectral analyses.

**Theorem B.2** (Joint residual bound for projected feedback)**.** *Under the update and initialization of Theorem B.1, let $\beta_t > 0$ be a separately specified loss coefficient and define*

$$\widehat{\boldsymbol{\lambda}}^t = B^T\boldsymbol{\mu}^t + \beta_t\boldsymbol{g}(\boldsymbol{V}^t), \qquad \epsilon_t = \|\operatorname{grad}_{\mathcal{M}} F_{\beta_t,\boldsymbol{\mu}^t}(\boldsymbol{V}^t)\|_F, \tag{44}$$

$$A_T = \frac{\sum_{t<T}\eta_t\epsilon_t^2}{S_T}, \qquad H_T = \frac{\|\boldsymbol{\mu}^T\|_1}{2S_T} + \frac{\sum_{t<T}\eta_t\mathcal{P}(\boldsymbol{V}^t)}{S_T}. \tag{45}$$

*For arbitrary primal outputs $\boldsymbol{V}^t \in \mathcal{M}$ and positive $\eta_t$,*

$$\min_{t<T}\mathcal{R}(\boldsymbol{V}^t, \widehat{\boldsymbol{\lambda}}^t) \le \frac{\sum_{t<T}\eta_t\mathcal{R}(\boldsymbol{V}^t, \widehat{\boldsymbol{\lambda}}^t)}{S_T} \le \sqrt{A_T} + H_T. \tag{46}$$

*Proof.* Proposition 3 gives $\mathcal{R}(\boldsymbol{V}^t, \widehat{\boldsymbol{\lambda}}^t) = \|\boldsymbol{g}(\boldsymbol{V}^t)\|_1 + \epsilon_t$. Theorem B.1 bounds the weighted average of the first term by $H_T$. Weighted Cauchy–Schwarz gives

$$\frac{\sum_{t<T}\eta_t\epsilon_t}{S_T} \le \left(\frac{\sum_{t<T}\eta_t\epsilon_t^2}{S_T}\right)^{1/2}.$$

Adding these inequalities proves the average bound. The minimum is at most this positive-weight average. □

Both residual components in equation 46 are evaluated at the same iterate. The bound separates primal stationarity error from the multiplier growth and polarization terms controlled by the feedback identity. The best-iterate bound characterizes the trajectory; the experiments use the decoding protocol in Appendix C.2.

**Proposition 4** (KKT accumulation from effective multipliers)**.** *Suppose a sequence $t_j \to \infty$ satisfies $\mathcal{R}(\boldsymbol{V}^{t_j}, \widehat{\boldsymbol{\lambda}}^{t_j}) \to 0$, and its effective multipliers are bounded. A further subsequence converges to a pair $(\boldsymbol{V}^*, \boldsymbol{\lambda}^*)$ satisfying*

$$\boldsymbol{g}(\boldsymbol{V}^*) = \boldsymbol{0}, \qquad \operatorname{grad}_{\mathcal{M}} f_w(\boldsymbol{V}^*) + D_{\mathcal{M}}\boldsymbol{g}(\boldsymbol{V}^*)^*\boldsymbol{\lambda}^* = \boldsymbol{0}. \tag{47}$$

*Proof.* Compactness of $\mathcal{M}$ and boundedness of the effective multipliers give a convergent subsequence. Continuity of $\boldsymbol{g}$, its differential, and the tangent projections transfers the vanishing residual to equation 47. □

Bounded node states and penalties suffice for bounded effective multipliers. If $\boldsymbol{g} \to 0$ and penalties are bounded, the correction $\beta_t \boldsymbol{g}$ vanishes; with unbounded penalties, edge feasibility alone does not control this correction.

The polarization bound gives a second route specific to this problem, for which bounded effective multipliers are unnecessary.

**Proposition 5** (Conditional accumulation under polarization). *If $S_T \to \infty$, $A_T \to 0$, and $H_T \to 0$ in Theorem B.2, there is a sequence $t_j \to \infty$ such that $\mathcal{R}(\boldsymbol{V}^{t_j}, \widehat{\boldsymbol{\lambda}}^{t_j}) \to 0$. A further subsequence converges to an edge-feasible $\boldsymbol{V}^*$ whose nonisolated vertices are discrete and which satisfies equation 47 with $\boldsymbol{\lambda}^* = 0$, without a multiplier boundedness assumption.*

*Proof.* Theorems B.1 and B.2, together with the definition of $H_T$, imply

$$\frac{\sum_{t<T} \eta_t \big(\epsilon_t^2 + \|\boldsymbol{g}(\boldsymbol{V}^t)\|_1 + \mathcal{P}(\boldsymbol{V}^t)\big)}{S_T} \le A_T + 2H_T \longrightarrow 0. \tag{48}$$

All summands are nonnegative. A positive lower bound on any tail would contradict this limit because the total weight diverges and every finite prefix has finite weight. Hence one can select $t_j \to \infty$ along which all three terms vanish. In particular $\mathcal{R} = \|\boldsymbol{g}\|_1 + \epsilon_{t_j} \to 0$. Compactness gives a further subsequence with $\boldsymbol{V}^{t_j} \to \boldsymbol{V}^*$. Continuity gives $\boldsymbol{g}(\boldsymbol{V}^*) = 0$ and $\mathcal{P}(\boldsymbol{V}^*) = 0$. Every nonisolated row is therefore $\pm \boldsymbol{e}_1$, where the corresponding objective and constraint gradients vanish. At an isolated vertex, the constraints have no derivative at any iterate, so its objective-gradient block equals the block of $\operatorname{grad} F$ and tends to zero with $\epsilon_{t_j}$. Thus $\operatorname{grad} f_w(\boldsymbol{V}^*) = 0$, and $\boldsymbol{\lambda}^* = 0$ proves equation 47. □

The propositions characterize KKT cluster points under their stated trajectory conditions. Every discrete feasible cover, including a suboptimal one, satisfies KKT with zero edge multipliers. The spectral certificates below assess objective quality through the dual slack.

### B.8 From Network Parameters to Embedding Stationarity

At a differentiable training state, hold the graph, multipliers, loss coefficient, and feature coefficient fixed, and write

$$\Psi_t(\theta) = F_{\beta_t, \boldsymbol{\mu}^t}(\boldsymbol{V}_\theta(\mathcal{G}; \boldsymbol{\mu}^t, \alpha_t)), \qquad J_t = D_\theta \boldsymbol{V}_\theta(\mathcal{G}; \boldsymbol{\mu}^t, \alpha_t). \tag{49}$$

An embedding KKT point yields a parameter KKT point by the chain rule, with the same multiplier. For approximate stationarity, the chain rule gives $\nabla_\theta \Psi_t = J_t^* \operatorname{grad}_{\mathcal{M}} F_{\beta_t, \boldsymbol{\mu}^t}$.

**Lemma 2** (Transfer of first-order accuracy). *If, for some $\sigma > 0$, the output differential satisfies*

$$\|J_t^* \boldsymbol{q}\|_2 \ge \sigma \|\boldsymbol{q}\|_F \quad \text{for every } \boldsymbol{q} \in T_{\boldsymbol{V}_\theta} \mathcal{M}, \tag{50}$$

*then $\| \operatorname{grad}_{\mathcal{M}} F_{\beta_t, \boldsymbol{\mu}^t}(\boldsymbol{V}_\theta)\|_F \le \|\nabla_\theta \Psi_t\|_2 / \sigma$.*

*Proof.* Apply equation 50 to the Riemannian gradient and use the chain rule. □

Condition equation 50 is a quantitative surjectivity condition on the output differential, stronger than a parameter-count requirement. For shared-parameter training, it applies to the joint output map into the product of the graph manifolds and a consistently scaled joint loss. Averaged parameter gradients can cancel across graphs, so per-graph accuracy requires control of this joint map. For the experimental architecture, the analysis uses embedding residuals directly; the tangent-coverage condition remains an additional assumption for parameter-to-embedding transfer.

### B.9 BM Factorization and SDP Optimality Certificates

The BM representation separates a rank restriction from the original SDP constraints. Its standard landscape guarantees depend on the constraint geometry and the cost family (Boumal et al., 2020). Here we give certificates for a specified output. SDP dual certificates from neural embeddings also appear in OptGNN (Yau et al., 2024); we specialize weak duality to the MVC constraints and relate the resulting slack matrix to ALON's effective multipliers.

**Proposition 6** (Rank-restricted representation)**.** *Let $p_{\mathrm{SDP}}$, $p_d$, and $p_{\mathrm{MVC}}$ be the global optimal values of the SDP, the constrained BM problem at dimension $d$, and discrete MVC, respectively, with the same positive vertex weights. Then*

$$p_{\mathrm{SDP}} \le p_{d+1} \le p_d \le p_{\mathrm{MVC}}. \tag{51}$$

*The BM problem is equivalent to the SDP with $\mathrm{rank}(X) \le d$. If the SDP admits an optimal solution of rank at most $d$, then $p_d = p_{\mathrm{SDP}}$; in particular, this holds for $d \ge n+1$.*

*Proof.* Adjoining the anchor gives a unit-row factor $Y \in \mathbb{R}^{(n+1)\times d}$ and $X = YY^T$ with diagonal one. This preserves all objective and edge-constraint values. Conversely, every such rank-restricted matrix admits a unit-row factor, whose anchor can be aligned with $\boldsymbol{e}_1$ by a common orthogonal transformation. Increasing $d$ enlarges the feasible set. Every discrete cover gives a rank-one feasible matrix. These inclusions prove equation 51 and the equality claim. □

For the certificates, let $N = n+1$ and $\mathcal{E}_N = \{X \succeq 0 : \mathrm{diag}(X) = \mathbf{1}\}$. Define symmetric matrices $A$ and $G_{ij}$ by

$$\langle A, X\rangle = \frac{1}{2}\sum_i w_i(X_{00} + X_{0i}), \tag{52}$$

$$\langle G_{ij}, X\rangle = X_{00} - X_{0i} - X_{0j} + X_{ij}. \tag{53}$$

Thus the constants in $f_w$ and $\boldsymbol{g}$ are represented by $X_{00} = 1$. Write $\mathcal{G}(X)_e = \langle G_e, X\rangle$ for the edge operator, so $f_w(\boldsymbol{V}) = \langle A, X\rangle$ and $\boldsymbol{g}(\boldsymbol{V}) = \mathcal{G}(X)$ for $X = YY^T$.

For any $X \in \mathcal{E}_N$ and any $\boldsymbol{\lambda} \in \mathbb{R}^m$, define

$$C_{\boldsymbol{\lambda}} = A + \sum_e \lambda_e G_e, \qquad \boldsymbol{y} = \mathrm{diag}(C_{\boldsymbol{\lambda}} X), \tag{54}$$

$$S_{\boldsymbol{\lambda}} = C_{\boldsymbol{\lambda}} - \mathrm{Diag}(\boldsymbol{y}), \qquad \kappa_{\boldsymbol{\lambda}} = \max\{0, -\lambda_{\min}(S_{\boldsymbol{\lambda}})\}. \tag{55}$$

These definitions give $\langle S_{\boldsymbol{\lambda}}, X\rangle = 0$ and $\mathbf{1}^T\boldsymbol{y} = \langle A, X\rangle + \boldsymbol{\lambda}^T\mathcal{G}(X)$, even without stationarity or edge feasibility.

**Proposition 7** (SDP lower bound from an arbitrary output)**.** *The quantity*

$$\ell(X, \boldsymbol{\lambda}) = \mathbf{1}^T\boldsymbol{y} - N\kappa_{\boldsymbol{\lambda}} \tag{56}$$

*satisfies $\ell(X, \boldsymbol{\lambda}) \le p_{\mathrm{SDP}} \le p_{\mathrm{MVC}}$ for every $X \in \mathcal{E}_N$ and every $\boldsymbol{\lambda}$. If $\mathcal{G}(X) = 0$, then*

$$0 \le \langle A, X\rangle - p_{\mathrm{SDP}} \le N\kappa_{\boldsymbol{\lambda}}. \tag{57}$$

*In particular, edge feasibility and $S_{\boldsymbol{\lambda}} \succeq 0$ certify global optimality for the SDP.*

*Proof.* For any SDP-feasible $Z$, $\mathcal{G}(Z) = 0$, $\mathrm{diag}(Z) = \mathbf{1}$, and $\mathrm{Tr}(Z) = N$. Since $S_{\boldsymbol{\lambda}} + \kappa_{\boldsymbol{\lambda}} I \succeq 0$,

$$\langle A, Z\rangle = \langle C_{\boldsymbol{\lambda}}, Z\rangle = \mathbf{1}^T\boldsymbol{y} + \langle S_{\boldsymbol{\lambda}}, Z\rangle \ge \mathbf{1}^T\boldsymbol{y} - N\kappa_{\boldsymbol{\lambda}}. \tag{58}$$

Minimizing over feasible $Z$ proves the lower bound. If $X$ itself is edge-feasible, then $\mathbf{1}^T\boldsymbol{y} = \langle A, X\rangle$, and feasibility also gives $p_{\mathrm{SDP}} \le \langle A, X\rangle$, proving equation 57. □

For an infeasible output $X = YY^T$, the valid one-sided statement is

$$f_w(\boldsymbol{V}) - p_{\mathrm{SDP}} \le N\kappa_{\boldsymbol{\lambda}} - \boldsymbol{\lambda}^T\boldsymbol{g}(\boldsymbol{V}) \le N\kappa_{\boldsymbol{\lambda}} + \|\boldsymbol{\lambda}\|_\infty \|\boldsymbol{g}(\boldsymbol{V})\|_1. \tag{59}$$

Its left-hand side need not be nonnegative. For a feasible cover $C_{\mathrm{out}}$ returned by rounding and repair, weak duality instead yields the discrete quality certificate

$$0 \le w(C_{\mathrm{out}}) - p_{\mathrm{MVC}} \le w(C_{\mathrm{out}}) - \ell(X, \boldsymbol{\lambda}). \tag{60}$$

The certificate provides an output diagnostic whose tightness and computation time are outside the reported experiments.

To relate this slack matrix to the training loss, fix $\rho > 0$ and $\boldsymbol{\mu}$, and define the convex quadratic function

$$\phi_{\rho,\boldsymbol{\mu}}(X) = \langle A, X\rangle + \boldsymbol{\mu}^T B\mathcal{G}(X) + \frac{\rho}{2}\|\mathcal{G}(X)\|_2^2. \tag{61}$$

At $X = YY^T$, its gradient is $C_{\widehat{\boldsymbol{\lambda}}}$ with $\widehat{\boldsymbol{\lambda}} = B^T\boldsymbol{\mu} + \rho\boldsymbol{g}(\boldsymbol{V})$, exactly the effective multiplier in Proposition 3.

**Proposition 8** (Certificate for the fixed augmented subproblem)**.** *For $X \in \mathcal{E}_N$, form the slack matrix and $\kappa_{\widehat{\boldsymbol{\lambda}}}$ using the effective multiplier. Then*

$$0 \le \phi_{\rho,\boldsymbol{\mu}}(X) - \min_{Z\in\mathcal{E}_N} \phi_{\rho,\boldsymbol{\mu}}(Z) \le N\kappa_{\widehat{\boldsymbol{\lambda}}}. \tag{62}$$

*Proof.* The compact set $\mathcal{E}_N$ admits a minimizer $Z_*$. Convexity of $\phi = \phi_{\rho,\boldsymbol{\mu}}$ and $\langle S_{\widehat{\boldsymbol{\lambda}}}, X\rangle = 0$ give

$$\phi(X) - \phi(Z_*) \le \langle C_{\widehat{\boldsymbol{\lambda}}}, X - Z_*\rangle = -\langle S_{\widehat{\boldsymbol{\lambda}}}, Z_*\rangle \le N\kappa_{\widehat{\boldsymbol{\lambda}}}.$$

The first inequality in equation 62 follows from $X \in \mathcal{E}_N$. □

At an edge-feasible output, $\widehat{\boldsymbol{\lambda}} = B^T\boldsymbol{\mu}$, so the slack matrix for fixed $X, \boldsymbol{\mu}$ is independent of $\rho$. If it is positive semidefinite, $X$ minimizes the lifted linear objective $\langle C_{B^T\boldsymbol{\mu}}, X\rangle$ and is SDP optimal; its factor satisfies the global persistence condition in Proposition 1. Consequently, one effective multiplier supplies both a first-order certificate for the constrained BM problem and a spectral certificate for the lifted objective. A positive semidefinite slack certifies the fixed augmented subproblem; additional edge feasibility certifies the original SDP. A small first-order residual alone does not imply a small spectral defect.

**Remark 1** (Relation to rank-deficient second-order points)**.** *The usual rank-deficient argument underlying BM guarantees (Boumal et al., 2020) also applies to the convex objective equation 61. On the full unit-row factor manifold, let $Y$ be a second-order stationary point of $\phi(YY^T)$ with $\mathrm{rank}(Y) < d$. Choose $\boldsymbol{z} \neq 0$ with $Y\boldsymbol{z} = 0$. For any $\boldsymbol{x} \in \mathbb{R}^N$, $U = \boldsymbol{x}\boldsymbol{z}^T$ is tangent and $YU^T + UY^T = 0$. Writing $S = S_{\widehat{\boldsymbol{\lambda}}}$, the Riemannian Hessian satisfies*

$$\langle U, \mathrm{Hess}(\phi\circ\pi)(Y)[U]\rangle = 2\langle U, SU\rangle + D^2\phi(X)[YU^T + UY^T, YU^T + UY^T] = 2\|\boldsymbol{z}\|^2\boldsymbol{x}^T S\boldsymbol{x}, \tag{63}$$

*where $\pi(Y) = YY^T$. Second-order stationarity forces $S \succeq 0$, and Proposition 8 proves global optimality for the lifted augmented subproblem. The full factor formulation and the fixed-anchor formulation have the same objective values by common orthogonal rotations. Locally, any full-factor curve can be smoothly rotated to fix its anchor, so first- and second-order stationarity at critical points transfer between them. This gives a sufficient condition for global optimality of a specified rank-deficient second-order point.*

### B.10 Nonnegative Node Multipliers and Simultaneous Rotations

Injectivity of $B$ on edge residuals is distinct from the ability to represent a desired edge multiplier as $B^T\boldsymbol{\mu}$ with $\boldsymbol{\mu} \ge 0$. The following example identifies this distinction even on a tree.

**Proposition 9** (A restriction on stabilizing a discrete optimum)**.** *Consider the unweighted path* 1–2–3–4–5–6 *and the minimum cover $C = \{2, 4, 6\}$. For every $\boldsymbol{\mu} \ge 0$ and finite $\rho \ge 0$, the corresponding discrete embedding has a negative-curvature direction for $F_{\rho,\boldsymbol{\mu}}$. Unrestricted nonnegative edge multipliers can instead make its Riemannian Hessian positive semidefinite.*

*Proof.* The three disjoint edges $(1, 2), (3, 4), (5, 6)$ imply that every cover has size at least three, so $C$ is optimal. At its embedding, all residuals and their first derivatives vanish. Consequently,

the squared penalty has zero Riemannian Hessian there. Choose a unit $\boldsymbol{u} \perp \boldsymbol{e}_1$ and scalar tangent coefficients $x_i$. Along the curves $\boldsymbol{v}_i(t) = s_i \cos(tx_i)\boldsymbol{e}_1 + \sin(tx_i)\boldsymbol{u}$, with $s_i = 1$ on $C$ and $s_i = -1$ on its complement, the second variation equals $\boldsymbol{x}^T H \boldsymbol{x}$, where

$$H_{ii} = \begin{cases} 2\sum_{j\in\mathcal{N}(i)} \lambda_{ij} - 1/2, & i \in C, \\ 1/2, & i \notin C, \end{cases} \qquad H_{ij} = \lambda_{ij} \quad \text{for } (i,j) \in \mathcal{E}, \tag{64}$$

and other off-diagonal entries are zero. Direct substitution gives $\boldsymbol{s}^T H \boldsymbol{s} = 0$. If $H \succeq 0$, this implies $H\boldsymbol{s} = 0$, hence $\sum_{j\in\mathcal{N}(i)} \lambda_{ij} = 1/2$ at every vertex. Starting at the endpoints determines

$$(\lambda_{12}, \lambda_{23}, \lambda_{34}, \lambda_{45}, \lambda_{56}) = (1/2, 0, 1/2, 0, 1/2). \tag{65}$$

If $\lambda_{ij} = \mu_i + \mu_j$ with $\boldsymbol{\mu} \geq 0$, then $\lambda_{23} = 0$ forces $\mu_2 = \mu_3 = 0$. The equality $\lambda_{34} = 1/2$ then forces $\mu_4 = 1/2$, contradicting $\lambda_{45} = 0$. Thus $H$ is not positive semidefinite and has a negative eigenvalue. With free edge multipliers $(1/2, 0, 1/2, 0, 1/2)$, it instead consists of three positive semidefinite blocks $\frac{1}{2}\left(\begin{smallmatrix} 1 & 1 \\ 1 & 1 \end{smallmatrix}\right)$. Every tangent coordinate has this same block structure, proving the final statement. □

Together with Theorem 4.1, this example distinguishes the single-vertex restoring threshold from stability under coupled motion. It shows that residual visibility can coexist with restricted stabilizing feedback, even at a discrete optimum on a tree. Rounded-output quality is assessed in the experiments.

# C Experimental Details and Additional Results

## C.1 Datasets, Hyperparameters and Training Details

We evaluate on the base setting of Yau et al. (2024). We use (1) synthetic graphs ER ($p$=0.15), BA ($m$=4), WS ($k$=4, $p$=0.25) and HK ($m$=4, $p$=0.25) with $|V| \in \{[50, 100], [100, 200], [400, 500]\}$. (2) Real world datasets MUTAG, ENZYMES, PROTEINS, IMDB-BIN and COLLAB. (3) Hard Forced-RB instances (RB200 with $N \in [6, 15]$ and $K \in [12, 21]$, RB500 with $N \in [20, 34]$ and $K \in [10, 29]$). All experiments were run on NVIDIA A100-PCIE-40GB GPUs (40 GB each, driver 580.173.02, CUDA 13.0). Each training run occupies a single GPU.

Training proceeds in epochs over a finite set of graphs, since a fixed number of graphs is required to define an epoch. For the synthetic families we therefore generate finite datasets of 2000 graphs, while the TU datasets use all available samples. The batch size is chosen per dataset.

The four synthetic families use a batch size of 16. Among the TU datasets, MUTAG uses 2 (188 graphs), ENZYMES 5 (600), PROTEINS 9 (1113), IMDB-BIN 8 (1000) and COLLAB 40 (5000). The Forced-RB instances, generated near the phase transition, use 16 or 32. Every dataset is trained for 200 epochs, except ENZYMES, which uses 1000 epochs as in the ceiling experiment, and the RB instances, which use the longer 1000-epoch budget of the earlier baseline protocol. All RB backbone arms use two seeds. The full configuration is given in Table 5. Per-cell sd values are listed in Table 6.

## C.2 Feasibility of the Reported Solutions

Every returned cover is verified feasible. The discrete MVC solution of each test graph is decoded from the continuous embeddings $\boldsymbol{V}_\theta(\mathcal{G})$ of the Burer–Monteiro factorization by randomized hyperplane rounding. We draw 1000 unit hyperplanes from a fixed seed, project every node embedding onto them, and read off the induced $\pm 1$ assignment, where the first coordinate of each embedding is its alignment with the anchor $\boldsymbol{e}_1$ of the lifted representation and a vertex enters the cover when its projection is nonnegative (a zero projection is treated as selected). We score each candidate by $-(|C| + 4u)$, with $|C|$ the cover size and $u$ the number of uncovered edges, and keep the best-scoring candidate. The selected candidate is finalized by a greedy repair that repeatedly adds the vertex covering the most uncovered edges (ties broken by lowest index) until no uncovered edge remains. Because the repair only adds vertices, it cannot destroy the covering property, so the final output is a valid vertex cover by construction. We certify this per instance by enumerating all edges of each graph and asserting exactly zero uncovered edges. This verifies feasibility separately for every returned cover. Decoding applies to the methods that output continuous embeddings; the classical, local-search and exact solvers return discrete covers directly and are evaluated as returned.

Table 5: Hyperparameter configuration per dataset group. The RB instances use the longer 1000-epoch budget of the earlier baseline protocol. Loss, feature, and feedback coefficients are distinct. T1 coefficients below use the unhalved residual and count each undirected edge once. The loss coefficient, feature coefficient, and feedback step have separate definitions.

| **Parameter** | **Synthetic** | **RB200** | **RB500** | **TU-small** |
|---|---|---|---|---|
| Epochs | 200 | 1,000 | 1,000 | 200 (1,000 on ENZYMES) |
| Validation interval | 20–100 | 20–100 | 20–100 | 20–100 |
| Batch size | 16 | 32 | 16 | 2–40 |
| Embedding dimension $d$ | 32 | 64 | 64 | 32 |
| Layers $L$ | 16 | 16 | 16 | 16 |
| Optimizer | Adam | | | |
| Learning rate | $1 \times 10^{-3}$ (constant) | | | |
| Gradient clipping | global norm $5.0$ | | | |
| T1 loss coefficient $\beta_t$ | linear $1 \to 5$; code `penalty`: $0.5 \to 2.5$ | | | |
| T1 feature coefficient $\alpha_t$ | $0.25$ (constant) | | | |
| T1 feedback step $\eta_t$ | $a_t/(2N_b)$; stored state $\boldsymbol{\mu}^{\text{stored}} = 2\boldsymbol{\mu}$ | | | |
| Checkpoint selection | final epoch | | | |
| Dual scale $\gamma$ | 0.1 | 0.1 | 0.1 | 0.1 |
| Seeds | 5 | 2 | 2 | 5 |

Table 6: Run-to-run standard deviation (sd) of the repaired cover size for each cell of Table 1, computed over repeated independent training runs of that cell's protocol ($n-1$ denominator). Out-of-distribution $\pm$: Table 3.

| **Dataset** | **GIN** | | **GAT** | | **GCNN** | | **GatedGCNN** | | **OptGNN** | **ALON** |
|---|---|---|---|---|---|---|---|---|---|---|
| | Raw | +AL | Raw | +AL | Raw | +AL | Raw | +AL | | |
| *Synthetic Datasets* | | | | | | | | | | |
| ER [50,100] | 4.79 | 5.85 | 0.03 | 0.01 | 0.54 | 0.48 | 0.43 | 0.30 | 0.95 | 0.24 |
| BA [50,100] | 1.15 | 1.20 | 0.00 | 1.74 | 0.33 | 1.16 | 0.00 | 0.37 | 0.09 | 0.05 |
| WS [50,100] | 0.66 | 0.04 | 2.24 | 3.84 | 0.58 | 3.49 | 0.11 | 0.34 | 0.01 | 0.02 |
| HK [50,100] | 1.02 | 0.35 | 0.07 | 2.88 | 0.94 | 0.44 | 0.41 | 0.33 | 0.09 | 0.04 |
| ER [100,200] | 0.13 | 0.83 | 0.00 | 0.00 | 0.50 | 1.13 | 0.49 | 7.16 | 5.26 | 3.11 |
| BA [100,200] | 1.82 | 0.54 | 0.02 | 3.55 | 1.47 | 2.58 | 1.18 | 0.69 | 0.33 | 0.05 |
| WS [100,200] | 1.91 | 0.35 | 6.93 | 7.31 | 4.82 | 0.01 | 0.32 | 0.01 | 0.09 | 0.10 |
| HK [100,200] | 1.36 | 0.62 | 0.02 | 0.69 | 0.81 | 2.85 | 2.37 | 2.11 | 0.09 | 0.02 |
| ER [400,500] | 0.13 | 0.01 | 0.00 | 0.00 | 0.04 | 0.14 | 0.47 | 0.81 | 0.02 | 0.01 |
| BA [400,500] | 4.61 | 0.66 | 3.97 | 7.94 | 3.57 | 8.06 | 3.05 | 3.95 | 0.87 | 0.45 |
| WS [400,500] | 7.92 | 4.45 | 5.47 | 9.61 | 0.27 | 6.01 | 0.23 | 5.95 | 0.24 | 0.81 |
| HK [400,500] | 0.21 | 7.91 | 4.82 | 8.94 | 4.98 | 8.77 | 0.59 | 9.45 | 0.81 | 0.13 |
| *Real-world Datasets (TU-small)* | | | | | | | | | | |
| MUTAG | 0.67 | 0.07 | 0.18 | 0.22 | 0.00 | 0.04 | 0.04 | 0.00 | 0.00 | 0.00 |
| ENZYMES | 0.52 | 0.22 | 0.57 | 0.86 | 0.49 | 0.62 | 0.15 | 0.07 | 0.03 | 0.03 |
| PROTEINS | 0.36 | 0.33 | 1.41 | 2.98 | 0.03 | 0.07 | 0.03 | 0.21 | 0.00 | 0.01 |
| IMDB-BIN | 0.01 | 0.01 | 0.00 | 0.01 | 0.01 | 0.06 | 0.01 | 1.31 | 0.83 | 0.00 |
| COLLAB | 0.08 | 0.25 | 0.02 | 0.01 | 0.21 | 0.12 | 0.07 | 1.38 | 0.39 | 0.02 |
| *Hard RB Datasets* | | | | | | | | | | |
| RB200 | 0.05 | 0.03 | 0.00 | 0.00 | 0.09 | 0.06 | 0.03 | 0.30 | 0.34 | 0.09 |
| RB500 | 0.07 | 0.02 | 0.05 | 0.02 | 0.08 | 0.02 | 0.76 | 0.20 | 0.14 | 0.53 |

Table 7: Solver families used as classical baselines.

| Family | Solvers |
|---|---|
| Exact (branch-and-reduce) | Akiba–Iwata (PACE 2019 Track 1a) |
| Exact (MIP) | Gurobi (10 s limit) |
| Local search | NuMVC, FastVC, FastWVC |
| Approximation | Greedy 2-approx, Maximal Matching 2-approx |
| LP rounding | LP relaxation + rounding |

### C.3 Classical Baseline Details

To evaluate MVC solvers, we compare classical algorithms, local search heuristics, and an exact solver. Greedy builds a vertex cover by repeatedly selecting a vertex that covers the largest number of uncovered edges. Match. takes both endpoints of a maximal matching, whose union is a 2-approximation of the optimal cover. LP rounds an optimal solution of the vertex cover LP relaxation, also achieving a 2-approximation. NuMVC and FastVC are representative local search heuristics for MVC, and FastWVC is a local search heuristic for the weighted variant. Gurobi solves the integer programming formulation of MVC to optimality. Table 7 groups the eight solvers by family and Table 8 reports, for every dataset, the average vertex cover size over the full test split.

**Provenance and validation.** The exact branch-and-reduce solver is the vendored PACE 2019 Implementation Challenge Track 1a entry (GPL-3.0), patched only for portability and anytime output. It was validated against exhaustive search on 300 random small graphs and against Gurobi on 40 medium graphs, with zero mismatches. Every returned cover is additionally verified again feasible in Python.

**Protocol.** All solvers run on the same test split as the paper (an $80/10/10$ train/validation/test partition with a fixed seed) and a per graph wall clock cutoff of 10 s. Every returned cover is verified feasible, and local search receives the proven optimal size as an early stop bound on graphs where it is known.

**Findings.** Local search reaches the best-known references at millisecond scale on small and medium graphs, while the 2-approximation family (Greedy and Maximal Matching) trails by roughly 17–62% on the small and medium synthetic instances. The exact branch-and-reduce solver agrees with Gurobi wherever Gurobi certifies optimality. On ER [100,200] — the one dataset where the ten-second Gurobi budget does not suffice — it is at least as good on all but a single graph (idx 883, $n$=150, branch-and-reduce 124 against Gurobi 123). Classical solvers attain better cover quality on these small benchmarks; the comparison with ALON also examines inference cost and transfer across graph distributions.

### C.4 Detailed Reading of the Main Table

Table 1 shows gains from +AL across graph families. On synthetic graphs, the +AL variants improve their raw counterparts in the $[50, 100]$ and $[100, 200]$ ranges. At $[400, 500]$, the covers approach the all-vertex solution, and all three regressions in the backbone–dataset pair count occur at this scale. On the real-world TU datasets, ALON reaches the certified optimum on MUTAG and IMDB-BIN, stays within $1\%$ of it on PROTEINS, ENZYMES and COLLAB. On both Forced-RB instances, the neural methods return larger covers than the non-neural reference under the 10 s budget. ALON attains the best printed mean on 16 of the 19 datasets; the remaining rows are led by a +AL backbone on the hard RB instances (RB200/RB500) and by a raw backbone at the largest synthetic scale (ER$[400, 500]$, raw GatedGCNN). Appendix C.3 lists the solvers behind the **Reference** column.

**Relation to learned dual updates.** Learned primal–dual architectures (Hadou & Ribeiro, 2025; 2026) are relevant related approaches. ALON uses analytic projected node feedback on a factorized MVC relaxation with finite edge equalities and spherical variables. The relevant differences concern nonconvexity, feedback restrictions, supervision, and update architecture. Our OptGNN comparison assesses the architectural lineage, and the node-versus-edge experiment assesses feedback parameterization. These comparisons do not establish superiority over every learned primal–dual method.

Table 8: Classical baselines and ALON. Average vertex cover size over the full test split (lower is better, ↓). The Gurobi column is computed locally with Gurobi on the full test split. A–I is the branch-and-reduce reference. Bold marks the minimum value per row (ties included).

| Dataset | Greedy | Match. | LP | NuMVC | FastVC | FastWVC | A–I | Gurobi | ALON |
|---|---|---|---|---|---|---|---|---|---|
| *Synthetic Datasets* | | | | | | | | | |
| ER [50,100] | 67.79 | 70.22 | 59.41 | 53.58 | **53.52** | **53.52** | **53.52** | **53.52** | 55.28 |
| BA [50,100] | 50.67 | 58.18 | 48.21 | 42.17 | **42.05** | **42.05** | **42.05** | **42.05** | 42.13 |
| WS [50,100] | 72.19 | 73.28 | 54.85 | 45.54 | 45.47 | **45.46** | **45.46** | **45.46** | 45.60 |
| HK [50,100] | 52.79 | 60.85 | 50.35 | 44.04 | **43.96** | **43.96** | **43.96** | **43.96** | 44.04 |
| ER [100,200] | 143.66 | 146.26 | 133.21 | 122.71 | 122.66 | 122.66 | **122.57** | 122.90 | 137.30 |
| BA [100,200] | 102.55 | 118.25 | 97.24 | 84.25 | 84.18 | **84.17** | **84.17** | **84.17** | 84.49 |
| WS [100,200] | 143.79 | 145.41 | 108.36 | 89.66 | 89.66 | **89.63** | **89.63** | **89.63** | 90.43 |
| HK [100,200] | 101.58 | 117.97 | 97.07 | 84.75 | **84.66** | **84.66** | **84.66** | **84.66** | 85.04 |
| ER [400,500] | 442.75 | 445.18 | 429.44 | 409.90 | 409.89 | **409.88** | 414.34 | 414.10 | 449.80 |
| BA [400,500] | 302.73 | 351.07 | 288.48 | 246.94 | 246.92 | **246.89** | **246.89** | **246.89** | 250.22 |
| WS [400,500] | 437.09 | 441.26 | 329.23 | 271.21 | 271.30 | **271.19** | **271.19** | **271.19** | 278.01 |
| HK [400,500] | 298.81 | 347.92 | 285.04 | 247.02 | 246.95 | **246.94** | **246.94** | **246.94** | 249.09 |
| *Real-world Datasets (TU-small)* | | | | | | | | | |
| MUTAG | 12.84 | 15.26 | 8.16 | **7.74** | **7.74** | **7.74** | **7.74** | **7.74** | **7.74** |
| ENZYMES | 27.35 | 29.77 | 22.87 | **20.00** | 20.02 | **20.00** | **20.00** | **20.00** | 20.04 |
| PROTEINS | 33.93 | 37.07 | 28.75 | 24.98 | 25.01 | **24.96** | **24.96** | **24.96** | 25.09 |
| IMDB-BIN | 17.24 | 18.36 | **16.76** | **16.76** | **16.76** | **16.76** | **16.76** | **16.76** | **16.76** |
| COLLAB | 71.74 | 73.74 | 67.72 | **67.46** | **67.46** | **67.46** | **67.46** | **67.46** | 67.55 |
| *Hard RB Datasets* | | | | | | | | | |
| RB200 | 155.46 | 157.30 | 153.61 | 149.42 | 149.46 | 149.40 | 149.58 | **145.79** | 154.77 |
| RB500 | 494.35 | 498.71 | 489.57 | 477.73 | 478.04 | 477.75 | 484.90 | **467.93** | 491.98 |

**Variation across backbones and graph families.** On Forced-RB200, GatedGCNN+AL and GCNN+AL attain $150.10$ and $150.90$, compared with ALON's $154.77$ and OptGNN's $154.96$. At the largest synthetic scale, raw GatedGCNN leads on ER ($445.85$), while ALON leads on BA and WS. On ENZYMES the best neural entry is ALON ($20.04$), just ahead of GatedGCNN+AL ($20.05$). Both ALON and the +AL backbones use node feedback, so these comparisons do not isolate the effect of multiplier compression. Proposition 9 establishes a structural limitation of nonnegative node multipliers; the separate edge-dual experiment below evaluates their empirical effect under a specified budget.

**Edge level dual control.** To compare multiplier parameterizations, we trained an edge-level ALON variant with free per-edge multipliers $\nu_e$ on WS$[100, 200]$ and PROTEINS. Both arms use a 200-epoch budget, the paper's split and decoding protocol, and five dual schedules (Table 9); each cell reports the mean over three independent training seeds, so the entries average over run-to-run variation. The node arm has the smaller mean repaired-cover size in all ten dataset–schedule comparisons (direction $5/5$ on each dataset) while using fewer dual parameters; the edge arm has $1.9$–$2.0\times$ as many. The largest margins appear on WS$[100, 200]$ under S2 and S5 ($1.23$ and $1.59$ cover units), whereas the PROTEINS gaps ($0.03$–$0.18$ cover units) are small in absolute terms, so the ten-for-ten direction is a consistent ordering rather than a uniform, individually resolved improvement. Both arms outperform the raw GIN fixed-penalty baseline in this control. The schedules specify $(\mathrm{freq}, \rho_{\mathrm{init}}, \tau, \rho_{\max})$, with frequency measured in epochs: S1 $(1, 0.02, 2.0, 0.5)$, S2 $(5, 0.02, 2.0, 0.5)$, S3 $(25, 0.02, 2.0, 0.5)$, S4 $(5, 0.1, 2.0, 2.0)$, and S5 $(1, 0.1, 4.0, 2.0)$. S1 uses the once-per-epoch update frequency of Algorithm 1. In this control, $\rho$ starts at $\rho_{\mathrm{init}}$ and is capped at $\rho_{\max}$ when increased by factor $\tau$; the node and edge arms share each schedule.

Table 9: Edge level dual control. Mean repaired cover size (lower is better, ↓) under the paper's split, a 200 epoch budget, and the paper's decoding protocol (randomized hyperplane rounding followed by greedy repair); entries are means over three training seeds, and S1–S5 index the five dual schedules. **Node** is the paper's node level parameterization ($\lambda_{ij} = \mu_i + \mu_j$); **Edge** uses free per edge multipliers $\nu_e$ (1.9–2.0× the dual parameters); **Fixed** is the raw GIN fixed penalty baseline rather than a penalty only arm of the same architecture. The node–edge differences should be read against the run-to-run spread of the repaired cover on these datasets, which under this protocol is of order $10^{-2}$ cover units on PROTEINS and $10^{-1}$ on WS$[100, 200]$ (five-seed sd of the ALON arm on the same datasets; Table 6). Bold marks the smaller mean of the two dual arms per schedule and dataset.

| Schedule | WS$[100, 200]$ | | | PROTEINS | | |
|---|---|---|---|---|---|---|
| | Node | Edge | Fixed | Node | Edge | Fixed |
| S1 | **90.70** | 91.44 | 101.82 | **25.04** | 25.07 | 32.22 |
| S2 | **91.85** | 93.08 | 101.82 | **25.07** | 25.13 | 32.22 |
| S3 | **92.10** | 92.25 | 101.82 | **25.06** | 25.24 | 32.22 |
| S4 | **91.46** | 92.11 | 101.82 | **25.06** | 25.11 | 32.22 |
| S5 | **91.06** | 92.64 | 101.82 | **25.02** | 25.07 | 32.22 |

### C.5 Time/Quality Trade-off

Figure 5 compares the reported cover sizes and per-graph times of classical and neural solvers across all 19 datasets. The axes use mean cover size and logarithmic wall-clock time. The direct relaxation solver is compared separately in Table 12.

**Findings.** The plotted methods span different quality and runtime regimes. On the small TU datasets, classical local search is competitive in both cover size and time. The neural points show the cost of amortized prediction and decoding. Table 12 and the timing comparison in Appendix C.8 separately assess amortization against direct optimization of the continuous relaxation.

### C.6 Extension to Weighted Objectives

**Setup.** The extension retains the network architecture and covering constraints, and uses a vertex-weighted objective with the weighted decoder specified below. Vertex weights are integers drawn uniformly from $\{1, \ldots, 10\}$ with the dataset seed, so they span one order of magnitude. Rounding draws an anchor-threshold candidate together with $64$ random hyperplanes under a fixed seed, and the greedy weighted repair adds the vertex with the largest ratio of newly covered edges to weight before pruning redundant vertices in decreasing weight order. Every reported constrained solution is checked edge by edge (violation $= 0$). The reference is computed by Gurobi, which certifies optimality on 100% of the weighted-MVC test graphs and agrees with the FastWVC heuristic.

**Dual activity.** On the weighted-MVC extension the dual is demonstrably active during training. The multipliers live on the training vertices, where the per-node potential is accumulated over every batch, and the resulting ALM term shapes the parameters $\theta$ (see the mechanism analysis in the main text). Consistent with this, inference uses $\mu = \mathbf{0}$ in a single forward pass: the dual information is distilled into $\theta$, so no multiplier vector is stored or evaluated at test time.

**Reading.** The corresponding numbers are reported in Table 2 of the main text; ALON attains the smaller mean cover cost on all four families, with the largest absolute margin over the fixed-penalty baseline on the synthetic families. The extension retains the architecture and uses a weighted objective and weighted decoding. Each returned cover is verified edge by edge.

### C.7 Time and Memory Complexity

Table 10 gives the per graph cost of each ALON component (one inference pass, $L$ layers, embedding dimension $d$, graph $(\mathcal{V}, \mathcal{E})$).

Dual parameters are node level ($\mu \in \mathbb{R}^{|\mathcal{V}|}$), giving $\mathcal{O}(|\mathcal{V}|)$ dual-state memory versus $\mathcal{O}(|\mathcal{E}|)$ for edge level ALM. This is the dual-state compression quantified in Table 4. Total-memory scaling also includes the per-layer gradient buffers. The retained per-layer constraint-gradient buffers are

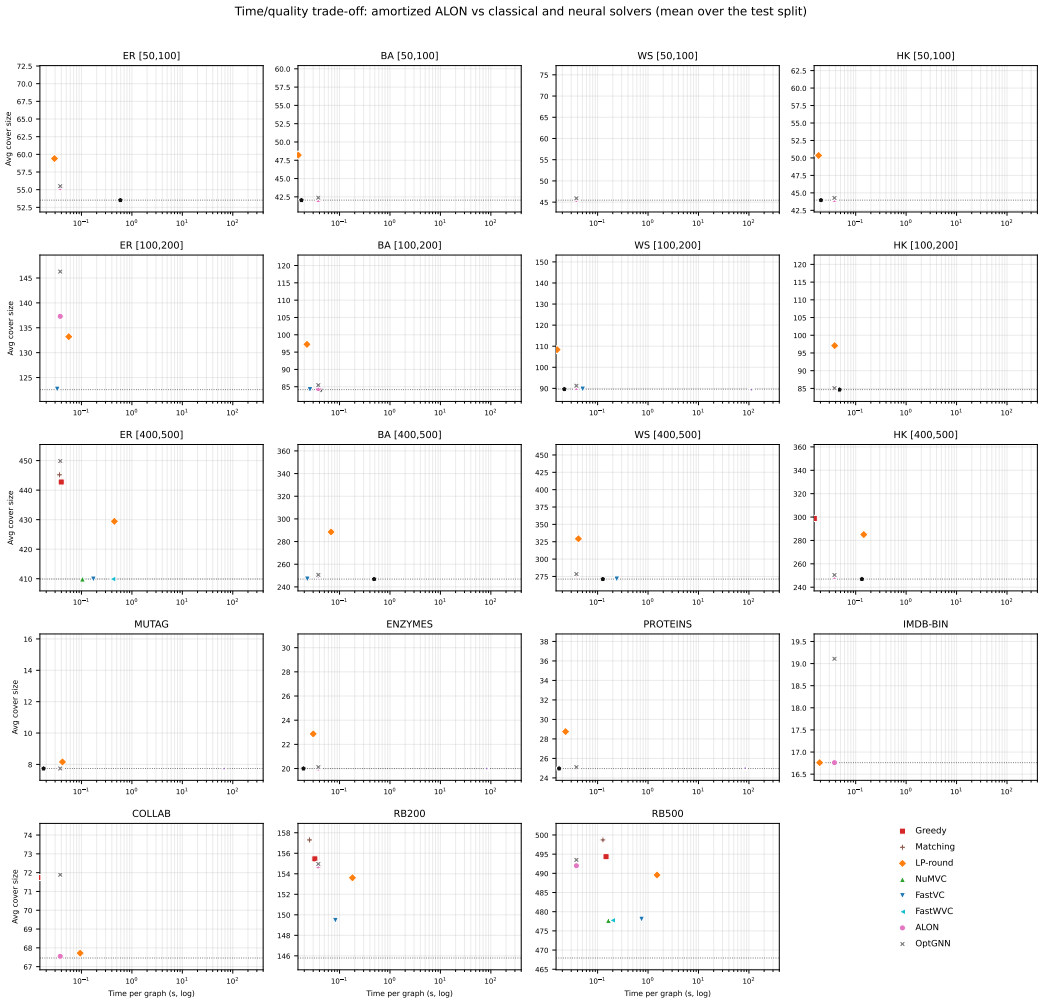


Figure 5: **Time/quality comparison.** Mean cover size versus wall-clock time per graph on a logarithmic axis, with one panel per dataset. Markers identify the classical and neural solvers listed in the legend. Gurobi-10s denotes the incumbent under the reported 10 s budget; neural inference includes the forward pass and rounding.

Table 10: **Complexity of ALON components** (per graph; the projected dual update is used during training). Message passing and gradient computations dominate. Rounding is a single $\mathcal{O}(1000\,|\mathcal{V}|d)$ matmul. Dual state is $\mathcal{O}(|\mathcal{V}|)$, while total memory is dominated by the retained per-layer constraint-gradient buffers.

| **Component** | **Time** | **Memory** |
|---|---|---|
| Objective gradient $\nabla f$ (per layer) | $\mathcal{O}(\lvert\mathcal{V}\rvert d)$ | $\mathcal{O}(\lvert\mathcal{V}\rvert d)$ |
| Constraint gradients $\nabla g_{ij}$, $\nabla g_{ij}^2$ (per layer) | $\mathcal{O}(\lvert\mathcal{E}\rvert d)$ | $\mathcal{O}(\lvert\mathcal{E}\rvert d)$ |
| Node level scatter aggregation (per layer) | $\mathcal{O}(\lvert\mathcal{E}\rvert)$ | $\mathcal{O}(\lvert\mathcal{V}\rvert)$ |
| Linear fusion $[x; g_{\text{obj}}; g_{\text{dual}}] \to x'$ (per layer) | $\mathcal{O}(\lvert\mathcal{V}\rvert d^2)$ | $\mathcal{O}(d^2)$ |
| **Total forward** ($L$ layers) | $\mathcal{O}\big(L(\lvert\mathcal{E}\rvert d + \lvert\mathcal{V}\rvert d^2)\big)$ | $\mathcal{O}(L\lvert\mathcal{E}\rvert d + \lvert\mathcal{V}\rvert d)$ |
| Projected node update (given $\boldsymbol{c}$) | $\mathcal{O}(\lvert\mathcal{V}\rvert)$ per update | $\mathcal{O}(\lvert\mathcal{V}\rvert)$ |
| Rounding (1000 hyperplanes) | $\mathcal{O}(1000\,\lvert\mathcal{V}\rvert d)$ | $\mathcal{O}(1000\,\lvert\mathcal{V}\rvert)$ |

$\mathcal{O}(|\mathcal{E}|d)$ (Table 10) and dominate, and inference stores no dual state ($\boldsymbol{\mu} = \mathbf{0}$), so the compression is a training-time saving. Inference is a single forward pass with no inner loop iterations, so the *runtime* complexity is that of $L$ sparse message-passing layers, the same order as the backbone GNNs (GIN/GAT/GCNN) surveyed in the main table, plus one gradient evaluation per layer. The gradient evaluations are obtained by automatic differentiation with a retained computation graph. This constant factor dominates our implementation, which is visible in the measured per graph latency (Table 11).

**Training cost.** Training adds one backward pass through the same $L$ layers per gradient step, plus a per epoch dual ascent update. The projected node update takes $\mathcal{O}(|\mathcal{V}|)$ time once the incident edge residuals have been aggregated; forming that aggregate takes $\mathcal{O}(|\mathcal{E}|)$ time. The per-step training cost is $\mathcal{O}\big(L(|\mathcal{E}|d + |\mathcal{V}|d^2)\big)$, identical in order to a standard $L$-layer GNN trained with a penalty loss. There is no inner solver loop. The primal-dual dynamics are unrolled in the forward pass and differentiated through, so training cost per step does not scale with an iteration budget.

**Measured dual update overhead.** On an A100 a single dual ascent update costs $179.0\,\mu$s at $|\mathcal{V}| = 43{,}471$ and $189.3\,\mu$s at $|\mathcal{V}| = 298{,}685$. A $6.9\times$ increase in $|\mathcal{V}|$ therefore costs only $1.06\times$ more time, indicating that the update is bound by kernel launch and synchronization latency rather than bandwidth. At one update per epoch this amounts to $0.001\%$ of an epoch, and paired 200-epoch runs with and without per epoch dual ascent differ by $+0.061\%$ (3640.6 s vs. 3638.4 s). Inference requires no dual state at all. The reported solutions use $\boldsymbol{\mu} = \mathbf{0}$ in a single forward pass, since the dual information is distilled into $\theta$ during training.

**Node level vs. edge level duals: time, not just memory.** Node parameterization reduces the dual state from $\mathcal{O}(|\mathcal{E}|)$ to $\mathcal{O}(|\mathcal{V}|)$. Updating the stored state is linear in the number of nodes after edge residual aggregation. On the RB500 instances ($|\mathcal{E}|/|\mathcal{V}| \approx 89.7$) the edge level dual state alone would occupy 172.3 MB per the memory study, whereas the node level state is 1.9 MB. In wall clock terms the dual update is a negligible fraction of a training step (a scatter add against $L$ message-passing layers), so the practical gain of the node level parameterization is a smaller dual-state footprint, not a constant-factor speedup of the gradient computation. Total memory remains dominated by the $\mathcal{O}(|\mathcal{E}|d)$ constraint-gradient computation even on dense graphs.

**Measured scaling.** We validate the analysis on 3,964 test graphs spanning $|\mathcal{E}| \in [1, 36{,}728]$ (Table 11).

Table 11: **Measured inference time vs. graph size** (ALON forward pass, $L$=16, $d$=32, A100, median over test graphs). Time is essentially flat across two orders of magnitude of $|\mathcal{E}|$. The per layer autograd graph construction is a fixed $\approx 89$ ms cost, while the $\mathcal{O}(L|\mathcal{E}|d)$ compute term contributes $\approx 0.26\,\mu$s per edge ($\approx 8$ ms at $|\mathcal{E}| = 3 \times 10^4$).

| $\lvert\mathcal{E}\rvert$ range | graphs | median $\lvert\mathcal{V}\rvert$ | median forward (ms) |
|---|---|---|---|
| $< 100$ | 1020 | 21 | 89.1 |
| 100–500 | 1263 | 46 | 90.6 |
| 500–2000 | 1245 | 85 | 88.0 |
| 2000–$10^4$ | 332 | 118 | 92.8 |
| $> 10^4$ | 104 | 281 | 101.2 |

Across the measured edge-count groups, median inference times range from 88.0 to 101.2 ms. The weak variation over this range is consistent with the substantial per-layer autograd overhead in the current implementation. The asymptotic edge-dependent term becomes more relevant as graph size grows.

### C.8 Impact of the Relaxation and Amortization

**Setup.** ALON predicts a unit-row embedding, which is decoded into a cover. We compare it with a direct augmented Lagrangian solve of the same relaxation using $K = 8$ random initializations per graph. Both outputs, and the exactly solved LP relaxation, use the same decoding procedure: 1000 randomized hyperplanes followed by greedy repair and an edge-by-edge feasibility check. The exact combinatorial optimum is computed by branch-and-reduce. The direct solver provides an empirical reference for cover quality after optimization and decoding.

Table 12: Final cover size (mean) on the test splits of the main protocol. The direct solver, ALON, and LP-relaxation outputs use the same randomized rounding, greedy repair, and per-edge feasibility verification. The direct relaxation solver is the same continuous relaxation solved directly by an augmented Lagrangian method. The exact optimum is certified by branch-and-reduce. ALON uses the configuration of Table 1 (200 training epochs on MUTAG, PROTEINS, and WS$[100, 200]$; 1000 on ENZYMES), and for these stochastic solver outputs, $\pm$ denotes the run-to-run standard deviation over repeated independent runs. Timing was measured with both solvers under identical conditions (see the Amortization paragraph).

| Dataset | Exact optimum | Direct solver | ALON | LP relaxation |
|---|---|---|---|---|
| MUTAG | $7.74 \pm 0.00$ | $7.74 \pm 0.00$ | $7.74 \pm 0.00$ | $8.16 \pm 0.73$ |
| ENZYMES | $20.00 \pm 0.00$ | $20.00 \pm 0.00$ | $20.04 \pm 0.03$ | $22.87 \pm 0.30$ |
| PROTEINS | $24.96 \pm 0.00$ | $24.99 \pm 0.04$ | $25.09 \pm 0.01$ | $28.75 \pm 2.44$ |
| WS$[100, 200]$ | $89.63 \pm 0.00$ | $90.36 \pm 0.30$ | $90.43 \pm 0.10$ | $108.36 \pm 0.56$ |

**Reading.** Table 12 compares repaired covers from direct and amortized optimization. ALON matches the direct solver on MUTAG, is $+0.04$ above it on ENZYMES, $+0.10$ on PROTEINS, and $+0.07$ on WS$[100, 200]$, $+0.80$ above the exact optimum. These results compare the combined effect of continuous optimization and decoding under the reported configurations. Proposition 7 complements this empirical comparison with a continuous-output certificate.

**Amortization.** Under identical conditions (same machine, CPU-only, the same pool of 12 single-threaded workers, no exclusivity granted to either), on the full 200-graph WS$[100, 200]$ test split one direct augmented Lagrangian solve with $K{=}8$ initializations takes $111.40$ s per graph (median $111.56$, sd $8.39$, $n{=}200$), whereas one ALON forward pass plus rounding takes $39.11$ ms per graph ($29.14$ ms forward + $9.97$ ms rounding, sd $4.85$), a $2848.6\times$ speedup at the cover qualities of Table 12. The factor is the ratio of the full $K{=}8$-restart direct-solve budget to one amortized forward pass plus rounding, not a per-iteration comparison. Because the direct solve is per instance while the model is trained once, the amortized approach breaks even after $N^*{=}2.8$, $12.0$, $23.5$ and $43.3$ new graphs on MUTAG, PROTEINS, WS$[100, 200]$ and ENZYMES respectively, where $N^* = T_{\text{train}}/(t_{\text{direct}} - t_{\text{infer}})$; beyond that the training cost is amortized, while each new graph still incurs inference and decoding cost.

**Timing scope.** The direct-solver timer covers continuous optimization and excludes decoding; the ALON timer includes forward computation, randomized rounding, and repair. The ratio compares these recorded scopes and is not a measurement of identical end-to-end pipelines.

**Where the advantage does not hold.** The $2848.6\times$ factor compares ALON with the *direct solver of the continuous relaxation*, not with fast classical heuristics. On the same instances the local-search heuristics NuMVC and FastVC reach ALON's repaired-cover quality within $\leq 38$ ms on all 19 datasets (FastWVC on 18), so there is no time-to-quality advantage over them; their mean covers are equal or smaller on 18 of 19 datasets. ALON's advantage is therefore confined to the regime where the alternative is a per-instance solve of the relaxation, and it beats the LP relaxation on both axes (Table 12). It is not competitive with NuMVC/FastVC on time-to-quality at any tested size.

The $39.11$ ms here and the $88.0$–$101.2$ ms medians of Table 11 are different regimes and are not contradictory: the former is inference-mode CPU forward plus rounding on the WS$[100, 200]$ set under the 12-worker pool, whereas the latter is batch-of-one A100 forward latency over the $3{,}964$-graph scaling sweep, dominated by a fixed $\approx$89 ms per-layer autograd-construction cost rather than by the inference path.