# GAPCYCLE: LEARNING FROM RESIDUAL CYCLES FOR MINIMUM-COST FLOW

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

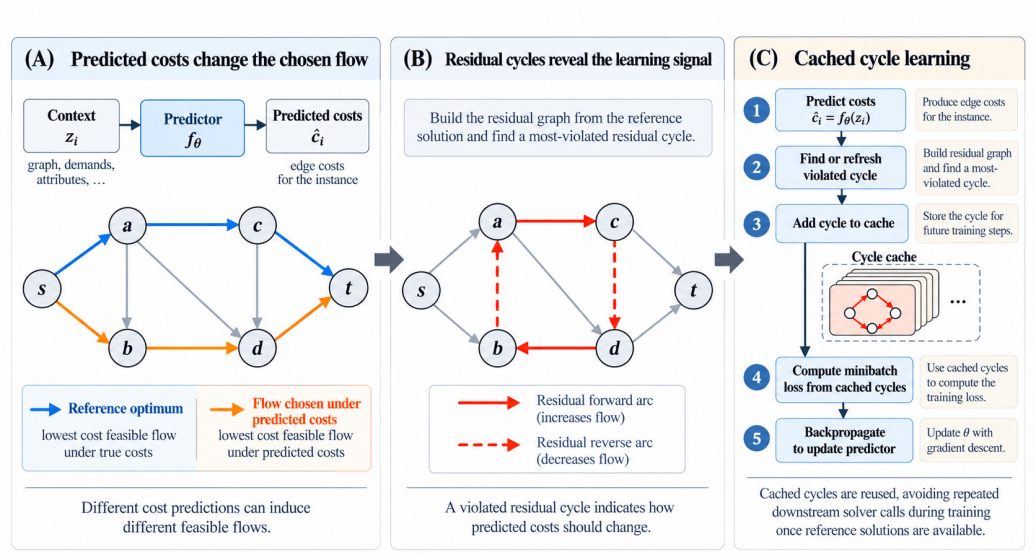



Figure 1: GapCycle intuition. Predicted costs $\hat{c}_i$ can induce a flow different from the true-cost reference $x_i^\star$. Its residual graph exposes feasible circulation adjustments, and loss-augmented minimum-mean-cycle separation identifies a most-violated cycle when invoked. A bounded working set reuses separated cycles between refreshes; in the full-information model, the cached cycle hinge is combined with normalized mean-squared error (nMSE) to update the predictor without downstream MCF solver calls once the reference optima are available.

## ABSTRACT

Decision-focused learning aims to predict costs that induce good downstream decisions, but training can require repeated solves of the downstream optimization problem. For minimum-cost flow (MCF), GapCycle instead exploits the classical residual-cycle optimality condition: an optimal flow has no improving residual cycle. GapCycle defines a length-normalized hinge over residual cycles and penalizes the cycle that most violates this condition under the predicted costs. When separation is invoked, the most violated cycle is found exactly by a minimum-mean-cycle computation; a bounded working set then reuses discovered cycles between separator calls. This removes downstream MCF solves from the training signal while retaining efficient graph-based separation. For the full, uncached cycle hinge, we prove a pointwise upper bound on decision regret; for a positive margin parameter $\kappa$, zero full hinge implies zero regret for any flow that is optimal under the predicted costs. On a held-out 24-topology NETGEN panel, GapCycle achieves aggregate normalized regret of $0.020800$, the lowest among the evaluated methods, compared with $0.023338$ for the next-lowest method, MOM, while using zero downstream MCF training solves. Additional shortest-path experiments test robustness across misspecification and noise. Together, these results show how MCF structure can replace repeated downstream solves with a decision-aware training signal.



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.

# 1 INTRODUCTION

In contextual minimum-cost flow, a predictive model maps features to arc costs and an optimizer converts those costs into a flow. Prediction error is only an indirect objective: similar cost errors can induce different optimal flows and substantially different downstream regret. Decision-focused learning (DFL) addresses this mismatch by training predictions according to the quality of the decisions they induce (Elmachtoub & Grigas, 2022).

A central computational challenge is that many DFL methods obtain their training signal through repeated optimization calls, including SPO+ and perturbed-optimizer approaches (Elmachtoub & Grigas, 2022; Berthet et al., 2020). Recent methods show that avoiding such calls is possible using optimality conditions, geometric structure, or precomputed solution information (Sun et al., 2023; Berden et al., 2026; Wan & Liu, 2026). This raises a more specific question for minimum-cost flow: can its graph structure provide a direct decision-aware training signal without repeatedly solving the downstream flow problem?

GapCycle builds on classical residual-cycle and inverse-flow geometry. Residual cycles characterize feasible circulation adjustments, and a minimum-cost flow is optimal exactly when its residual network contains no improving directed cycle (Ahuja et al., 1993). In inverse optimization, Ahuja and Orlin showed that the unit-weight $\ell_\infty$ inverse minimum-cost-flow problem reduces to a minimum-mean-cycle computation (Ahuja & Orlin, 2001; 2002). GapCycle uses this classical geometry for a different purpose: rather than repairing one fixed cost vector so that a prescribed flow becomes optimal, it turns residual cycle-mean violations into a contextual decision-focused training loss. The loss incorporates the realized true costs through a margin, is evaluated repeatedly as the predictor changes, and is coupled with a bounded working set and a pointwise decision-regret analysis.

For the full hinge, we prove a pointwise decision-regret bound. On the held-out NETGEN panel, GapCycle achieves aggregate normalized regret $0.0208$, compared with $0.0233$ for MOM, while making no downstream minimum-cost-flow calls in its gradient-based training signal once reference flows are available. We also evaluate computational cost and shortest-path robustness. The contributions are: (i) a contextual, true-cost-aware residual-cycle margin for MCF that extends classical inverse-flow geometry to decision-focused learning; (ii) bounded working-set training together with a certificate connecting the cached and full hinges and pointwise regret control; and (iii) evaluation under held-out topologies, matched supervision, computational scaling, and complementary network-flow settings.

# 2 RELATED WORK

**Decision-focused learning.** DFL trains predictions for downstream decision quality rather than predictive accuracy alone. SPO+ provides a convex surrogate for predict-then-optimize problems, while differentiable optimization layers and perturbed optimizers construct gradients through or around the downstream optimization map (Elmachtoub & Grigas, 2022; Amos & Kolter, 2017; Agrawal et al., 2019; Berthet et al., 2020; Blondel et al., 2020). Contrastive and learning-to-rank approaches instead learn from subsets of feasible decisions, reducing dependence on repeated solves through cached or sampled alternatives (Mulamba et al., 2021; Mandi et al., 2022).

**Optimality-based and solver-free training.** Several methods avoid repeated downstream optimization using structural information. MOM derives a max-margin objective from LP optimality conditions (Sun et al., 2023); LAVA compares a reference optimum with precomputed adjacent vertices (Berden et al., 2026); and WISE obtains solver-free training for linear predict-then-optimize problems through a measure transformation (Wan & Liu, 2026). GapCycle is specialized instead to minimum-cost flow. It uses the classical negative-residual-cycle optimality condition and minimum-mean-cycle computation (Ahuja et al., 1993; Karp, 1978) to define and exactly separate a length-normalized residual-cycle hinge. Its distinction is therefore not solver-free training alone, but the combination of an MCF-specific residual-cycle surrogate, exact graph-based separation of the full hinge, a bounded working set, and pointwise regret control.

**Inverse optimization and network flows.** Classical inverse optimization asks for a minimum perturbation of an objective that makes a prescribed feasible solution optimal. For minimum-cost flow,

Ahuja and Orlin showed that the unit-weight $\ell_\infty$ inverse problem reduces to minimum-mean-cycle computation, with related combinatorial algorithms developed specifically for inverse network-flow problems (Ahuja & Orlin, 2001; 2002). Zhang and Liu developed a more general $\ell_\infty$ framework for inverse combinatorial optimization (Zhang & Liu, 2002). More recently, MOM connects optimality-margin learning with contextual and inverse linear programming (Sun et al., 2023). GapCycle does not claim the residual-cycle or minimum-mean-cycle geometry itself as new. Its contribution is to use this geometry inside a contextual decision-focused surrogate with a true-cost-aware margin, bounded working-set training, and decision-regret control.

# 3 GapCycle

## 3.1 Problem Setting: Minimum-Cost Flow and Decision Regret

Consider a minimum-cost-flow instance on a directed graph $G = (V, E)$ with $m = |E|$ arcs. Let $A \in \mathbb{R}^{|V|\times m}$ be the node–arc incidence matrix, $\beta \in \mathbb{R}^{|V|}$ the node supply/demand vector, and $l, u \in \mathbb{R}^m$ the arc lower and upper bounds. Each context has dimension $p$. We assume the feasible set

$$\mathcal{X} = \{x \in \mathbb{R}^m : Ax = \beta,\ l \le x \le u\} \neq \varnothing. \tag{1}$$

Each instance provides context $z \in \mathbb{R}^p$ and true arc costs $c \in \mathbb{R}^m$. A predictor $f_\theta$, with trainable parameters $\theta$, produces $\hat{c} = f_\theta(z)$, inducing

$$x(\hat{c}) \in \arg\min_{x\in\mathcal{X}} \hat{c}^\top x, \qquad x^\star \in \arg\min_{x\in\mathcal{X}} c^\top x. \tag{2}$$

Here $x^\star$ is a reference true-cost optimum. Neither optimization problem is assumed to have a unique optimum: $x(\hat{c})$ denotes the selected predicted-cost optimizer, while the guarantee developed later applies to every predicted-cost optimizer.

Following the predict-then-optimize setting (Elmachtoub & Grigas, 2022), decision regret is

$$R(\hat{c}, c) = c^\top x(\hat{c}) - c^\top x^\star \ge 0, \tag{3}$$

where the true costs evaluate both decisions. With $i$ indexing evaluation instances, our primary metric is aggregate normalized regret,

$$R_{\text{agg}} = \frac{\sum_i R(\hat{c}_i, c_i)}{\sum_i |c_i^\top x_i^\star|}, \qquad R_{\text{PI}}(\hat{c}, c) = \frac{R(\hat{c}, c)}{\max\{\varepsilon_R, |c^\top x^\star|\}}, \quad \varepsilon_R = 1. \tag{4}$$

We assume the aggregate denominator is positive. $R_{\text{agg}}$ is our primary benchmark metric and matches the normalized-regret convention used by PyEPO (Tang & Khalil, 2022); $R_{\text{PI}}$ is retained as a per-instance sensitivity measure. These quantities are evaluation metrics, not training objectives. In the full-information setting, training provides $(z_i, c_i, x_i^\star)$; we later also consider a decision-only variant that does not use $c_i$ during training.

Figure 2 summarizes the full-information GapCycle training pipeline; the following sections define its residual-cycle surrogate, separator, and bounded working set.

## 3.2 Residual-Cycle Optimality

Fix the reference optimum $x^\star$. For an original arc $e = (v, w)$, define its forward and reverse residual capacities as

$$\rho_e^+ = u_e - x_e^\star, \qquad \rho_e^- = x_e^\star - l_e. \tag{5}$$

The residual graph $\mathcal{R}(x^\star)$ contains a forward copy $e^+ = (v, w)$ whenever $\rho_e^+ > 0$ and a reverse copy $e^- = (w, v)$ whenever $\rho_e^- > 0$. Write $E_R(x^\star)$ for its residual-arc set. The forward copy represents increasing $x_e$, whereas the reverse copy represents decreasing it. We retain the identity and orientation of every residual copy, so parallel or antiparallel original arcs remain distinct.

For a generic cost vector $q' \in \mathbb{R}^m$, let $\sigma_r = +1$ for a forward residual copy and $\sigma_r = -1$ for a reverse copy. A residual arc $r$ associated with original arc $e(r)$ therefore has cost $\sigma_r q'_{e(r)}$. For a directed residual cycle $C$, define its signed direction in original-arc coordinates by

$$(a_C)_e = \sum_{\substack{r\in C \\ e(r)=e}} \sigma_r. \tag{6}$$

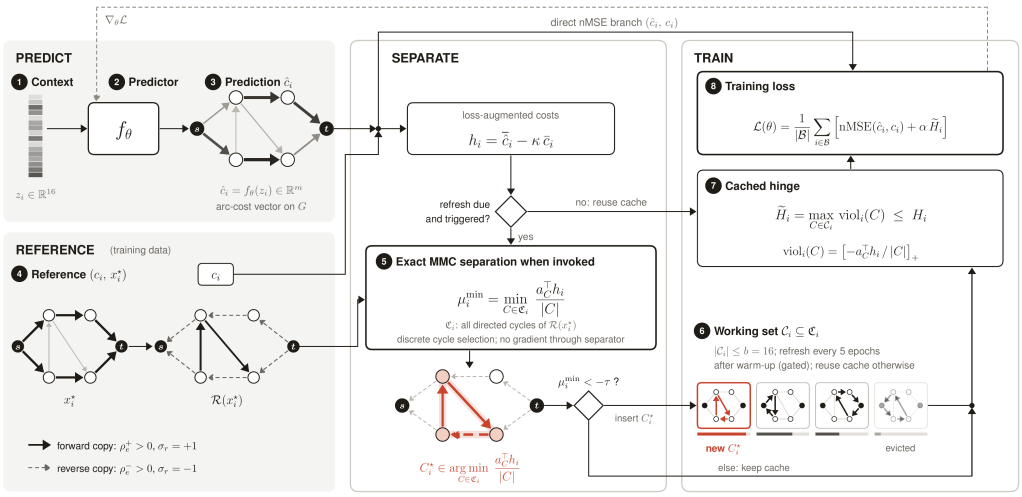



Figure 2: GapCycle training in the full-information setting. Given $(z_i, c_i, x_i^\star)$, the predictor produces $\hat{c}_i$ and GapCycle forms loss-augmented weights on the residual graph of $x_i^\star$. When invoked, exact minimum-mean-cycle separation adds a most-violated cycle to a bounded working set; between refreshes, the cached hinge $\widetilde{H}_i \le H_i$ is combined with normalized mean-squared error (nMSE) to update $f_\theta$. Cycle selection is discrete, and, given reference optima, the gradient-based training signal uses no downstream MCF solver call.

Since $C$ is a circulation, $Aa_C = 0$. Let $\rho_r$ denote the residual capacity of copy $r$ and $\delta_C = \min_{r \in C} \rho_r$. Then, for $0 \le \delta \le \delta_C$,

$$x^\star + \delta a_C \in \mathcal{X}, \qquad q'^\top (x^\star + \delta a_C) - q'^\top x^\star = \delta\, a_C^\top q' = \delta \sum_{r \in C} \sigma_r q'_{e(r)}. \tag{7}$$

Thus $a_C^\top q' < 0$ identifies a feasible improving circulation under costs $q'$.

Residual cycles are also sufficient for global optimality. For any $x \in \mathcal{X}$, let $d = x - x^\star$. Routing $[d_e]_+$ on the forward residual copy of $e$ and $[-d_e]_+$ on its reverse copy gives a feasible circulation in $\mathcal{R}(x^\star)$. If $d \neq 0$, classical circulation decomposition yields

$$d = \sum_{j=1}^{K} \lambda_j a_{C_j}, \qquad \lambda_j > 0, \tag{8}$$

for directed residual cycles $C_j$; $d = 0$ corresponds to the empty decomposition (Ahuja et al., 1993). Consequently, letting $\mathcal{E}(x^\star)$ denote all directed cycles in $\mathcal{R}(x^\star)$,

$$x^\star \in \arg\min_{x \in \mathcal{X}} q'^\top x \quad \Longleftrightarrow \quad a_C^\top q' \ge 0 \quad \forall C \in \mathcal{E}(x^\star). \tag{9}$$

This is the classical negative-cycle optimality condition for minimum-cost flow (Ahuja et al., 1993). In particular,

$$a_C^\top c \ge 0 \qquad \forall C \in \mathcal{E}(x^\star). \tag{10}$$

Hence each cycle describes a feasible adjustment, while the complete family of residual-cycle inequalities provides a global certificate for preserving the optimality of $x^\star$.

### 3.3 Cycle-Margin Loss

For training instance $i$, write $\mathcal{E}_i = \mathcal{E}(x_i^\star)$. We first apply the stabilized normalization

$$\bar{v} = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon_c^2}}, \qquad \varepsilon_c = 10^{-8}. \tag{11}$$

For each fixed $v$, this transformation is multiplication by a positive scalar and therefore preserves the set of minimizers induced by that objective, while reducing direct dependence of the surrogate on its raw magnitude.

By equation 9, preserving $x_i^\star$ as a predicted-cost optimum is equivalent to requiring $a_C^\top \bar{\hat{c}}_i \geq 0$ for every $C \in \mathcal{E}_i$. GapCycle strengthens this condition to the cost-dependent margin

$$a_C^\top \bar{\hat{c}}_i \geq \kappa\, a_C^\top \bar{c}_i, \qquad C \in \mathcal{E}_i, \tag{12}$$

where $\kappa \geq 0$ controls the margin strength; the main configuration uses $\kappa = \frac{1}{2}$. A direction with larger normalized true-cost increase receives a larger target margin. If $a_C^\top c_i = 0$, its additional margin is zero, so true-cost-neutral directions are not assigned an artificial strict preference. Setting $\kappa = 0$ recovers the classical predicted-cost optimality condition.

For a cycle $C$, define

$$\mathrm{viol}_i(C) = \left[\frac{\kappa a_C^\top \bar{c}_i - a_C^\top \bar{\hat{c}}_i}{|C|}\right]_+, \qquad [t]_+ = \max\{t, 0\}, \tag{13}$$

where $|C|$ is the number of labeled residual arcs in the cycle. Dividing by $|C|$ is a modeling choice: GapCycle compares mean violation per residual arc rather than total accumulated violation. This normalization will yield minimum-mean-cycle separation and will also match the cycle-length accounting used by the later regret analysis.

Define

$$h_i = \bar{\hat{c}}_i - \kappa \bar{c}_i. \tag{14}$$

The full per-instance cycle hinge is

$$H_i(\hat{c}_i) = \max_{C \in \mathcal{E}_i} \mathrm{viol}_i(C) = \max_{C \in \mathcal{E}_i} \left[-\frac{a_C^\top h_i}{|C|}\right]_+, \tag{15}$$

with the convention $H_i = 0$ when $\mathcal{E}_i = \varnothing$. Thus $H_i$ measures the largest normalized residual-cycle margin violation. For $\kappa = 0$, $H_i = 0$ is exactly the certificate that $x_i^\star$ remains optimal under the predicted costs; the stronger regret implication for $\kappa > 0$ is established later.

**Relation to inverse minimum-cost flow.** A classical inverse MCF problem starts from a feasible reference flow $x^\star$ and a fixed cost vector $c^0$ and seeks

$$\min_d \|d - c^0\|_\infty \quad \text{s.t.} \quad x^\star \in \arg\min_{x \in \mathcal{X}} d^\top x.$$

For unit weights, this problem has a minimum-mean-cycle characterization (Ahuja & Orlin, 2001; 2002). GapCycle does not solve this static cost-repair problem. Instead, the predictor produces a context-dependent vector

$$h_i = \bar{\hat{c}}_i - \kappa \bar{c}_i,$$

and the negative minimum residual cycle mean becomes the training hinge

$$H_i = [-\mu_i^{\min}(h_i)]_+.$$

Thus the cycle-mean geometry is classical; GapCycle uses it as a differentiable, repeatedly evaluated decision-focused margin whose target changes with the learned predictor and whose violation is connected below to downstream regret.

### 3.4 Exact Cycle Separation

Applying this classical cycle-mean geometry to the GapCycle margin gives an exact separator for the full hinge. For the current margin vector $h_i$, assign each labeled residual arc $r$ the weight

$$\omega_i(r) = \sigma_r h_{i,e(r)}. \tag{16}$$

Therefore every $C \in \mathcal{E}_i$ has total and mean separator weights

$$\sum_{r \in C} \omega_i(r) = a_C^\top h_i, \qquad \mu_i(C) = \frac{1}{|C|} \sum_{r \in C} \omega_i(r) = \frac{a_C^\top h_i}{|C|}. \tag{17}$$

For $\mathcal{E}_i \neq \varnothing$, let

$$\mu_i^{\min} = \min_{C \in \mathcal{E}_i} \mu_i(C). \tag{18}$$

Because $[\cdot]_+$ is monotone,

$$H_i(\hat{c}_i) = \left[\max_{C \in \mathcal{E}_i} \left(-\frac{a_C^\top h_i}{|C|}\right)\right]_+ = [-\mu_i^{\min}]_+. \tag{19}$$

Hence, if $\mu_i^{\min} < 0$, any

$$C_i^\star \in \arg\min_{C \in \mathcal{E}_i} \mu_i(C) \tag{20}$$

satisfies $\mathrm{viol}_i(C_i^\star) = H_i(\hat{c}_i)$ and is an exact most-violated cycle for the current full hinge. If $\mu_i^{\min} \geq 0$, no residual cycle has positive hinge violation. Thus a generic negative-cycle test can detect the existence of a violation, but exact separation requires the minimum-*mean* cycle because the surrogate is normalized by $|C|$.

The classical minimum-mean-cycle (MMC) value can be computed with a Karp-style dynamic program in $O(|V||E_R(x_i^\star)|)$ time (Karp, 1978). Since every original arc contributes at most one forward and one reverse residual copy, $|E_R(x_i^\star)| \leq 2|E|$. For a residual graph that is not strongly connected, the computation can be applied componentwise to its cycle-containing strongly connected components. This complexity is for one separator invocation, not for end-to-end training.

Computing the minimum mean value and recovering a cycle that attains it are distinct operations. Our implementation recovers a candidate minimizing cycle separately and checks that its mean agrees with the computed minimum before admitting it to the working set; numerical details are given in Appendix B.1. Given the reference flow, separation operates only on its weighted residual graph and does not solve the downstream predicted-cost minimum-cost-flow problem. Accordingly, "exact separation" means exact solution of the current full most-violated-cycle problem when the separator is invoked.

### 3.5 Solver-Free Training with a Bounded Cycle Cache

Calling the exact separator after every gradient update would repeatedly incur the graph-separation cost. We instead maintain, for each instance, a bounded working set

$$\mathcal{C}_i \subseteq \mathcal{E}_i, \qquad |\mathcal{C}_i| \leq b, \qquad b = 16, \tag{21}$$

and optimize the cached hinge. Here $\mathrm{viol}_i(C;\theta)$ denotes the violation in equation 13 evaluated at $\hat{c}_i = f_\theta(z_i)$, and $H_i(\theta) := H_i(f_\theta(z_i))$:

$$\widetilde{H}_i(\theta) = \max_{C \in \mathcal{C}_i} \mathrm{viol}_i(C;\theta), \qquad \widetilde{H}_i(\theta) = 0 \text{ if } \mathcal{C}_i = \varnothing. \tag{22}$$

Because $\mathcal{C}_i \subseteq \mathcal{E}_i$,

$$\widetilde{H}_i(\theta) \leq H_i(\theta). \tag{23}$$

Previously separated cycles remain valid after parameter updates because $\mathcal{R}(x_i^\star)$ is fixed by the reference flow and its capacity bounds; changing $\theta$ changes their scores, not their feasibility. The approximation is therefore in the maximum over constraints, not in the validity of the stored cycles.

The locked configuration uses a 20-epoch warm-up and subsequently considers cache refresh every five epochs. Eligible refreshes may be skipped when the cache already contains a positively violated cycle; such a cycle supplies a valid training signal but need not maximize the current full hinge. When separation is performed, a returned minimum-mean cycle is admitted only if

$$\mu_i^{\min} < -\tau, \qquad \tau = 10^{-7}. \tag{24}$$

If the current exact maximizer is inserted and retained, then $\widetilde{H}_i = H_i$ at that parameter value. The threshold may suppress insertion when the positive violation is at most $\tau$, and equality need not persist after subsequent model updates because an uncached cycle can become more violated. Thus equation 23 is the general relation between separator calls. When the active cache reaches capacity, cycles are retained according to their current and recent usefulness; canonicalization, duplicate handling, admission, and eviction are specified in Appendix B.2.

Under the specified insertion policy, a newly separated cycle receives an active-cache slot, and the active set changes only on a later separator insertion for that instance. Consequently, an admitted exact maximizer is retained until the next exact separator call, which enables the drift-dependent certificate in Corollary 1.

The full-information model additionally uses a normalized regression anchor. This anchor deliberately uses a different normalization from equation 11:

$$\tilde{v} = \frac{v}{\|v\|_2 + \varepsilon_a}, \qquad \varepsilon_a = 10^{-12}, \qquad \mathrm{nMSE}(\hat{c}_i, c_i) = \frac{1}{m} \left\| \tilde{\hat{c}}_i - \tilde{c}_i \right\|_2^2. \tag{25}$$

For a minibatch $\mathcal{B}$, the practical objective is

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \mathrm{nMSE}(\hat{c}_i, c_i) + \alpha \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \widetilde{H}_i(\theta), \qquad \alpha = 1. \tag{26}$$

The anchor provides a normalized coordinate-level prediction signal, while the cycle term targets decision-relevant residual directions. In the decision-only variant, true costs are unavailable during training; we therefore set $\kappa = 0$ and remove the regression anchor.

Throughout, "solver-free training" refers specifically to downstream minimum-cost-flow optimization. Given the reference optimal flows used for training, ordinary gradient updates score cached cycles and selected refreshes perform minimum-mean-cycle separation, but neither requires a call to the downstream minimum-cost-flow optimizer. GapCycle is therefore not optimization-free or oracle-free, and downstream solves used to compute validation or test regret are accounted for separately. The regret guarantee below applies directly to the full hinge $H_i$; Corollary 1 gives a conditional certificate for the cached hinge in terms of prediction drift since the most recent exact separator call.

# 4 THEORY

Section 3.4 establishes the exact cycle-mean identity used by the separator. Here we show that, for $\kappa > 0$, the full hinge pointwise controls downstream decision regret and derive a drift-dependent certificate for the cached hinge used in training. Complete proofs are given in Appendix A.

## 4.1 DECISION-REGRET CONTROL

Let

$$r_c = \sqrt{\|c\|_2^2 + \varepsilon_c^2}, \qquad U = \|u - l\|_1,$$

so that $c = r_c \bar{c}$.

**Theorem 1** (Pointwise decision-regret control)**.** *Let $\kappa > 0$. For any*

$$\hat{x} \in \arg\min_{x \in \mathcal{X}} \hat{c}^\top x,$$

*the true-cost regret of $\hat{x}$ satisfies*

$$c^\top \hat{x} - c^\top x^\star \le \frac{r_c}{\kappa} \|\hat{x} - x^\star\|_1 H(\hat{c}) \le \frac{r_c U}{\kappa} H(\hat{c}).$$

**Proof sketch.** Let $d = \hat{x} - x^\star$. A conformal residual decomposition gives

$$d = \sum_k \lambda_k a_{C_k}, \qquad \sum_k \lambda_k |C_k| = \|d\|_1.$$

For every such cycle, its reversal is residual at $\hat{x}$; optimality of $\hat{x}$ under $\hat{c}$ therefore gives $a_{C_k}^\top \bar{\hat{c}} \le 0$. Combining this sign with the full-hinge definition yields

$$a_{C_k}^\top \bar{c} \le \frac{|C_k|}{\kappa} H(\hat{c}).$$

Summing over the conformal decomposition gives the first inequality, and feasibility gives $\|\hat{x} - x^\star\|_1 \le \|u - l\|_1 = U$ for the second. The complete argument, including the residual decomposition and tie handling, is given in Appendix A.5.

**Corollary 1** (Cached-hinge regret certificate). *Assume $\kappa > 0$, and let $s$ be the most recent exact separator call. If an exact maximizing cycle with $H_s > \tau$ is retained until the next exact call, then, for*

$$\Delta_{s,t} = \|h_t - h_s\|_\infty,$$

*the current full hinge satisfies*

$$H_t \le \widetilde{H}_t + \tau + 2\Delta_{s,t}.$$

*Consequently,*

$$c^\top \hat{x}_t - c^\top x^\star \le \frac{r_c}{\kappa} \|\hat{x}_t - x^\star\|_1 \left(\widetilde{H}_t + \tau + 2\Delta_{s,t}\right).$$

**Proof sketch.** For every residual cycle, the normalized violation $\phi_C(h) = [-a_C^\top h/|C|]_+$ is 1-Lipschitz in $\|\cdot\|_\infty$ because $\|a_C\|_1 \le |C|$. If $H_s > \tau$, the retained maximizing cycle lower-bounds $\widetilde{H}_t$ within $\Delta_{s,t}$ while the full maximum changes by at most $\Delta_{s,t}$; if $H_s \le \tau$, Lipschitz continuity gives $H_t \le \tau + \Delta_{s,t}$. The two cases yield the displayed certificate, which combines with Theorem 1. The full proof is in Appendix A.6.

**Corollary 2** (Zero full hinge implies zero regret). *For $\kappa > 0$,*

$$H(\hat{c}) = 0 \quad \Longrightarrow \quad c^\top \hat{x} - c^\top x^\star = 0$$

*for every predicted-cost optimizer.*

The guarantee is pointwise. Theorem 1 applies directly to the full hinge, while Corollary 1 controls the cached hinge only under the stated retention condition and through the observable drift term $\Delta_{s,t}$. Neither result covers the $\kappa = 0$ decision-only variant or establishes calibration, Fisher consistency, population excess-risk control, or a statistical generalization guarantee.

# 5 Experiments

## 5.1 Experimental Setup

We separate method development from final evaluation. All modeling choices, hyperparameters, and cache rules are selected using four development topologies with two seeds per topology. We then freeze the configuration and evaluate on 24 previously unseen NETGEN topologies (Klingman et al., 1974), obtained from six network settings with four topologies per setting. Each confirmation topology has 100 nodes and contains 800 training, 200 validation, and 400 test instances, with context dimension $p = 16$. The confirmation topologies and datasets are generated only after configuration freezing and are hash-verified to be disjoint from the development data. The contextual-cost generator and complete NETGEN settings are given in Appendix C.

The comparison includes MOM (Sun et al., 2023), PFYL (Berthet et al., 2020), SPO+ (Elmachtoub & Grigas, 2022), single-basis LAVA (Berden et al., 2026), WISE (Wan & Liu, 2026), NCE and CMAP (Mulamba et al., 2021), LTR-pairwise (Mandi et al., 2022), nMSE, and MSE. Full-information GapCycle trains on $(z_i, c_i, x_i^\star)$; the decision-only setting is reported separately in the Appendix. Hyperparameters are selected using topology-macro validation regret on development data, with seeds averaged within topology. Confirmation labels are never used for model selection. Complete search spaces and the frozen configuration are reported in Appendix C.

Aggregate normalized regret $R_{\text{agg}}$ is primary. Seeds are averaged within topology before inference, giving $n = 24$ paired units; six-setting means provide a sensitivity analysis with Holm correction. Controlled compute runs use identical single-threaded hardware and record downstream MCF solves, separator calls, and preprocessing separately.

## 5.2 Results

**Held-out NETGEN performance.** On the 24-topology panel, GapCycle obtains the lowest aggregate normalized regret, $0.020800$, versus $0.023338$ for MOM, $0.028945$ for PFYL, and $0.031111$ for SPO+ (Table 1). The $0.002538$ difference from MOM is a $10.9\%$ reduction; larger nMSE and MSE values show that cost accuracy alone is insufficient on this benchmark.

Table 1: Decision quality on the held-out NETGEN confirmation panel. Lower aggregate normalized regret is better.

| Method | Aggregate normalized regret ↓ |
|---|---|
| GapCycle | **0.020800** |
| MOM | 0.023338 |
| PFYL | 0.028945 |
| SPO+ | 0.031111 |
| LTR-pairwise | 0.081165 |
| CMAP | 0.082610 |
| LAVA (single basis) | 0.093885 |
| nMSE | 0.113916 |
| MSE | 0.171385 |
| WISE | 0.460275 |
| NCE | 0.491166 |

Topology-level comparisons favor GapCycle after Holm correction at $0.05$; six-setting and comparator-specific results are in Appendix D. This conclusion is restricted to the registered NETGEN protocol.

**Computational cost.** GapCycle and MOM use zero downstream training solves and require $1.4500$ and $1.5244$ seconds per epoch, respectively. PFYL and SPO+ use $1{,}083{,}444$ and $92{,}200$ solves and require $7.1474$ and $2.6382$ seconds per epoch, or approximately $4.9\times$ and $1.8\times$ GapCycle's time. These are per-epoch rather than end-to-end comparisons; separator work, reference-label generation, and preprocessing are reported in Appendix D.

**Shortest-path robustness.** Across a 600-fit PyEPO shortest-path sweep over degree, noise, predictor class, and five seeds (Tang & Khalil, 2022), mean regret is $0.0544$ for GapCycle, $0.0573$ for PFYL, and $0.0595$ for SPO+. The ordering is not uniform across configurations, so this supports robustness within network-flow structure rather than uniform dominance; see Appendix D.

## 6 Limitations

GapCycle is specialized to capacitated MCF and assumes reference optima; its full-information model also uses realized costs. The pointwise cached-hinge certificate requires retention and drift control, while all regret results require $\kappa > 0$ and exclude the decision-only variant. Evaluation is limited to synthetic costs and network-flow benchmarks, leaving larger real deployments open.

## 7 Conclusion

GapCycle uses classical residual-cycle geometry in a contextual, true-cost-aware MCF surrogate. A bounded working set amortizes separation, while full- and cached-hinge certificates provide pointwise regret control for $\kappa > 0$. On held-out NETGEN topologies, GapCycle achieves the lowest aggregate normalized regret among the evaluated methods without downstream MCF training solves once reference optima are available.

## AI Use Statement

Generative AI tools assisted with drafting and editing, the authors take full responsibility for the paper.

## Reproducibility Statement

The appendix provides complete proofs and experimental details, while the supplementary artifact contains the code, data generators, configurations, seeds, environment and solver settings, run records, integrity hashes, and scripts used to regenerate the reported results.

## References


Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems 32*, 2019.

Ravindra K Ahuja and James B Orlin. Inverse optimization. *Operations research*, 49(5):771–783, 2001.

Ravindra K Ahuja and James B Orlin. Combinatorial algorithms for inverse network flow problems. *Networks: An International Journal*, 40(4):181–187, 2002.

Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1993. ISBN 0-13-617549-X.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *PMLR*, pp. 136–145, 2017.

Senne Berden, Ali Mahmutoğulları, Dimos Tsouros, and Tias Guns. Solver-free decision-focused learning for linear optimization problems. *Advances in Neural Information Processing Systems*, 38:158127–158145, 2026.

Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. Learning with differentiable perturbed optimizers. In *Advances in Neural Information Processing Systems 33*, 2020.

Mathieu Blondel, André F. T. Martins, and Vlad Niculae. Learning with Fenchel-Young losses. *Journal of Machine Learning Research*, 21(35):1–69, 2020.

Mmanu Chaturvedi and Ross M McConnell. A note on finding minimum mean cycle. *Information Processing Letters*, 127:21–22, 2017.

Adam N. Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68 (1):9–26, 2022. doi: 10.1287/mnsc.2020.3922.

Richard M Karp. A characterization of the minimum cycle mean in a digraph. *Discrete mathematics*, 23(3):309–311, 1978.

Darwin Klingman, Albert Napier, and Joel Stutz. Netgen: A program for generating large scale capacitated assignment, transportation, and minimum cost flow network problems. *management science*, 20(5):814–821, 1974.

Jayanta Mandi, Vıctor Bucarey, Maxime Mulamba Ke Tchomba, and Tias Guns. Decision-focused learning: Through the lens of learning to rank. In *International conference on machine learning*, pp. 14935–14947. PMLR, 2022.

Maxime Mulamba, Jayanta Mandi, Michelangelo Diligenti, Michele Lombardi, Victor Bucarey, and Tias Guns. Contrastive losses and solution caching for predict-and-optimize. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, pp. 2833–2840, 2021. doi: 10.24963/ijcai.2021/390.

Chunlin Sun, Shang Liu, and Xiaocheng Li. Maximum optimality margin: A unified approach for contextual linear programming and inverse linear programming. In *International Conference on Machine Learning*, pp. 32886–32912. PMLR, 2023.

Bo Tang and Elias Boutros Khalil. Pyepo: A pytorch-based end-to-end predict-then-optimize library with linear objective function. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*, 2022.

Beichen Wan and Mo Liu. A solver-free training method for predict-then-optimize. *arXiv preprint arXiv:2606.19587*, 2026.

Jianzhong Zhang and Zhenhong Liu. A general model of some inverse combinatorial optimization problems and its solution method under $\ell_\infty$ norm. *Journal of Combinatorial Optimization*, 6(2): 207–227, 2002.

# A PROOFS

This section gives complete proofs of the exact-separation and pointwise regret-control claims stated in the main text. Residual arcs retain their original-arc identities throughout, so the residual graph is a labeled multigraph. We do not assume integral flows, integral costs, unique optima, or zero lower bounds.

## A.1 RESIDUAL-CYCLE PRELIMINARIES

Recall

$$\mathcal{X} = \{x \in \mathbb{R}^m : Ax = \beta,\ l \le x \le u\}.$$

For an original arc $e$, define the forward and reverse residual capacities at $x^\star$ by

$$\rho_e^+ = u_e - x_e^\star, \qquad \rho_e^- = x_e^\star - l_e.$$

A residual arc $r$ retains the original-arc identity $e(r)$ and has sign

$$\sigma_r = \begin{cases} +1, & r \text{ is forward,} \\ -1, & r \text{ is reverse.} \end{cases}$$

We write $E_R(x^\star)$ for the residual-arc set and abbreviate it as $E_R$ within Appendix A. For a directed residual cycle $C$, its signed direction in original-arc coordinates is

$$(a_C)_e = \sum_{\substack{r \in C \\ e(r)=e}} \sigma_r.$$

Because labeled residual contributions associated with the same original arc can cancel, we do not use the global identities $a_C \in \{-1, 0, 1\}^m$ or $\|a_C\|_1 = |C|$.

**Lemma 1** (Feasible residual-cycle direction)**.** *For every directed residual cycle $C$,*

$$Aa_C = 0.$$

*Let*

$$\delta_C = \min_{r \in C} \rho_r,$$

*where $\rho_r$ is the residual capacity of $r$. Then*

$$x^\star + \delta a_C \in \mathcal{X} \qquad \textit{for every } 0 \le \delta \le \delta_C.$$

*For any cost vector $q$,*

$$q^\top (x^\star + \delta a_C) - q^\top x^\star = \delta a_C^\top q.$$

*Proof.* A directed residual cycle has zero net incidence at every node. Mapping a forward residual copy to $+1$ on its original arc and a reverse copy to $-1$ therefore gives $Aa_C = 0$, so flow balance is preserved. If $C$ uses a forward copy of $e$, an augmentation of size at most $\rho_e^+$ cannot exceed $u_e$; if it uses a reverse copy, an augmentation of size at most $\rho_e^-$ cannot fall below $l_e$. Hence every $0 \le \delta \le \delta_C$ preserves the box constraints. The objective identity follows by linearity. □

**Proposition 1** (Residual-cycle optimality)**.** *For any $q \in \mathbb{R}^m$,*

$$x^\star \in \arg\min_{x \in \mathcal{X}} q^\top x \quad \Longleftrightarrow \quad a_C^\top q \ge 0 \quad \forall C \in \mathcal{E}(x^\star).$$

*Proof.* If $a_C^\top q < 0$ for some residual cycle, Lemma 1 provides an improving feasible augmentation, so $x^\star$ is not optimal. Conversely, let $x \in \mathcal{X}$. The feasible difference $x - x^\star$ lifts to a nonnegative circulation in the residual graph and therefore decomposes into directed residual cycles. The construction is given explicitly in Lemma 2. If every residual cycle has nonnegative $q$-cost, then $q^\top(x - x^\star) \geq 0$ for every feasible $x$, proving optimality. This is the classical residual-cycle optimality condition for minimum-cost flow (Ahuja et al., 1993). □

### A.2 Proof of the Exact-Separation Identity

For

$$\bar{v} = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon_c^2}}, \qquad \varepsilon_c = 10^{-8},$$

define

$$h = \bar{\hat{c}} - \kappa \bar{c}.$$

The full cycle violation is

$$\mathrm{viol}(C) = \left[ -\frac{a_C^\top h}{|C|} \right]_+,$$

where $|C|$ is the number of labeled residual arcs, and

$$H(\hat{c}) = \max_{C \in \mathcal{E}(x^\star)} \mathrm{viol}(C),$$

with $H(\hat{c}) = 0$ if the residual graph contains no directed cycle. Assign residual weight

$$\omega(r) = \sigma_r h_{e(r)}.$$

Then

$$\sum_{r \in C} \omega(r) = a_C^\top h, \qquad \mu(C) = \frac{a_C^\top h}{|C|}.$$

Suppose $\mathcal{E}(x^\star) \neq \emptyset$, and define

$$\mu^{\min} = \min_{C \in \mathcal{E}(x^\star)} \frac{a_C^\top h}{|C|}.$$

Then

$$H(\hat{c}) = [-\mu^{\min}]_+.$$

If $\mu^{\min} < 0$, every minimum-mean residual cycle maximizes the full cycle violation. If $\mu^{\min} \geq 0$, all cycle violations are zero.

*Proof.*

$$\begin{aligned} H(\hat{c}) &= \max_C \left[ -\frac{a_C^\top h}{|C|} \right]_+ \\ &= \left[ \max_C \left( -\frac{a_C^\top h}{|C|} \right) \right]_+ \\ &= \left[ -\min_C \frac{a_C^\top h}{|C|} \right]_+ \\ &= [-\mu^{\min}]_+. \end{aligned}$$

The two cases follow from the sign of $\mu^{\min}$. □

The separator therefore solves a minimum-*mean* cycle problem, not a minimum-total-cycle problem. A generic negative-cycle test determines whether a negative cycle exists but does not, in general, maximize the length-normalized violation. Karp's dynamic program computes the minimum cycle-mean value in $O(|V||E_R|)$ time, where $|E_R| \leq 2|E|$ (Karp, 1978). For non-strongly-connected residual graphs, cycle-containing strongly connected components can be processed separately. Implementation and cycle-recovery safeguards are given in Appendix B.1.

### A.3 Conformal Residual Decomposition

**Lemma 2** (Conformal residual decomposition)**.** *Let $x^\star, \hat{x} \in \mathcal{X}$ and set*

$$d = \hat{x} - x^\star.$$

*Then there exist residual cycles $C_1, \dots, C_K \in \mathcal{E}(x^\star)$ and coefficients $\lambda_k > 0$ such that*

$$d = \sum_{k=1}^{K} \lambda_k a_{C_k}$$

*and*

$$\sum_{k=1}^{K} \lambda_k |C_k| = \|d\|_1.$$

*The decomposition is sign-conformal: a cycle uses a forward residual copy of $e$ only if $d_e > 0$, and a reverse residual copy only if $d_e < 0$.*

*Proof.* Construct a nonnegative flow $y$ on the residual graph by

$$y_{e^+} = d_e \quad \text{if } d_e > 0, \qquad y_{e^-} = -d_e \quad \text{if } d_e < 0,$$

and zero otherwise. If $d_e > 0$,

$$y_{e^+} = \hat{x}_e - x^\star_e \le u_e - x^\star_e = \rho^+_e;$$

if $d_e < 0$,

$$y_{e^-} = x^\star_e - \hat{x}_e \le x^\star_e - l_e = \rho^-_e.$$

Thus $y$ respects residual capacities. Since both flows satisfy the same flow balance,

$$Ad = A\hat{x} - Ax^\star = 0,$$

so $y$ is a nonnegative residual circulation. Decompose $y$ into directed cycles,

$$y = \sum_{k=1}^{K} \lambda_k \chi_{C_k}, \qquad \lambda_k > 0.$$

Here $\chi_{C_k} \in \{0,1\}^{|E_R|}$ is the residual-arc incidence vector of cycle $C_k$. Mapping residual arcs back to signed original-arc coordinates gives

$$d = \sum_{k=1}^{K} \lambda_k a_{C_k}.$$

By construction, residual flow is placed only in the sign of $d_e$, so the decomposition is conformal. Finally,

$$\begin{aligned}
\sum_k \lambda_k |C_k| &= \sum_{r \in E_R} y_r \\
&= \sum_{e: d_e > 0} d_e + \sum_{e: d_e < 0} (-d_e) \\
&= \sum_e |d_e| \\
&= \|d\|_1.
\end{aligned}$$

This mass identity is obtained from the lifted residual circulation; it does not rely on the generally false identity $\|a_C\|_1 = |C|$ for an arbitrary labeled residual cycle. □

### A.4 PREDICTED-COST SIGN

**Lemma 3** (Predicted-cost sign on conformal cycles)**.** *Let*

$$\hat{x} \in \arg\min_{x \in \mathcal{X}} \hat{c}^\top x,$$

*and let $C$ occur in the conformal decomposition of $\hat{x} - x^\star$. Then*

$$a_C^\top \hat{c} \le 0 \qquad \textit{and} \qquad a_C^\top \bar{\hat{c}} \le 0.$$

*Proof.* If $C$ uses the forward residual copy $e^+$ at $x^\star$, conformality gives $d_e > 0$, hence $x_e^\star < \hat{x}_e$ and the reverse copy $e^-$ has positive residual capacity at $\hat{x}$. If $C$ uses $e^-$, then $d_e < 0$, hence $\hat{x}_e < x_e^\star$ and $e^+$ is residual at $\hat{x}$. Reversing every labeled arc of $C$ therefore gives a valid residual cycle at $\hat{x}$ with signed direction $-a_C$. Optimality of $\hat{x}$ under $\hat{c}$ implies

$$(-a_C)^\top \hat{c} \ge 0,$$

so $a_C^\top \hat{c} \le 0$. The normalization multiplies $\hat{c}$ by a positive scalar, preserving the sign. □

### A.5 PROOF OF THEOREM 1

Define

$$r_c = \sqrt{\|c\|_2^2 + \varepsilon_c^2}, \qquad c = r_c \bar{c}, \qquad U = \|u - l\|_1.$$

For $\kappa > 0$ and any

$$\hat{x} \in \arg\min_{x \in \mathcal{X}} \hat{c}^\top x,$$

$$c^\top \hat{x} - c^\top x^\star \le \frac{r_c}{\kappa} \|\hat{x} - x^\star\|_1 H(\hat{c}) \le \frac{r_c U}{\kappa} H(\hat{c}).$$

*Proof.* Let $d = \hat{x} - x^\star$ and apply Lemma 2:

$$d = \sum_k \lambda_k a_{C_k}, \qquad \sum_k \lambda_k |C_k| = \|d\|_1.$$

For every decomposition cycle,

$$H(\hat{c}) \ge \left[ \frac{\kappa a_{C_k}^\top \bar{c} - a_{C_k}^\top \bar{\hat{c}}}{|C_k|} \right]_+ .$$

Since $[t]_+ \ge t$,

$$|C_k| H(\hat{c}) \ge \kappa a_{C_k}^\top \bar{c} - a_{C_k}^\top \bar{\hat{c}}.$$

Lemma 3 gives $a_{C_k}^\top \bar{\hat{c}} \le 0$, so

$$a_{C_k}^\top \bar{c} \le \frac{|C_k|}{\kappa} H(\hat{c}).$$

Therefore

$$\begin{aligned} c^\top \hat{x} - c^\top x^\star &= c^\top d \\ &= r_c \sum_k \lambda_k a_{C_k}^\top \bar{c} \\ &\le \frac{r_c}{\kappa} H(\hat{c}) \sum_k \lambda_k |C_k| \\ &= \frac{r_c}{\kappa} \|d\|_1 H(\hat{c}). \end{aligned}$$

Finally, feasibility gives $|d_e| \le u_e - l_e$ for every arc, and hence

$$\|d\|_1 \le \|u - l\|_1 = U.$$

□

### A.6 Proof of Corollary 1

For a residual cycle $C$, define

$$\phi_C(h) = \left[ -\frac{a_C^\top h}{|C|} \right]_+ .$$

Each labeled residual arc contributes either $+1$ or $-1$ to one original-arc coordinate of $a_C$. The triangle inequality therefore gives

$$\|a_C\|_1 \le |C|,$$

where strict inequality is possible when labeled residual contributions on the same original arc cancel. Since the positive-part map is 1-Lipschitz, for any $h, h'$,

$$\begin{aligned} |\phi_C(h) - \phi_C(h')| &\le \frac{|a_C^\top (h - h')|}{|C|} \\ &\le \frac{\|a_C\|_1}{|C|} \|h - h'\|_\infty \\ &\le \|h - h'\|_\infty. \end{aligned}$$

Thus every $\phi_C$ is 1-Lipschitz in $\|\cdot\|_\infty$. The full hinge, being the maximum of these functions, satisfies

$$|H(h) - H(h')| \le \|h - h'\|_\infty.$$

Let $s$ be the most recent exact separator call and set

$$\Delta_{s,t} = \|h_t - h_s\|_\infty.$$

If $H_s > \tau$, let $C_s$ be an exact maximizing cycle found at $s$. By the retention assumption, $C_s$ remains in the active cache at time $t$, so

$$\widetilde{H}_t \ge \phi_{C_s}(h_t) \ge \phi_{C_s}(h_s) - \Delta_{s,t} = H_s - \Delta_{s,t}.$$

The Lipschitz property of the full hinge also gives

$$H_t \le H_s + \Delta_{s,t},$$

and hence

$$0 \le H_t - \widetilde{H}_t \le 2\Delta_{s,t}.$$

If instead $H_s \le \tau$, then

$$H_t \le H_s + \Delta_{s,t} \le \tau + \Delta_{s,t},$$

while $\widetilde{H}_t \ge 0$, so

$$0 \le H_t - \widetilde{H}_t \le \tau + \Delta_{s,t}.$$

Combining the two cases yields

$$H_t - \widetilde{H}_t \le \tau + 2\Delta_{s,t}$$

and therefore

$$H_t \le \widetilde{H}_t + \tau + 2\Delta_{s,t}.$$

Substituting this inequality into Theorem 1 proves Corollary 1.

### A.7 Zero-Hinge Corollary and Scope

Corollary 2 states that, for $\kappa > 0$,

$$H(\hat{c}) = 0 \quad \Longrightarrow \quad c^\top \hat{x} - c^\top x^\star = 0$$

for every predicted-cost optimizer.

*Proof.* Theorem 1 gives $c^\top \hat{x} - c^\top x^\star \le 0$, while true-cost optimality of $x^\star$ gives the reverse inequality. □

At $\kappa = 0$, zero full hinge guarantees that $x^\star$ is a predicted-cost optimum, but another predicted-cost optimum selected under a tie can have positive true regret. The theorem therefore does not cover GapCycle-DecisionOnly. More generally, the result is deterministic and pointwise. The full-hinge theorem combines with Corollary 1 to give a cached-hinge certificate under the stated retention condition and drift term; without those conditions, $\widetilde{H} \le H$ alone is insufficient. The results do not establish calibration, Fisher consistency, population excess-risk control, finite-sample generalization, or optimization convergence.

# B Additional Method Details

## B.1 Residual representation and exact separator

Residual arcs are stored using pairs consisting of an original-arc identifier and a residual sign. Parallel and antiparallel original arcs therefore remain distinct. Canonical cache keys are invariant to cyclic rotation of one directed traversal, while the reverse traversal remains a different key. Cycles whose signed original-arc incidence is identically zero are rejected.

For instance $i$, the separator uses

$$h_i = \bar{\hat{c}}_i - \kappa \bar{c}_i, \qquad \omega_i(r) = \sigma_r h_{i,e(r)}.$$

Karp-style dynamic programming computes the minimum cycle-mean value, after which cycle recovery is performed separately. The implementation re-evaluates the recovered cycle under the original residual weights and checks agreement with the dynamic-program value. The minimum-mean value algorithm itself is classical (Karp, 1978); the separation between value computation and cycle recovery avoids relying on Karp's incorrect suggested recovery procedure (Chaturvedi & McConnell, 2017).

## B.2 Bounded working set and refresh policy

For each training instance,

$$\mathcal{C}_i \subseteq \mathcal{E}_i, \qquad |\mathcal{C}_i| \le b, \qquad b = 16.$$

The practical hinge is

$$\widetilde{H}_i = \max_{C \in \mathcal{C}_i} \left[ -\frac{a_C^\top h_i}{|C|} \right]_+, \qquad \widetilde{H}_i = 0 \text{ if } \mathcal{C}_i = \varnothing,$$

so $\widetilde{H}_i \le H_i$.

The frozen schedule uses a 20-epoch warm-up. After warm-up, exact separation is considered every five epochs. An eligible call can be skipped when the current cache already contains a positively violated cycle. Among instances without a cached violation, the implementation prioritizes those that are overdue and then those with larger

$$\|\bar{\hat{c}}_i - \bar{\hat{c}}_i^{\text{last}}\|_\infty.$$

Here $\bar{\hat{c}}_i^{\text{last}}$ is the normalized prediction stored at the most recent separator call for instance $i$. A newly separated cycle is admitted only if

$$\mu_i^{\min} < -\tau, \qquad \tau = 10^{-7}.$$

Hence a positive full-hinge value with $0 < H_i \le \tau$ can be suppressed by the admission tolerance. Even when the current full-hinge maximizer is inserted, $\widetilde{H}_i = H_i$ is guaranteed only at that model state if the maximizer is retained; later parameter updates can change the ordering of cycle violations.

Each cached cycle $C$ carries an exponential moving-average (EMA) score $s_C^{\text{EMA}}$, updated by

$$s_C^{\text{EMA}} \leftarrow 0.9\, s_C^{\text{EMA}} + 0.1\ \text{viol}_i(C),$$

and initialized from its insertion violation. On insertion into a full cache, let

$$K_{\text{keep}} = \min(4, b-1, |\mathcal{C}_i|).$$

The $K_{\text{keep}}$ entries with largest current violation are retained. Remaining slots are ranked by $(s_C^{\text{EMA}}$, last violated epoch$)$, the new cycle receives the final slot, and ties are broken deterministically by canonical key. The active set changes only upon a later separator insertion for the same instance, so the newly inserted exact maximizer remains active until that instance's next exact separator call. The active cache is bounded; the implementation separately retains previously seen cycle keys for diagnostics, and that diagnostic history is not the object used for the active-memory bound.

### B.3 NUMERICAL SAFEGUARDS

Three numerical tolerances play different roles. The admission threshold is $10^{-7}$; agreement between the Karp value and the recovered cycle mean is checked to $10^{-10}$. Cycle recovery uses a distinct shift. With $\mathrm{ulp} = \texttt{np.finfo(float).eps}$,

$$\mathrm{scale} = \max\left(1, \max_{r \in E_S} |w_r|, |\mu^\star|\right),$$

where $E_S$ is the residual-arc set of the selected strongly connected component, $w_r$ is the residual weight of arc $r$, $\mu^\star$ is the computed minimum cycle-mean value in that component, and $n$ is its number of vertices. The recovery shift is

$$\max\big(64\,\mathrm{ulp} \cdot \mathrm{scale} \cdot n, \min(10^{-10}/8, 10^{-12}\mathrm{scale})\big).$$

These tolerances are not interchangeable.

The cost normalization used by the cycle loss is

$$\bar{v} = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon_c^2}}, \qquad \varepsilon_c = 10^{-8},$$

whereas the regression anchor uses

$$\tilde{v} = \frac{v}{\|v\|_2 + \varepsilon_a}, \qquad \varepsilon_a = 10^{-12}.$$

The two normalizers serve different purposes and are kept distinct.

### B.4 TRAINING OBJECTIVE AND INFORMATION SETTINGS

The normalized regression anchor is

$$\mathrm{nMSE}(\hat{c}_i, c_i) = \frac{1}{m}\|\tilde{\hat{c}}_i - \tilde{c}_i\|_2^2.$$

For minibatch $\mathcal{B}$, the full-information objective is

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|}\sum_{i \in \mathcal{B}} \mathrm{nMSE}(\hat{c}_i, c_i) + \alpha \frac{1}{|\mathcal{B}|}\sum_{i \in \mathcal{B}} \widetilde{H}_i(\theta), \qquad \alpha = 1.$$

Full GapCycle uses $(z_i, c_i, x_i^\star)$. The decision-only variant uses $(z_i, x_i^\star)$, sets $\kappa = 0$, and removes nMSE. A run with $\kappa = 0$ but nMSE still present is a margin ablation rather than GAPCYCLE-DECISIONONLY.

Gradients flow through the predicted costs and their normalization. The MMC argmin is discrete and is not differentiated through: separation selects a cycle, after which the corresponding current violation is evaluated as part of the differentiable loss.

### B.5 TRAINING PROCEDURE AND COMPUTATIONAL SCOPE

For each minibatch, GapCycle predicts costs, evaluates the cached cycles, optionally invokes exact separation according to the refresh and gating rules, updates the bounded cache, computes the cached hinge, adds the regression anchor in the full-information setting, and backpropagates through the resulting loss. A separator call has $O(|V||E_R(x_i^\star)|)$ graph complexity for minimum-mean value computation; updates that reuse the cache avoid another downstream minimum-cost-flow solve.

Our use of "solver-free training" is deliberately scoped: given the reference optimal flows used for training, gradient-based updates make no calls to the downstream minimum-cost-flow solver. GapCycle still performs MMC separation, cached-cycle scoring, and neural optimization. Reference-label generation and validation/test regret evaluation remain separate optimization tasks.

Table 2: Registered NETGEN settings. Each setting contributes four held-out confirmation topologies.

| Setting | Nodes | Arcs | Sources/sinks | Total supply | Capacity range |
|---|---|---|---|---|---|
| baseline | 100 | 400 | 10/10 | 1000 | 25–500 |
| high flow tight caps | 100 | 400 | 20/20 | 2000 | 25–500 |
| low flow dense | 100 | 800 | 5/5 | 100 | 25–500 |
| many terminals | 100 | 400 | 40/40 | 2000 | 25–500 |
| many terminals dense | 100 | 800 | 45/45 | 2000 | 25–500 |
| mid flow half capacitated | 100 | 800 | 20/20 | 500 | 25–500 |

# C Experimental Details

## C.1 NETGEN confirmation panel and contextual costs

The confirmation panel contains six registered settings from one NETGEN family (Klingman et al., 1974), with four newly generated topologies per setting. The settings are summarized below.

Each topology contains 800 training, 200 validation, and 400 test instances, with context dimension $p = 16$. Configuration freezing precedes confirmation generation and evaluation, and topology and dataset hashes are disjoint from all development data.

NETGEN supplies topology and flow parameters; the contextual feature-to-cost mapping is generated separately. For each topology,

$$B_{ej} \sim \text{Bernoulli}(0.5), \qquad B_e = (B_{e1}, \dots, B_{ep})^\top,$$

for $e \in E$ and $j \in \{1, \dots, p\}$, so $B \in \{0,1\}^{m \times p}$ and $B_e$ is its row for arc $e$. The matrix $B$ is sampled once from the topology seed. For each instance,

$$z_i \sim \mathcal{N}(0, I_p),$$

and

$$c_{i,e} = \frac{\left(B_e^\top z_i/\sqrt{p} + 3\right)^d + 1}{3.5^d} \varepsilon_{i,e}, \qquad p = 16, \quad d = 4,$$

where

$$\varepsilon_{i,e} \sim \mathcal{U}(0.9, 1.1).$$

The instance seed draws $z_i$ and $\varepsilon_{i,e}$. No additional clipping, rounding, normalization, or rescaling is applied.

## C.2 Shortest-path protocol

The PyEPO shortest-path sweep (Tang & Khalil, 2022) varies polynomial degree over $\{1, 2, 4, 6, 8\}$, noise over $\{0, 0.5\}$, predictor class over linear and MLP models, and five random seeds. Six methods are evaluated: GapCycle, PFYL, SPO+, nMSE, MSE, and LAVA. The complete sweep contains

$$5 \times 2 \times 2 \times 5 \times 6 = 600$$

fits. A separate registered train–test distribution-shift experiment is kept separate from this degree/noise sweep.

## C.3 Baselines, campaigns, and supervision

The locked NETGEN comparison includes GapCycle, MOM, PFYL, SPO+-N, SPO+, LTR-pairwise, CMAP, LAVA (single basis), nMSE, MSE, WISE, and NCE. SPO+-N is an additional diagnostic arm. Full LAVA with Transition Node Pivoting and backend-specific CaVE evaluations are separate diagnostics rather than rows from the same locked run matrix.

The original core confirmation campaign contains

$$24 \text{ topologies} \times 3 \text{ seeds} \times 5 \text{ core methods} = 360$$

fits. Additional comparator campaigns are tracked in separate registered ledgers. The full 12-method table should therefore not be described as a single 360-fit campaign.

Table 3: Training information and downstream-solver use in the locked information-setting comparison.

| Method | Training information | Downstream training solves |
|---|---|---|
| GapCycle (full) | $z_i, c_i, x_i^\star$ | 0 |
| GAPCYCLE-DECISIONONLY | $z_i, x_i^\star$ | 0 |
| LAVA | $z_i, x_i^\star$ | 0 |
| PFYL | verified implementation information path | $> 0$ |

Table 4: Development tuning ledger. Exploratory probes are listed separately from selectable configurations.

| Method/stage | Search grid | Cells | Development fits | Selected configuration |
|---|---|---|---|---|
| GapCycle LR | $\{0.001, 0.003, 0.01\} \times \{\text{constant}, \text{cosine}\}$ | 6 | 48 | LR 0.001, cosine |
| GapCycle LR extension | $0.0001 \times \{\text{constant}, \text{cosine}\}$ | 2 | 16 | not selected |
| GapCycle cache scoring | {predicted, loss-augmented} | 2 | 16 | loss-augmented |
| GapCycle probes | $\alpha$/loss-top-$k$ variants | 3 | 24 | $\alpha = 1$, loss-top-$k$ = 1 |
| PFYL LR | $\{0.0001, 0.001, 0.003, 0.01\} \times \{\text{constant}, \text{cosine}\}$ | 8 | 64 | LR 0.001, cosine |
| PFYL $\sigma \times M$ | $\{0.25, 0.5, 1, 2\} \times \{10, 25, 50\}$ | 12 | 96 | $\sigma = 1.0,\ M = 10$ |
| MOM | $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\} \times \{0.1, 1, 10\}$ | 15 | 120 | LR 0.001, margin 10 |
| SPO+-N | same five LRs $\times\{0.05, 0.1, 1\}$ | 15 | 120 | LR 0.001, ratio 0.05 |
| LAVA single-basis | six LRs $\{10^{-7}, \ldots, 10^{-2}\} \times \{-0.1, 0\}$ | 12 | 96 | LR 0.001, threshold $-0.1$ |
| WISE | $\{0.001, 0.01, 0.1\} \times \{0, 10^{-5}, 10^{-4}\}$ | 9 | 72 | LR 0.01, decay $10^{-4}$ |
| NCE | six LRs $\{10^{-6}, \ldots, 10^{-2}\}$, cosine | 6 | 48 | LR 0.001, solve ratio 0.05 |
| CMAP | six LRs $\{10^{-6}, \ldots, 10^{-2}\}$, cosine | 6 | 48 | LR 0.001, solve ratio 0.05 |
| LTR-pairwise | six LRs $\{10^{-6}, \ldots, 10^{-2}\}$, cosine | 6 | 48 | LR 0.001, cosine |
| SPO+ | registered optimizer only | 1 | 8 | LR 0.001, cosine |
| nMSE | registered optimizer only | 1 | 8 | LR 0.001, cosine |
| MSE | registered optimizer only | 1 | 8 | LR 0.001, cosine |

LR denotes learning rate; for PFYL, $\sigma$ is the perturbation scale and $M$ is the number of perturbation samples. The loss-top-$k$ parameter is the number of cached violations aggregated in the cycle loss. Hyperparameters were selected using topology-macro validation regret at the 540-second checkpoint. Seeds were averaged within topology. Confirmation test labels were not consulted. Exploratory probes are reported separately and are not counted as independent selectable configurations. SPO+, nMSE, and MSE inherited the registered optimizer rather than receiving dedicated final grids.

### C.4 DEVELOPMENT SELECTION AND FROZEN CONFIGURATION

Development uses four topologies and two seeds per topology. Selection is based on topology-macro validation regret at the common 540-second checkpoint after averaging seeds within topology; confirmation test labels are not consulted. The complete method-specific tuning ledger is reported in Table 4. The search spaces are intentionally reported as method-specific rather than described as identical.

The frozen PFYL perturbation configuration uses $\sigma = 1.0$ and $M = 10$. GapCycle and PFYL both use Adam with learning rate $0.001$, a cosine schedule ending at $0.0001$ over $12{,}500$ updates, no restarts, zero weight decay, global gradient-norm clipping at $1.0$, Adam coefficients $(0.9, 0.999)$, numerical constant $10^{-8}$, and batch size 32. SPO+, nMSE, and MSE inherit their registered optimizer configuration and do not receive dedicated final grids.

### C.5 CHECKPOINTING AND STATISTICAL ANALYSIS

A run that reaches 540 seconds is evaluated at the budget checkpoint. If early stopping terminates a run before 540 seconds, evaluation uses the restored best-validation checkpoint. A separate "@stop" diagnostic, when reported, is not substituted for the primary budgeted comparison.

Training seeds are averaged within topology before inference. The primary paired unit is therefore the topology, $n = 24$. As a conservative sensitivity analysis, the four topologies within each NETGEN setting are first averaged, producing $n = 6$ setting means. For comparator $M$,

$$\Delta_M = R_{\text{agg},M} - R_{\text{agg,GapCycle}},$$

so $\Delta_M > 0$ favors GapCycle. Holm correction is applied across the 11 locked comparators at both statistical units; all corrected comparisons satisfy $p < 0.05$.

Table 5: The configuration frozen before the confirmation panel ran. Predictor, cache capacity and retention were held fixed throughout the campaign; only the learning rate, schedule and cache score space were selected.

| Setting | Value |
|---|---|
| surrogate | `gapcycle` |
| $\kappa$ | `0.5` |
| $\alpha$ | `1.0` |
| loss_top_k | `1` |
| cache capacity $b$ | `16` |
| retention_top_k | `4` |
| cache score space | `loss_augmented` |
| refresh trigger space | `predicted` |
| optimizer | Adam |
| learning rate | 0.001 |
| schedule | cosine to 0.0001 |
| schedule horizon | 12,500 updates |
| restarts | none |
| weight decay | 0 |
| gradient clipping | global norm 1.0 |
| Adam betas | $(0.9, 0.999)$ |
| Adam epsilon | $10^{-8}$ |
| batch size | 32 |
| predictor depth | `2` |
| predictor width | `128` |
| separator | Karp-style MMC value with verified cycle recovery |
| train size | `800` |
| val size | `200` |
| test size | `400` |
| max epochs | `500` |
| patience | `50` |
| optimization budget (s) | `540` |

`loss_top_k` is the number of cached violations aggregated in the loss; `retention_top_k` is the number of highest-scoring cached cycles protected during eviction.

The primary PFYL comparison is superiority. The secondary registered non-inferiority analysis defines

$$d_{\mathrm{PFYL}} = R_{\mathrm{GapCycle}} - R_{\mathrm{PFYL}}, \qquad \delta_{\mathrm{NI}} = 0.25\,(R_{\text{worst arm}} - R_{\text{best arm}}),$$

and concludes non-inferiority when the upper one-sided confidence bound for $d_{\mathrm{PFYL}}$ lies below $\delta_{\mathrm{NI}}$.

### C.6 Computational environment and reproduction controls

Controlled training uses batch size 32 and single-threaded CPU execution, with one OMP, MKL, OpenBLAS, Torch, and Gurobi thread. The verified environment is Linux `5.14.0-570.17.1.el9_6`, Python `3.9.25`, NumPy `2.0.2`, PyTorch `2.5.1+cu121`, Gurobi `12.0.3`, and OR-Tools `9.15.6755`. The CUDA-enabled PyTorch build does not imply GPU execution for the controlled training jobs.

Downstream MCF solves, MMC separator calls, preprocessing calls, oracle time, seconds per epoch, training time, throughput, and memory are recorded as separate quantities. MMC separation is not counted as a downstream MCF solve, but neither is it treated as zero computation. Reference-label generation is also separate from gradient-based training. We do not reinterpret the archived "total training time" field relative to the 540-second checkpoint, and we do not relabel the archived "Mean RSS" (resident set size) field as peak memory.

The supplementary artifact contains training and evaluation code, generators, configs, seeds, environment and solver/thread settings, raw run records, run ledgers, topology and dataset hashes,

Table 6: Sensitivity to the paired statistical unit using aggregate regret. Topology analysis uses 24 paired means; setting analysis first averages four topologies within each of six settings. Here $\Delta_M = R_{\mathrm{agg},M} - R_{\mathrm{agg,GapCycle}}$, so positive values favor GapCycle; CI denotes confidence interval and NI denotes non-inferiority.

| Method | Aggregate regret | $\Delta_M$ | Topology-level result | Six-setting result |
|---|---|---|---|---|
| PFYL | 0.028945 | +0.008145 | CI excludes 0; Holm $p < 0.05$ | superiority and NI pass; Holm $p < 0.05$ |
| SPO+ | 0.031111 | +0.010311 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| SPO+-N | 0.030863 | +0.010063 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| LTR pairwise | 0.081165 | +0.060365 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| CMAP | 0.082610 | +0.061810 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| LAVA single-basis | 0.093885 | +0.073085 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| nMSE | 0.113916 | +0.093116 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| MSE | 0.171385 | +0.150585 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| WISE | 0.460275 | +0.439475 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| NCE | 0.491166 | +0.470366 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |
| MOM | 0.023338 | +0.002538 | CI excludes 0; Holm $p < 0.05$ | positive advantage; Holm $p < 0.05$ |

Table 7: Registered PFYL superiority and non-inferiority (NI) checks. Superiority is the primary conclusion; NI is a secondary robustness statement. CI denotes confidence interval, and the superiority difference is $d_{\mathrm{PFYL}} = R_{\mathrm{GapCycle}} - R_{\mathrm{PFYL}}$.

| Unit/metric | Superiority result | NI bound | Conclusion |
|---|---|---|---|
| 24 topologies, aggregate regret | CI below zero; Holm $p < 0.05$ | upper bound below registered margin | superiority and NI pass |
| 6 settings, aggregate regret | registered test passes | upper bound below registered margin | superiority and NI pass |
| Per-instance sensitivity | GapCycle lower | registered NI criterion passes | consistent sensitivity result |

configuration/source/analysis hashes, and table/figure regeneration scripts. Analyses use explicit campaign-directory whitelists. The artifact also records a clean-room reproduction check and final hash manifest.

# D Additional Results and Ablations

Unless stated otherwise, the ablations below use the four development topologies and are diagnostic rather than part of the locked confirmation comparison.

## D.1 Confirmation robustness

The statistical unit remains the topology after seed averaging; the three training seeds are not treated as independent observations. The six-setting analysis is a separate sensitivity analysis rather than a device for increasing the effective sample size.

## D.2 Shortest-path breakdown

Table 8 gives the overall 600-fit means. A registered PyEPO/LAVA configuration then illustrates the non-uniform ordering.

PFYL is numerically lower on aggregate regret in this configuration, whereas GapCycle is slightly lower on the per-instance measure. The registered linear, noiseless degree slice gives a second view:

The cellwise ordering therefore varies even though GapCycle has the lowest overall mean across the complete sweep. These experiments support robustness within network-flow structure, not uniform dominance across shortest-path configurations.

## D.3 Regret-definition robustness

Aggregate normalized regret remains the primary metric. The sensitivity analysis changes the per-instance denominator floor over $\varepsilon_R \in \{0.1, 1, 10\}$ and also evaluates absolute regret. Numerical values are reported only where the verified analysis artifact provides them.

Table 8: Overall mean regret across the 600-fit PyEPO shortest-path sweep. Lower is better.

| Method | Mean regret ↓ |
|---|---|
| GapCycle | **0.0544** |
| PFYL | 0.0573 |
| SPO+ | 0.0595 |
| nMSE | 0.0597 |
| MSE | 0.0769 |
| LAVA | 0.1852 |

Table 9: Detailed shortest-path diagnostic for one registered configuration. Lower is better.

| Method | Aggregate regret ↓ | Per-instance regret ↓ |
|---|---|---|
| GapCycle | 0.0548 | **0.0441** |
| PFYL | **0.0507** | 0.0449 |
| SPO+ | 0.0549 | 0.0495 |
| LAVA | 0.0860 | 0.0770 |

### D.4 Margin and one-component ablations

The dedicated $\kappa$ sweep and the one-component $\kappa = 0$ row in Table 13 are distinct registered diagnostics and are not pooled. In the dedicated sweep, $\kappa = 0.5$ is the lowest tested development value, while the $\kappa = 0$ comparison has $p = 0.0746$. The one-component diagnostic uses a different frozen comparison and reports its own confidence interval and $p$-value. Neither experiment establishes that $\kappa = 0.5$ is theoretically optimal.

### D.5 Cycle-loss variants

The frozen $1/|C|$ normalization without bottleneck weighting is lowest among the six tested development variants. This supports the selected surrogate in the registered diagnostic; it does not establish universal optimality of $1/|C|$ or mathematical invalidity of bottleneck-weighted alternatives.

### D.6 Working-set and cache diagnostics

The refresh and capacity comparisons are contained in Table 13. Refreshing every epoch is numerically slightly better than the five-epoch schedule, but its interval includes zero. Capacities 8 and 32 are likewise close to the frozen capacity 16 on this four-topology panel. These tests do not establish equivalence.

The cache audit reports a mean cache-to-oracle gap at most $2 \times 10^{-4}$, an oracle-better fraction at most $10\%$, mean active-cache saturation $0.983$, rediscovery rate $0.048$, mean per-fit peak active-cache storage approximately 2.3 MB, and largest audited active cache approximately 2.7 MB. These are empirical audit quantities rather than theoretical bounds and do not imply $\widetilde{H} = H$ throughout training.

### D.7 Decision-only information setting

GapCycle-DecisionOnly uses $(z, x^\star)$, sets $\kappa = 0$, and removes the nMSE anchor. It is distinct from the $\kappa = 0$ margin ablations, which retain nMSE. The verified result establishes only that its aggregate regret is above the full GapCycle value $0.020800$; no more precise value is inferred. The $\kappa > 0$ regret proposition does not apply to this variant.

### D.8 Extended computational diagnostics

Several methods therefore also use zero downstream MCF training solves. GapCycle's computational distinction is the use of residual-cycle separation to construct a decision-aware training signal,

Table 10: Regret on the registered linear, noiseless shortest-path slice as polynomial degree varies. Lower is better.

| Degree | GapCycle | PFYL | SPO+ | nMSE | LAVA | MSE |
|---|---|---|---|---|---|---|
| 1 | 0.0001 | 0.0001 | 0.0001 | 0.0000 | 0.0001 | 0.0000 |
| 2 | 0.0010 | **0.0009** | 0.0012 | 0.0013 | 0.0014 | 0.0017 |
| 4 | **0.0091** | 0.0095 | **0.0091** | 0.0153 | 0.0141 | 0.0242 |
| 6 | **0.0234** | 0.0235 | **0.0234** | 0.0450 | 0.0385 | 0.0745 |
| 8 | 0.0469 | **0.0430** | 0.0437 | 0.0855 | 0.0655 | 0.1386 |

Table 11: Regret-definition robustness. Aggregate normalized regret is primary; GapCycle retains the leading ordering under every registered normalization.

| Metric | Setting | Conclusion |
|---|---|---|
| Per-instance normalized regret | $\varepsilon_R = 0.1$ | GapCycle leads |
| Per-instance normalized regret | $\varepsilon_R = 1$ | GapCycle leads |
| Per-instance normalized regret | $\varepsilon_R = 10$ | GapCycle leads |
| Absolute regret | no denominator | GapCycle leads |
| Aggregate normalized regret | test-set aggregate (primary) | GapCycle leads at 0.020800 |

not the zero count by itself. We omit the archived total-time and memory columns here because their logged definitions are not reconciled with the 540-second checkpoint and peak-memory terminology in the verified manuscript record.

### D.9 Related-method diagnostics

Single-basis LAVA in the locked comparison and full LAVA with Transition Node Pivoting belong to separate campaigns and are not substituted for one another. CaVE is likewise evaluated as a backend-specific diagnostic. The exact Clarabel backend records approximately $145.8$ s/minibatch and $101$ h/fit on the flow timing diagnostic, whereas APGD records approximately $3.7$ s/minibatch and $2.6$ h/fit. In the California real-feature/synthetic-network diagnostic, the recorded regrets are $0.0669$ for Clarabel and $0.0924$ for APGD; on the shortest-path diagnostic, they are $0.0555$ and $0.0599$, respectively. These diagnostics are kept separate from the locked NETGEN comparison. The California setting uses real covariates with a synthetic downstream flow network and is not presented as a real logistics deployment.

Table 12: Replicated $\kappa$ sensitivity on four development topologies. $\kappa = 0.5$ is selected; $\Delta R$ is regret minus the selected row's regret, and the $\kappa = 0$ comparison has $p = 0.0746$.

| $\kappa$ | Regret | $\Delta R$ vs. $\kappa = 0.5$ | $p$ |
|---|---|---|---|
| 0 | 0.0543 | +0.0144 | 0.0746 |
| 0.25 | 0.0463 | +0.0064 | — |
| 0.5 | 0.0399 | — | selected |
| 1 | 0.1073 | +0.0674 | — |
| 2 | 0.2079 | +0.1680 | — |

Table 13: Each arm changes one component of the frozen configuration. Per-instance validation regret at the 540 s budget on four development topologies, two seeds each, is paired over topologies (three degrees of freedom). $\Delta$ is the arm minus the frozen configuration; positive means removing the component hurts. CI denotes confidence interval.

| Ablation | Regret | $\Delta$ vs frozen | 95% CI | $p$ | Topologies where ablation is better |
|---|---|---|---|---|---|
| frozen configuration | 0.040547 | — | — | — | — |
| $\kappa = 0$ (no margin) | 0.058737 | +0.018190 | [-0.004868, +0.041248] | 0.0869 | 0/4 |
| refresh every epoch | 0.039242 | -0.001304 | [-0.003909, +0.001300] | 0.2092 | 3/4 |
| cache capacity 8 | 0.040332 | -0.000215 | [-0.004723, +0.004294] | 0.8891 | 2/4 |
| cache capacity 32 | 0.040265 | -0.000281 | [-0.001918, +0.001356] | 0.6225 | 2/4 |

$n = 4$ topologies. These intervals are wide by construction; the population claim belongs to the confirmation panel.

Table 14: Cycle normalization and bottleneck weighting on the four development topologies. The frozen configuration is the first row. $\Delta$ is relative to that configuration; positive values are worse.

| Normalizer | Weight | Per-instance regret @540 s | $\Delta$ vs frozen | $p$ |
|---|---|---|---|---|
| $1/\lvert C\rvert$ | none | 0.0399 | — | — |
| $1/\lvert C\rvert$ | bottleneck $\delta_C$ | 0.0435 | +0.0036 | 0.0165 |
| none | none | 0.0415 | +0.0016 | 0.1614 |
| none | bottleneck $\delta_C$ | 0.0442 | +0.0043 | 0.0232 |
| $1/\sqrt{\lvert C\rvert}$ | none | 0.0417 | +0.0018 | 0.1898 |
| $1/\sqrt{\lvert C\rvert}$ | bottleneck $\delta_C$ | 0.0427 | +0.0028 | 0.0716 |

Development panel, four topologies and three degrees of freedom. The bottleneck-weighted variants test capacity-scaled cycle augmentations; they are not identified with GapCycle or with the complete SPO+ inner problem.

Table 15: Ranking cached cycles by the loss-augmented costs $h_i$ rather than the predicted costs, so that retention and eviction agree with the training objective. Four development topologies, two seeds, at the pre-selection learning rate. $\Delta$ is loss-augmented minus predicted-cost scoring, and CI denotes confidence interval.

| Scoring | Per-instance val. regret @540 s | $\Delta$ | 95% CI | $p$ |
|---|---|---|---|---|
| predicted (legacy) | 0.053778 | — | — | — |
| loss-augmented (selected) | 0.051004 | -0.002775 | [-0.005565, +0.000016] | 0.0507 |

Selected by the registered rule. NOT claimed as an established accuracy contribution: at the finally frozen learning rate the same effect shrinks by an order of magnitude.

Table 16: Two probes authorised in advance by registered numeric triggers. Neither improved on the frozen configuration under the registered rule, so the incumbent was retained and no combination cell was needed. Here $k$ is the number of top cached violations aggregated in the loss, and CI denotes confidence interval.

| Probe | Per-instance val. regret @540 s | $\Delta$ vs frozen | 95% CI | $p$ |
|---|---|---|---|---|
| $\alpha = 1$, $k = 1$ (frozen) | 0.040547 | — | — | — |
| $\alpha = 0.1$ | 0.044881 | +0.004334 | [-0.009489, +0.018158] | 0.3919 |
| $\alpha = 0.01$ | 0.140482 | +0.099936 | [-0.075556, +0.275428] | 0.1676 |
| loss_top_k $= 4$ | 0.041324 | +0.000778 | [-0.007154, +0.008709] | 0.7754 |

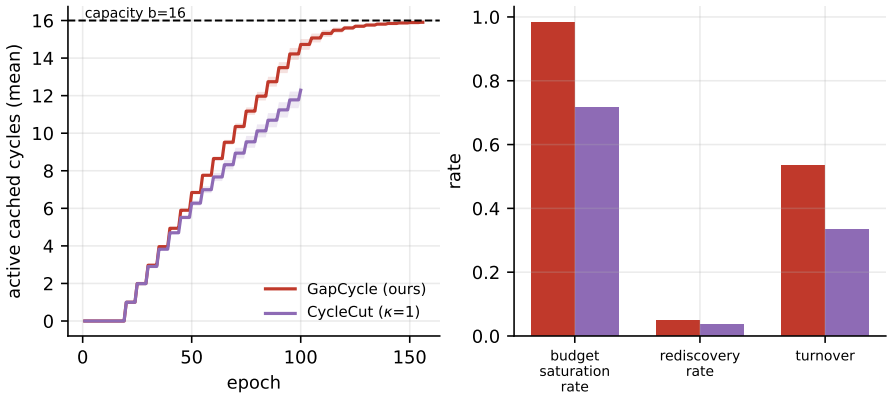



Figure 3: Working-set behavior and exact-separation audit across the registered cache diagnostic. The figure summarizes active-cache saturation, rediscovery, turnover, and oracle quality; it does not convert the cached hinge into the full hinge between separator calls.

Table 17: Information-setting comparison. Aggregate normalized regret is primary; true costs are used only where listed as training information.

| Method | Training information | Aggregate regret | MCF training solves |
|---|---|---|---|
| GapCycle (full) | $z_i, c_i, x_i^\star$ | 0.020800 | 0 |
| GapCycle-DecisionOnly | $z_i, x_i^\star$ | $>$ 0.020800 | 0 |
| LAVA (single basis) | $z_i, x_i^\star$ | 0.093885 | 0 |
| PFYL | full-information implementation | 0.028945 | 1,083,444 |

Table 18: Extended controlled computational comparison. "MCF train solves" counts downstream minimum-cost-flow optimizer calls used to construct the training signal; MMC separator work is recorded separately.

| Method | Agg. norm. regret ↓ | MCF train solves ↓ | s/epoch ↓ |
|---|---|---|---|
| GapCycle | **0.020800** | 0 | 1.4500 |
| MOM | 0.023338 | 0 | 1.5244 |
| PFYL | 0.028945 | 1,083,444 | 7.1474 |
| SPO+-N | 0.030863 | 108,078 | 2.3760 |
| SPO+ | 0.031111 | 92,200 | 2.6382 |
| LAVA (single basis) | 0.093885 | 0 | 2.5954 |
| WISE | 0.460275 | 0 | 1.7616 |
| nMSE | 0.113916 | 0 | 1.7015 |
| MSE | 0.171385 | 0 | 1.7066 |