# PISG: Constraint-Aligned Signal Amplification for Diffusion-Based Combinatorial Optimization

**Yuming Zhang** **Xianchen Zhou** **Hongxia Wang***
College of Science, National University of Defense Technology, Changsha, China
{zhangyuming, zhouxianchen13, wanghongxia}@nudt.edu.cn

## Abstract

Diffusion-based solvers have become a prominent approach to neural combinatorial optimization. At inference time, they assign each decision variable a prediction score. A greedy decoder ranks candidates by these scores and checks hard constraints to ensure feasibility. Top-ranked candidates should be both promising and jointly feasible, making prediction–constraint alignment critical. When this alignment is weak, greedy decoding skips constraint-violating candidates and falls back to lower-ranked alternatives, degrading solution quality. Existing test-time methods improve search, sampling, or decoding, but do not directly strengthen this alignment before decoding. We observe that trained denoisers already carry constraint-aligned structural signals. Building on this observation, we propose Perturbed Instance-Structure Guidance (PISG), a training-free method that amplifies these signals at test time. At each denoising step, PISG constructs a structure-degraded prediction and extrapolates the unperturbed prediction away from it. The resulting guided prediction improves candidate rankings before constrained decoding. Signal analysis shows that this guidance concentrates on constraint-critical variables rather than uniformly rescaling scores. Under greedy decoding and without downstream refinement, PISG improves all 16 TSP solver–scale pairs across DIFUSCO, T2T, FastT2T, and StruDiCO, and all four MIS-ER solvers, with MIS-ER solved-cost gains up to $+4.8\%$. The gains persist on most large-scale settings after 2-opt, and a preliminary study extends PISG to a Transformer-based solver. These results identify prediction–constraint alignment as an effective test-time enhancement direction for diffusion-based combinatorial optimization. Code is available at `https://github.com/Ming-321/PISG`.

## 1 Introduction

Diffusion-based solvers have become a prominent approach to neural combinatorial optimization [Sun and Yang, 2023, Li et al., 2023, 2024, Wang et al., 2025]. At inference time, the denoiser does not directly return the final feasible solution. Instead, it assigns each candidate a prediction score. For the Travelling Salesman Problem (TSP), the candidates are edges that may enter the tour; for the Maximum Independent Set problem (MIS), they are vertices that may enter the independent set. A greedy decoder considers candidates in descending score order and accepts each one that passes hard constraint checks. Feasibility is therefore guaranteed by the decoder, but the final solution quality is still shaped by the ranking produced before decoding. Top-ranked candidates should be both promising and jointly feasible. This makes prediction–constraint alignment before decoding a central factor in solution quality.

A decoder skip is the clearest symptom of weak prediction–constraint alignment. It occurs when a high-ranked candidate conflicts with previously accepted choices and is rejected by the feasibility

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

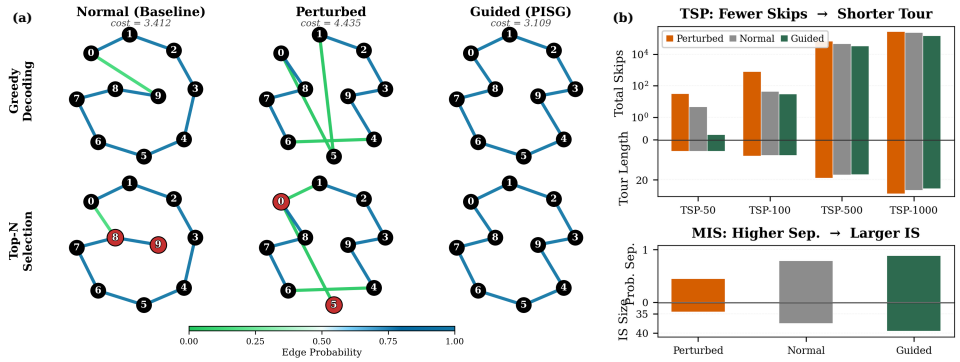


Figure 1: Prediction quality governs decoder skips and final solution quality. **(a)** Schematic 10-node TSP (synthetic, not solver output). *Top row*: greedy tours are always feasible, but lower prediction quality leads to costlier tours ($3.109 < 3.412 < 4.435$). *Bottom row*: top-$N$ selected edges. Normal already concentrates predictions on degree-compatible edges; weakening edge gating (Perturbed) visibly increases violations; Guided further reduces them to zero (no red nodes). **(b)** Quantitative evidence across four TSP scales and MIS: fewer decoder skips correspond to shorter tours and larger independent sets, confirming that prediction quality before decoding governs final solution quality. See Appendix D.1.

check. Figure 1(a) makes this concrete in a schematic TSP-10 instance. High-ranked candidate edges are shown in blue; green edges are lower-ranked fallback alternatives, chosen only after higher-ranked edges fail constraint checks. When feasibility checks skip a high-ranked edge, the decoder continues down the ranked list until it finds an edge that can still be inserted. In this example, fallback edges are longer, so a local ranking error becomes a longer final tour even though feasibility is preserved. Each skip is thus a compromise: in TSP it substitutes a costlier edge; in MIS it forgoes a node that could have enlarged the independent set. Figure 1(b) quantifies this pattern: across four TSP scales, skips decrease and tours shorten monotonically as prediction quality improves (at TSP-50, skips drop from 32.4 to 0.08); MIS probability separation and independent-set size follow the same ordering. These skip patterns motivate an upstream test-time target: improving prediction–constraint alignment before greedy decoding.

Several test-time strategies have improved diffusion-based CO solvers by targeting downstream or trajectory-level components. Objective-guided rewrite refines decoded solutions [Li et al., 2023]; consistency acceleration reduces trajectory length [Li et al., 2024]; structured noising with objective-aware decoding reshapes the sampling process [Wang et al., 2025]. None directly addresses how well the denoiser separates feasible from infeasible candidates at constraint boundaries. We refer to this separation as *constraint-boundary discriminability*: the degree to which the denoiser's predictions distinguish between candidates that should and should not be selected near hard constraints. Higher discriminability means fewer decoder skips, making it an upstream channel that remains largely unexploited.

Through training on feasible solutions, the denoiser produces predictions that already reflect constraint structure before any decoding. The bottom row of Figure 1(a) selects top-$N$ edges by predicted score alone, without any constraint enforcement. Even so, Normal already achieves near degree-2 compliance, closely matching what greedy decoding would accept. This suggests that trained denoisers have implicitly acquired sensitivity to constraint structure, a signal that might be amplified at test time. We study this within the *edge-gated AGNN family*—the denoisers underlying DIFUSCO, T2T, FastT2T, and StruDiCO [Sun and Yang, 2023, Li et al., 2023, 2024, Wang et al., 2025]—which share a structure-sensitive mechanism called *edge gating* (Section 3) [Bresson and Laurent, 2017, Sun and Yang, 2023]. Edge gating is the primary pathway through which instance-specific graph structure enters the prediction. Figure 1(a) shows this in a synthetic example: weakening edge gating produces predictions with more degree violations and a costlier greedy tour. Since this sensitivity enters through edge gating, weakening it should expose the direction along which instance structure shapes predictions.

We propose *Perturbed Instance-Structure Guidance (PISG)* to exploit this structural sensitivity. It follows the perturb-and-extrapolate form of visual diffusion guidance [Ahn et al., 2024, Hong

et al., 2023], but changes the perturbation target to edge gating, the instance-structure interface in edge-gated AGNN denoisers. The extrapolation rule is thus inherited; our contribution is a CO-specific intervention on the structure pathway, together with evidence that the direction it induces is constraint-aligned. PISG replaces the learned edge gates with a weakened surrogate, forming a structure-degraded branch that shares all inputs and parameters with the normal branch. Extrapolating the normal prediction away from the degraded one amplifies the structure-dependent component of each denoising step. We verify in Section 5 that this direction is constraint-aligned: it concentrates at task-specific constraint boundaries rather than rescaling predictions isotropically. PISG is training-free and uses no task objective, but it is not tuning-free: its scale and perturbation strategy are selected by a modest validation sweep for each task–distribution pair. The two parameter-sharing branches can be batched during inference.

Our experiments examine PISG at the signal level and after greedy decoding. Signal characterization shows that the extrapolation direction carries constraint-boundary discriminability: in TSP, it separates degree-2 boundary edges, and in MIS, it separates conflicting vertex choices (Section 5). Under greedy decoding without objective-guided refinement, PISG improves all 16 TSP solver–scale pairs and all four MIS-ER solvers, with MIS-ER solved-cost gains up to $+4.8\%$. Additional experiments show that this constraint-aligned signal is mechanistically distinct from objective-side refinement; where residual headroom remains, combining the two can yield further gains (Section 6.2). With 2-opt added, gains persist in three of four large-scale cells but become much smaller, and a preliminary transfer to a Transformer solver shows similarly condition-dependent effects. Our contribution is threefold. First, we identify prediction–constraint alignment as a pre-decoding bottleneck in diffusion-based combinatorial optimization and formalize its local manifestation as constraint-boundary discriminability. Second, we propose PISG, a training-free guidance method that perturbs edge-gated instance-structure processing and extrapolates away from the resulting degraded prediction at each denoising step, before greedy decoding. Third, we show that PISG amplifies constraint-aligned signals at TSP and MIS constraint boundaries rather than uniformly rescaling predictions; under greedy decoding without downstream refinement, this yields gains across all four edge-gated solvers.

## 2 Related Work

**Solver-level advances in diffusion-based CO.** DIFUSCO introduced discrete diffusion over graph representations for CO [Sun and Yang, 2023]. T2T added objective-guided rewrite ($T_g$) that iteratively improves solutions through gradient-guided denoising [Li et al., 2023]. FastT2T replaced iterative diffusion with optimization consistency training, significantly shortening inference [Li et al., 2024]. StruDiCO redesigned the noise and reverse processes via variable-absorption noising and introduced gradient-free objective-aware refinement [Wang et al., 2025]. Each targets a different axis of the pipeline: training targets, noise processes, or inference strategy. Yet all four build on the same edge-gated AGNN backbone [Bresson and Laurent, 2017] and follow a common skeleton of denoising followed by greedy decoding. Concurrent work extends this line with search-driven generation [Li et al., 2025], adaptive expansion [Ma et al., 2025], masked generation and diffusion [Chen et al., 2026, Wang et al., 2026], and energy-guided adaptation [Lei et al., 2026]. All of these innovations target trajectory design, refinement policy, or search strategy, treating the denoiser's per-step predictions as given. None explicitly targets the denoiser's constraint-boundary discriminability at the prediction level.

**Denoiser-internal guidance.** Image generation provides a precedent: several methods modify the denoiser's internal computation to steer sampling. Classifier guidance steers diffusion sampling with an external gradient [Dhariwal and Nichol, 2021]. Classifier-free guidance (CFG) replaces it by contrasting conditional and unconditional predictions but requires training-time modifications [Ho and Salimans, 2022]. Recent CFG extensions for discrete diffusion share this training requirement [Schiff et al., 2025, He et al., 2026]. Training-free variants go further: PAG, SAG, SEG, and CDG perturb internal components (e.g., self-attention maps or condition embeddings) to construct a degraded branch and extrapolate [Ahn et al., 2024, Hong et al., 2023, Hong, 2024, Han et al., 2026]; see also [Ahn et al., 2025, Bansal et al., 2024, Li et al., 2026, Ye et al., 2024]. These training-free methods share a common design template: they construct a degraded version of the denoiser by perturbing the mechanism that selectively routes information based on input content or structure (self-attention in image U-Nets and DiTs), then extrapolate away from the degraded prediction. In graph-based

CO denoisers, learned edge gates rather than self-attention serve as the primary information-routing mechanism, selectively scaling neighbor contributions during message passing. Whether perturbing this pathway can reveal constraint-aligned sensitivity in CO denoisers has not been tested. We investigate this question within the edge-gated AGNN family (DIFUSCO, T2T, FastT2T, StruDiCO), where the shared gating mechanism enables controlled isolation of the denoiser-internal effect. We additionally report a preliminary transfer to GenSCO [Li et al., 2025], which routes structure through attention biases. Our signal characterization shows that the resulting signal concentrates at constraint boundaries rather than rescaling isotropically, connecting the visual-domain template to a CO-specific, constraint-aligned finding.

## 3 Preliminaries

**Task definitions.** Given a graph $G = (V, E)$, a combinatorial optimization (CO) problem seeks a binary vector $\mathbf{x} \in \{0, 1\}^N$ that optimizes an objective $f(\mathbf{x}; G)$ subject to feasibility constraints $\Omega$. **TSP** is an edge-selecting problem on a complete graph of $n$ Euclidean nodes. The goal is to find a minimum-cost Hamiltonian cycle. Every node must be incident to exactly two tour edges (the *degree-2* constraint). **MIS** is a node-selecting problem on a given graph. The goal is to find a largest independent set. No two adjacent nodes may be simultaneously selected (*pairwise exclusion*). These constraints define the hard feasibility checks used by the constrained decoder and the constraint boundaries analyzed in Section 5: the degree-2 boundary in TSP and adjacency conflicts in MIS.

**Diffusion-based CO pipeline.** Recent diffusion and consistency solvers learn solution distributions conditioned on the instance graph [Sun and Yang, 2023, Li et al., 2023, 2024, Wang et al., 2025]. Inference starts from a solver-specific terminal state $\mathbf{x}_{T_s}$ (uniformly random binary for standard discrete-diffusion solvers; see Appendix B for StruDiCO) and proceeds through $T_s$ denoising steps. At each step $t$, a trained denoiser receives the noisy state $\mathbf{x}_t$, the denoising index $t$, and the instance graph $G$, and outputs $N$ per-element logit scores $\mathbf{p}_t$ estimating the clean solution $\mathbf{x}_0$. After the final step, these scores feed the greedy constrained decoder. Optional $T_g$ objective-guided rewrite steps operate downstream of decoding [Li et al., 2023, 2024, Wang et al., 2025].

**AGNN edge gating.** The denoisers in DIFUSCO, T2T, FastT2T, and StruDiCO share an anisotropic GNN backbone [Bresson and Laurent, 2017]. Each layer updates node features via edge-gated message passing

$$\mathbf{h}_i^{(\ell+1)} = \mathbf{h}_i^{(\ell)} + \mathrm{ReLU}\Big(\mathrm{LN}\Big(\mathbf{U}^{(\ell)}\mathbf{h}_i^{(\ell)} + \mathrm{Aggr}_{j\in\mathcal{N}(i)}\Big[\sigma\Big(\hat{\mathbf{e}}_{ij}^{(\ell+1)}\Big) \odot \mathbf{V}^{(\ell)}\mathbf{h}_j^{(\ell)}\Big]\Big)\Big), \qquad (1)$$

where $\hat{\mathbf{e}}_{ij}^{(\ell+1)}$ is a pre-activation edge feature aggregating node states and edge attributes via learned projections [Sun and Yang, 2023], $\mathbf{U}^{(\ell)}$ and $\mathbf{V}^{(\ell)}$ are node projection matrices, and LN denotes layer normalization. We denote the *edge gate* $\mathbf{g}_{ij}^{(\ell)} = \sigma(\hat{\mathbf{e}}_{ij}^{(\ell+1)})$; it selectively scales each neighbor's projected contribution $\mathbf{V}^{(\ell)}\mathbf{h}_j^{(\ell)}$, determining which pairwise relations the denoiser emphasizes at each layer. We collect all layer–edge gates into a map $\Gamma = \{\mathbf{g}_{ij}^{(\ell)}\}_{\ell,(i,j)}$, the structure-sensitive internal pathway that PISG perturbs in Section 4.

## 4 Method: Constraint-Aligned Signal Amplification via PISG

PISG is a training-free mechanism for constraint-aligned signal amplification before constrained decoding. As shown in Figure 2, it targets the denoiser's edge-gated instance-structure pathway and constructs a structure-degraded branch at each denoising step. The normal-minus-perturbed discrepancy is then used to amplify the structure-dependent component of the prediction before the decoder ranks candidates.

**Why target edge gating?** In the AGNN backbone (Section 3), the edge gate is the primary pathway through which instance-specific graph structure shapes per-step predictions. Weakening this pathway degrades structure-sensitive message routing while keeping the input graph, noisy state, model parameters, and decoder unchanged. The resulting difference is designed to isolate the denoiser's structure-dependent sensitivity. If this sensitivity is aligned with hard constraints, the

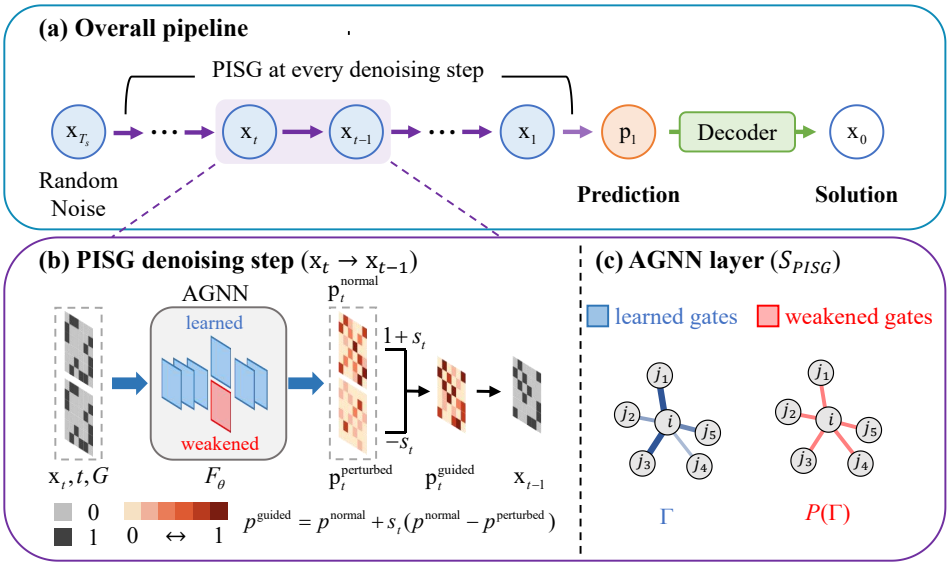


Figure 2: Overview of PISG. (a) PISG is applied at every denoising step to form guided logits for the reverse transition. (b) At step $t$, PISG computes two parallel, parameter-sharing denoising branches: a normal branch with learned AGNN gates and a structure-degraded branch with weakened gates. Their logit discrepancy defines the PISG signal, and the guided logits extrapolate along this direction (Equation 2) for sampling $x_{t-1}$. (c) In selected AGNN layers, PISG replaces heterogeneous learned edge gates with spatially uniform weakened surrogates, removing edge-wise structural sensitivity.

normal-minus-perturbed direction should improve constraint-boundary discriminability, which we test in Section 5.

### 4.1 Two-Branch Construction and Guidance Signal

Let $F_\theta(\mathbf{x}_t, t, G; \Gamma)$ denote a denoiser forward pass that receives the noisy state $\mathbf{x}_t$, denoising step $t$, instance graph $G$, and the internal edge-gate map $\Gamma = \{\mathbf{g}_{ij}^{(\ell)}\}$. The notation $F_\theta(\cdot; \Gamma)$ indicates that the realized gates inside selected AGNN layers are either kept unchanged or replaced by a perturbed gate map during the forward pass. Let $\mathcal{P}_{\mathcal{S}_{\text{pisg}}}$ denote a perturbation operator that weakens gates in the layer set $\mathcal{S}_{\text{pisg}}$ while leaving other layers unchanged. Here $\mathbf{p}$ denotes pre-sigmoid/pre-softmax logits (not probabilities). All PISG operations are performed in logit space. At each denoising step, PISG evaluates two parameter-sharing branches:

$$\mathbf{p}_t^{\text{normal}} = F_\theta(\mathbf{x}_t, t, G; \Gamma_t), \qquad \mathbf{p}_t^{\text{perturbed}} = F_\theta(\mathbf{x}_t, t, G; \mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t)),$$

where $\Gamma_t$ denotes the gate map realized at step $t$. The perturbed branch degrades only the gate-mediated structural selection inside the denoiser; it does not alter the instance graph, the noisy state, the task objective, or the constrained decoder. The guided prediction extrapolates along the direction from perturbed to normal:

$$\mathbf{p}_t^{\text{guided}} = \mathbf{p}_t^{\text{normal}} + s_t\,\boldsymbol{\delta}_t = \mathbf{p}_t^{\text{normal}} + s_t\big(\mathbf{p}_t^{\text{normal}} - \mathbf{p}_t^{\text{perturbed}}\big), \tag{2}$$

where $s_t$ follows a guidance schedule (Section 4.2). We call $\boldsymbol{\delta}_t = \mathbf{p}_t^{\text{normal}} - \mathbf{p}_t^{\text{perturbed}}$ the *PISG guidance signal*. It measures how the current prediction changes when structure-sensitive gate selectivity is weakened. The guided logits $\mathbf{p}_t^{\text{guided}}$ replace the normal logits in reverse sampling and in the final candidate ranking before constrained decoding; the decoder's feasibility checks themselves are unchanged.

The guidance signal $\boldsymbol{\delta}_t$ is a finite intervention on gate selectivity, not an injection of external constraint knowledge. Although the perturbation operator itself is task-agnostic, any task-specific pattern in $\boldsymbol{\delta}_t$ must arise from the trained denoiser's gate pathway rather than from externally injected information. This motivates a testable prediction: if the denoiser's gate sensitivity is constraint-aligned, $\boldsymbol{\delta}_t$ should concentrate at task-specific constraint boundaries rather than uniformly rescaling logits (a local-linearization derivation supporting this view appears in Appendix A.3).

### 4.2 Perturbation Operators and Inference

Equation (2) leaves three design choices open: how to construct the perturbed gate map $\mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t)$, how to schedule the guidance strength $s_t$, and which layers to perturb through $\mathcal{S}_{\text{pisg}}$. The perturbed branch should be degraded specifically along the instance-structure pathway, preserving all other pipeline components. This keeps the normal-minus-perturbed direction interpretable as a structure-sensitive signal before decoding.

**Perturbation operators.** For layers $\ell \in \mathcal{S}_{\text{pisg}}$, the perturbation operator replaces each gate vector $\mathbf{g}_{ij}^{(\ell)}$ by a spatially non-selective surrogate:

$$\mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t): \quad \mathbf{g}_{ij}^{(\ell)} \mapsto \begin{cases} c \cdot \mathbf{1} & (\texttt{uniform}), \\ \mu_t^{(\ell)} \cdot \mathbf{1} & (\texttt{adaptive}), \end{cases} \qquad \ell \in \mathcal{S}_{\text{pisg}}, \tag{3}$$

where $c = \frac{1}{2}$ by default and $\mu_t^{(\ell)} = \frac{1}{|E|} \sum_{(i',j') \in E} \text{mean}(\mathbf{g}_{i'j'}^{(\ell)})$ averages over all edges and hidden channels for the current instance. Both operators remove edge-wise spatial selectivity, i.e., which pairwise relations receive stronger message passing. They differ only in whether the replacement magnitude is fixed globally (`uniform`) or adapted to the current instance and layer (`adaptive`). An additional strategy (random-gate) is compared in Appendix A.

**Guidance schedule.** The guidance schedule decomposes the per-step strength into a global scale $s$ and a temporal weight $w(t)$:

$$s_t = s \cdot w(t), \qquad w(t) = \begin{cases} 1 & (\texttt{const}), \\ \dfrac{T_s - t}{T_s - 1} & (\texttt{linear},\ T_s > 1). \end{cases} \tag{4}$$

Here $t$ indexes the denoising step ($t = T_s, \ldots, 1$), distinct from the underlying diffusion timestep. Empirically, constant scheduling works best for MIS and linear for TSP (Section 6.1).

**Layer selection.** The perturbation need not span all GNN layers. A layer set $\mathcal{S}_{\text{pisg}} \subseteq \{1, \ldots, L\}$ restricts which layers have their gating replaced. In our 12-layer backbones, middle-to-late layers (roughly layers 5–9) form a broad safe zone where most configurations improve over the baseline. The sensitivity analysis is summarized in Section 6.1 and detailed in Appendix C.1.

These choices control where and how strongly the same structural discrepancy is amplified. They do not introduce objective information or modify feasibility checks; their role is to expose prediction–constraint alignment already present in the trained denoiser. The two branches share all parameters and can be evaluated by batch doubling, with pseudocode in Algorithm 1 (Appendix A). Section 5 empirically tests the resulting directional structure, and Section 6 evaluates PISG across all four solvers, with overhead measurements in Section 6.3. Edge-gating perturbation is one instance of a procedure: identify the structure pathway through which instance structure enters the prediction, degrade it, and extrapolate; Appendix A.5 gives a second instance, attention-bias perturbation in a Transformer solver.

## 5 Signal Analysis: Constraint-Boundary Discriminability

Section 4 motivates a testable prediction: $\boldsymbol{\delta}_t$ should concentrate at task-specific constraint boundaries rather than rescaling predictions uniformly. We test this through three diagnostics: local degree-boundary alignment in TSP, conflict-boundary separation in MIS, and a temperature-scaling negative control (reported in Appendix D.4). All results aggregate over 5 seeds (41–45, matching the main tables). PISG extrapolates along the logit-space $\boldsymbol{\delta}_t$ (Equation (2)); because decoding acts on probabilities, the diagnostics measure its probability-space counterpart $\tilde{\boldsymbol{\delta}}_t = \psi(\mathbf{p}_t^{\text{normal}}) - \psi(\mathbf{p}_t^{\text{perturbed}})$, where $\psi$ is the solver's standard logit-to-probability transform, and rank candidates by normal-branch probability.

### 5.1 TSP: Degree-2 Split

A valid TSP tour requires every node to connect to exactly two selected edges:

$$\textstyle\sum_{j:(i,j) \in E} x_{ij} = 2 \quad \forall\, i \in V. \tag{5}$$

We focus on the degree-2 boundary as the local constraint boundary most directly exposed in edge rankings; global subtour elimination is not directly measured by this diagnostic. Under the denoiser's current ranking, the top-two incident edges form the local degree-compatible candidate set. If $\tilde{\boldsymbol{\delta}}_t$ reflects improved degree-boundary discriminability, it should reinforce these two candidates and suppress lower-ranked incident edges, since accepting additional incident edges would breach the degree-2 limit.

We rank each node's incident edges by normal-branch probability and compute mean $\tilde{\boldsymbol{\delta}}_t$ per rank position across all nodes. Figure 3(a) confirms the predicted pattern: at step 3, mean $\tilde{\boldsymbol{\delta}}_t$ is positive for the top-two ranked edges and negative from the third-ranked edge onward; the sign reversal falls precisely at the degree-2 boundary. This indicates that PISG sharpens competition at the constraint boundary rather than uniformly amplifying the current leader. The same directional pattern persists across denoising steps with varying magnitude (Appendix D.2).

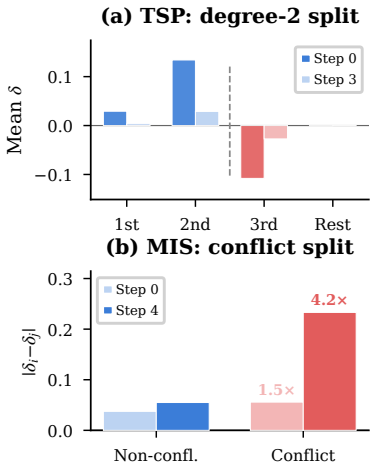


Figure 3: Probability-space signal analysis ($\tilde{\boldsymbol{\delta}}_t$). (a) TSP: positive for top-2 edges, negative beyond (degree-2 split). (b) MIS: conflict-edge endpoint separation is $4.2\times$ non-conflict at step 4.

### 5.2 MIS: Conflict-Edge Split

MIS imposes a structurally different constraint. Two adjacent nodes cannot both belong to an independent set.

$$x_i + x_j \leq 1 \quad \forall\,(i,j) \in E. \tag{6}$$

We call an edge a *predicted conflict edge* when both endpoints have transformed normal-branch scores above $\vartheta{=}0.3$; these are the decisions where the decoder must choose one endpoint over the other. If $\tilde{\boldsymbol{\delta}}_t$ reflects improved conflict-boundary discriminability, it should pull apart the scores of conflicting endpoints.

Unlike TSP, where the signal manifests as a sign reversal, MIS $\tilde{\boldsymbol{\delta}}_t$ is predominantly positive for high-score nodes. The constraint signal instead appears as endpoint-separation asymmetry between predicted conflict and non-conflict edges. Figure 3(b) shows this clearly. The endpoint separation $|\tilde{\delta}_i - \tilde{\delta}_j|$ on conflict edges is $1.5\times$ that of non-conflict edges at step 0 and grows to $4.2\times$ at step 4. This growing asymmetry shows that the guidance signal concentrates on the most contested constraint decisions as predictions sharpen, giving the greedy decoder a stronger cue for which node to retain (threshold robustness in Appendix D.3).

Together, the degree-2 sign reversal in TSP and the conflict-edge asymmetry in MIS show that the guidance direction improves constraint-boundary discriminability in task-specific ways, supporting the claim that PISG performs directional, constraint-aligned signal amplification rather than uniform score sharpening. A temperature-scaling negative control confirms that scalar rescaling cannot produce these patterns (Appendix D.4). Cross-solver replication and trajectory-level probes appear in Appendices D.5–D.6. These diagnostics characterize the geometry of the guidance direction rather than per-instance benefit. After controlling for baseline quality, the degree-2 sign-compliance rate (sc@2; Appendix D.2) no longer tracks final TSP quality, whereas the conflict-edge magnitude asymmetry (CMA; Appendix D.3) retains an association with final MIS quality; neither association is causal (Appendix D.7). The TSP evidence concerns the local degree-2 boundary: a fragment-closing probe finds no subtour-level signal (Appendix D.8).

## 6 Experiments

We next ask whether the constraint-boundary discriminability observed at the signal level translates into better feasible solutions after constrained decoding. The primary evaluation therefore disables downstream objective-guided refinement ($T_g{=}0$) and local search, so that the comparison focuses on the predictions passed to constrained decoding.

We evaluate PISG on all four solvers of the edge-gated AGNN family (Section 2): DIFUSCO [Sun and Yang, 2023] ($T_s{=}100$), T2T [Li et al., 2023] ($T_s{=}50$), FastT2T [Li et al., 2024] ($T_s{=}5$), and StruDiCO [Wang et al., 2025] ($T_s{=}5$). Benchmarks cover TSP at four scales (50–1,000 nodes; gap%

Table 1: TSP results across four solvers and four scales. Gap% = (cost − Concorde) / Concorde × 100. **Bold**: best gap within each solver block and scale.

| | | TSP-50 | | | TSP-100 | | | TSP-500 | | | TSP-1000 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | $T_s/T_g$ | Cost | Gap% | Time | Cost | Gap% | Time | Cost | Gap% | Time | Cost | Gap% | Time |
| Concorde | – | 5.688 | 0.000 | – | 7.756 | 0.000 | – | 16.55 | 0.00 | – | 23.12 | 0.00 | – |
| DIFUSCO | 100/0 | 5.708 | 0.367 | 6.2m | 7.864 | 1.388 | 6.7m | 18.52 | 11.91 | 3.8m | 26.19 | 13.31 | 9.6m |
| +PISG | 100/0 | 5.699 | **0.203** | 6.2m | 7.800 | **0.567** | 10.2m | 18.11 | **9.44** | 5.8m | 25.65 | **10.95** | 16.7m |
| T2T | 50/0 | 5.727 | 0.683 | 3.5m | 7.882 | 1.624 | 3.6m | 18.42 | 11.33 | 76s | 26.13 | 13.05 | 2.4m |
| +PISG | 50/0 | 5.719 | **0.543** | 3.5m | 7.815 | **0.758** | 5.0m | 18.18 | **9.88** | 96s | 26.09 | **12.87** | 3.6m |
| FastT2T | 5/0 | 5.689 | 0.023 | 45s | 7.762 | 0.079 | 52s | 17.51 | 5.84 | 32s | 24.61 | 6.46 | 48s |
| | 5/5 | 5.688 | **0.0069** | 135s | 7.758 | 0.025 | 154s | 16.92 | 2.27 | 68s | 23.99 | 3.78 | 177s |
| +PISG | 5/0 | 5.689 | 0.017 | 48s | 7.760 | 0.051 | 55s | 17.40 | 5.15 | 33s | 24.49 | 5.93 | 56s |
| | 5/5 | 5.688 | 0.0071 | 141s | 7.758 | **0.023** | 154s | 16.87 | **1.94** | 69s | 23.86 | **3.22** | 182s |
| StruDiCO | 5/0 | 5.689 | 0.027 | 45s | 7.770 | 0.187 | 46s | 17.34 | 4.81 | 33s | 25.83 | 11.75 | 46s |
| | 5/5 | 5.688 | 0.0125 | 71s | 7.758 | **0.031** | 77s | 17.00 | 2.71 | 42s | 25.11 | 8.60 | 74s |
| +PISG | 5/0 | 5.689 | 0.025 | 48s | 7.770 | 0.179 | 53s | 17.33 | 4.74 | 37s | 25.43 | 9.99 | 58s |
| | 5/5 | 5.688 | **0.0123** | 70s | 7.758 | 0.031 | 82s | 16.95 | **2.45** | 46s | 24.67 | **6.72** | 86s |

to Concorde [Applegate et al., 2006], lower is better) and MIS on two synthetic distributions: sparse Erdős–Rényi (ER) graphs and denser RB-model graphs (solved cost, higher is better). The controlled-isolation results report 5-seed means under $T_g$=0, greedy decoding (StruDiCO retains its native decoders), and no local search; Tables 1–2 additionally include $T_g$>0 rows to evaluate interaction with objective-side refinement. The $T_g$=0 setting is an identification design, not a deployment recommendation. Section 6.2 then evaluates the deployment pipeline with objective rewrite and 2-opt, and a preliminary transfer to a Transformer solver. Full details appear in Appendix B.

### 6.1 Main Results under Controlled Isolation

Table 1 reports TSP results. In the controlled $T_g$=0 rows, PISG improves all 16 solver–scale combinations. The largest absolute reductions occur in solvers with the largest residual headroom, notably DIFUSCO and T2T: on TSP-100, DIFUSCO gap drops from $1.39\%$ to $0.57\%$, and T2T drops from $1.62\%$ to $0.76\%$. Even near-optimal short-trajectory baselines improve: FastT2T gap on TSP-100 falls from $0.079\%$ to $0.051\%$.

Table 2: MIS results on ER and RB. Cost is higher-better; gap% is lower-better and negative means exceeding the reference. **Bold**: best within each solver block.

| | | ER | | | RB | | |
|---|---|---|---|---|---|---|---|
| Method | $T_s/T_g$ | Cost↑ | Gap%↓ | Time | Cost↑ | Gap%↓ | Time |
| Reference | – | 41.35 | 0.00 | — | 20.09 | 0.00 | — |
| DIFUSCO | 100/0 | 40.63 | 1.73 | 4.0m | — | — | — |
| +PISG | 100/0 | 42.55 | **−2.91** | 6.5m | — | — | — |
| T2T | 50/0 | 36.50 | 11.74 | 1.5m | — | — | — |
| +PISG | 50/0 | 38.25 | **7.50** | 2.3m | — | — | — |
| FastT2T | 5/0 | 37.65 | 8.92 | 33s | 18.81 | 6.38 | 38s |
| | 5/5 | 40.98 | 0.87 | 62s | 19.47 | 3.07 | 93s |
| +PISG | 5/0 | 38.89 | 5.92 | 38s | 18.89 | 5.98 | 41s |
| | 5/5 | 41.86 | **−1.25** | 67s | **19.49** | **2.98** | 96s |
| StruDiCO | 5/0 | 39.89 | 3.51 | 35s | 19.30 | 3.93 | 48s |
| | 5/5 | 42.46 | −2.72 | 49s | 19.74 | 1.77 | 73s |
| +PISG | 5/0 | 40.05 | 3.11 | 41s | 19.31 | 3.90 | 50s |
| | 5/5 | 42.55 | **−2.93** | 58s | 19.75 | **1.72** | 79s |

Table 2 reports MIS results. On ER, all four solvers improve: DIFUSCO gains $+4.7\%$ in solved cost and obtains a negative gap relative to the reported reference; T2T and FastT2T reduce their gaps by $4.2$ and $3.0$ percentage points, respectively. On RB, where only FastT2T and StruDiCO have pretrained checkpoints, FastT2T improves while StruDiCO is effectively neutral. Per-seed standard deviations and statistical significance tests are reported in Appendix B.3. With five seeds, 16 of the 22 comparisons reach $p<0.05$; in a 20-seed follow-up of the remaining six, four reach $p<0.05$ (T2T TSP-1000 and StruDiCO TSP-100/500/MIS-ER), StruDiCO TSP-50 remains inconclusive, and StruDiCO MIS-RB is effectively null.

On TSP, absolute gains scale with residual headroom: solvers with larger baseline gaps (DIFUSCO, T2T) benefit most. On both tasks, StruDiCO—which already exploits structure aggressively—shows smaller improvements that require lower guidance strength (Section 6.1).

**Hyperparameter sensitivity.** PISG is training-free but not tuning-free. Family defaults are identified from a full sensitivity analysis on FastT2T (Appendix C.1) and checked on held-out validation splits; they transfer to DIFUSCO and T2T without re-tuning. The selected defaults are: `adaptive`/$s$=2.0/L7/`linear` for TSP, `uniform`/$s$=2.0/L7/`constant` for MIS-ER, and `adaptive`/$s$=2.0/L7/`constant` for MIS-RB. On the validation splits, the MIS-ER default is also the best configuration, while the TSP-100 default improves over the baseline; the TSP default also transfers across scales. Full sweeps in Appendix C.1 show a broad safe region: on TSP-100, $82\%$ of 180 gate-level configurations improve over the baseline and all configurations in the central region (scale $\in [1, 3]$, layers $\in [5, 9]$) improve. StruDiCO is a systematic exception, requiring guidance strength an order of magnitude lower ($s$=0.1–0.2), consistent with its stronger built-in structure utilization (Appendix C). Frozen configurations transfer within the coverage of the training distribution but fail at a high-density MIS endpoint, in most extrapolations to larger TSP scales, and on large TSPLIB instances (Appendix C.4).

### 6.2 Deployment Pipeline and Transfer

**Objective-side refinement.** The $T_g$>0 rows in Tables 1–2 combine PISG with objective-side refinement for FastT2T and StruDiCO. The interaction is headroom-dependent: at larger scales (TSP-500/1000), PISG+$T_g$ consistently improves over baseline+$T_g$; at smaller scales where the baseline already approaches optimality, the combination is near-neutral. On MIS-ER, PISG+$T_g$ improves throughout: FastT2T at $T_g$=5 obtains a negative gap relative to the reported reference ($-1.25\%$). This supports the pipeline separation introduced in the Introduction: PISG acts before constrained decoding by improving prediction–constraint alignment, whereas $T_g$ acts through objective-side refinement. Their complementarity therefore depends on residual headroom rather than on redundant score sharpening.

**Full pipeline with 2-opt.** The settings above omit local search. In the full pipeline with $T_g$=5 and 2-opt run to convergence, the frozen configurations retain a positive mean effect in three of four large-scale cells (Appendix B.4). On FastT2T, PISG removes $6.1\%$ and $4.5\%$ of the remaining gap on TSP-500 and TSP-1000; StruDiCO TSP-500 retains a small gain, whereas StruDiCO TSP-1000 reverses at the converged 2-opt endpoint. These gains are much smaller than under controlled isolation, consistent with downstream refinement absorbing part of the residual headroom.

**Preliminary transfer to a Transformer solver.** We also apply PISG to GenSCO [Li et al., 2025], a Transformer solver in which instance structure enters through an adjacency bias on attention logits rather than through edge gating. The structure-degraded branch weakens this attention-bias pathway at selected layers, and configurations are frozen across search budgets (Appendix A.5). On TSP, PISG yields positive mean effects in 10 of 12 cells, including all six TSP-1000 cells, while TSP-500 with in-loop 2-opt reverses at two of three budgets. On MIS, gains appear only on ER at the two smaller budgets and only together with a higher search flip rate; RB shows no measurable effect. These results are preliminary evidence from one Transformer solver with five seeds.

### 6.3 Compute–Quality Trade-off

PISG adds one perturbed forward pass per denoising step; the normal and perturbed branches are batched. At $T_g$=0, this adds 3–17% wall-clock overhead on FastT2T and 4–26% on StruDiCO (5-seed mean across all tasks; Appendix B.5). Thus PISG is a targeted pre-decoding alignment step with moderate but nonzero inference cost.

Figure 4 uses FastT2T as a representative short-trajectory testbed that enables dense $T_s$ and $T_g$ sweeps. Along the $T_s$ axis (solid lines), PISG improves over the baseline from $T_s$=5 onward (from $T_s$=3 on MIS-ER). On large-scale TSP (500/1000), PISG at $T_s$=5 achieves lower gap than the baseline at $T_s$=10 at comparable or lower wall-clock cost (Appendix B.5). On TSP-50/100 it beats only the closest-time baseline, and on both MIS distributions adding denoising steps is more effective than PISG. Along the $T_g$ axis (dashed lines), PISG+$T_g$ reaches or exceeds baseline+$T_g$ at higher rewrite

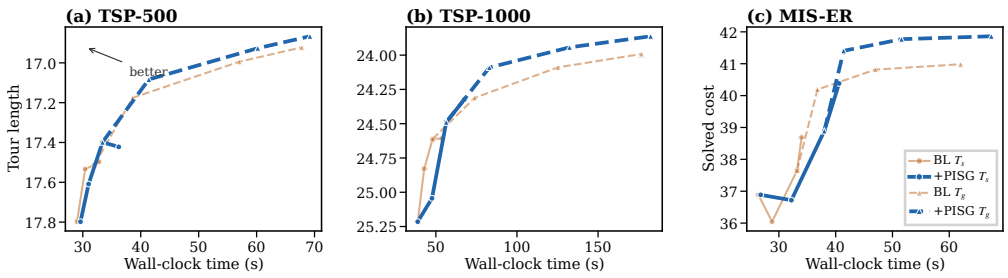


Figure 4: Compute–quality trade-off on FastT2T. Solid: $T_s$ scaling ($T_g$=0); dashed: $T_g$ scaling ($T_s$=5). PISG shifts the frontier in later-budget regimes and adds 3–17% wall-clock overhead (Appendix B.5).

budgets on TSP-500 and TSP-1000. On MIS-ER, PISG+$T_g$ improves throughout the tested range, though early budget points overlap.

## 7 Conclusion and Limitations

We introduced PISG, a training-free denoiser-internal guidance mechanism that reveals and amplifies constraint-aligned sensitivity in edge-gated AGNN denoisers. Signal analysis shows that task-agnostic edge-gating perturbations expose task-specific constraint-boundary discriminability—a degree-2 boundary split in TSP and conflict-edge endpoint separation in MIS—rather than uniform score rescaling. Under controlled isolation, PISG improves every tested solver–scale combination on TSP and every solver on MIS-ER (solved cost up to $+4.8\%$). Constraint-aligned signal amplification and objective-side refinement ($T_g$) are mechanistically distinct and non-redundant; where residual headroom remains, their combination yields further improvement. After 2-opt, the benefit persists in three of four large-scale cells, and attention-bias perturbation gives a preliminary transfer to the Transformer-based GenSCO.

Several limitations apply. The extra forward pass at each denoising step raises peak GPU memory, and the method offers limited value when $T_s$ is small—on MIS, allocating the same compute to additional denoising steps is more effective. PISG is training-free but not tuning-free: a validation sweep ($\sim$50 configurations) is needed for each task–distribution pair, and frozen configurations can fail under larger density or scale shifts. The constraint-aligned interpretation rests on empirical, partly correlational evidence; the TSP diagnostics cover only the local degree-2 boundary and lack subtour-level signal, while per-instance predictive power of the proposed metrics remains limited (Appendices D.7–D.8). Finally, edge-gating and attention-bias perturbation are two instances of degrading a structure pathway; extending to other backbones, constraint systems, or objectives the model has not been trained on would require further investigation.

## Acknowledgments and Disclosure of Funding

We would like to express our gratitude to Prof. Guangquan Cheng and Changjun Fan for their insightful comments. The work was supported by the National Natural Science Foundation of China (No. 12401677 and 12471401), and Youth Science Fund of National University of Defense Technology (No. ZK23-07). The authors declare no competing interests.

## References

D. Ahn, H. Cho, J. Min, W. Jang, J. Kim, S. Kim, H. H. Park, K. H. Jin, and S. Kim. Self-rectifying diffusion sampling with perturbed-attention guidance. In *European Conference on Computer Vision*, 2024.

D. Ahn, J. Kang, S. Lee, M. Kim, W. Jang, J. Min, S. Lee, S. Paul, and S. Kim. Where and how to perturb: On the design of perturbation guidance in diffusion and flow models. In *Advances in Neural Information Processing Systems*, 2025.

D. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. Concorde TSP solver, 2006. URL `https://www.math.uwaterloo.ca/tsp/concorde.html`. Software package.

A. Bansal, H.-M. Chu, A. Schwarzschild, R. Sengupta, M. Goldblum, J. Geiping, and T. Goldstein. Universal guidance for diffusion models. In *International Conference on Learning Representations*, 2024. Earlier versions appeared in CVPR Workshops 2023 and arXiv:2302.07121.

X. Bresson and T. Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

L. Chen, Y. Li, and J. Yan. MaskCO: Masked generation drives effective representation learning and exploiting for combinatorial optimization. In *International Conference on Learning Representations*, 2026.

P. Dhariwal and A. Q. Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

S. Han, Y. Zhang, and H. Wang. Guiding diffusion models with semantically degraded conditions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Y. He, K. Rojas, and M. Tao. What exactly does guidance do in masked discrete diffusion models. In *International Conference on Learning Representations*, 2026.

J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. A short version appeared in the NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications.

S. Hong. Smoothed energy guidance: Guiding diffusion models with reduced energy curvature of attention. In *Advances in Neural Information Processing Systems*, 2024.

S. Hong, G. Lee, W. Jang, and S. Kim. Improving sample quality of diffusion models using self-attention guidance. In *International Conference on Computer Vision*, 2023.

H. Lei, K. Zhou, Y. Li, Z. Chen, and F. Farnia. Boosting cross-problem generalization in diffusion-based neural combinatorial solver via inference time adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(43):36289–36297, 2026. doi: 10.1609/aaai.v40i43.40948.

T. Li, W. Luo, Z. Chen, L. Ma, and G.-J. Qi. Self-guidance: Boosting flow and diffusion generation on their own. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(1):781–791, 2026. doi: 10.1109/TPAMI.2025.3611831.

Y. Li, J. Guo, R. Wang, and J. Yan. T2T: From distribution learning in training to gradient search in testing for combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2023.

Y. Li, J. Guo, R. Wang, H. Zha, and J. Yan. Fast T2T: Optimization consistency speeds up diffusion-based training-to-testing solving for combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2024.

Y. Li, L. Chen, H. Wang, R. Wang, and J. Yan. Generation as search operator for test-time scaling of diffusion-based combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2025.

J. Ma, W. Pan, Y. Li, and J. Yan. COExpander: Adaptive solution expansion for combinatorial optimization. In *International Conference on Machine Learning*, 2025.

R. Qiu, Z. Sun, and Y. Yang. DIMES: A differentiable meta solver for combinatorial optimization problems. In *Advances in Neural Information Processing Systems*, 2022.

G. Reinelt. TSPLIB—a traveling salesman problem library. *ORSA Journal on Computing*, 3(4): 376–384, 1991.

Y. Schiff, S. S. Sahoo, H. Phung, G. Wang, S. Boshar, H. Dalla-Torre, B. P. de Almeida, A. Rush, T. Pierrot, and V. Kuleshov. Simple guidance mechanisms for discrete diffusion models. In *International Conference on Learning Representations*, 2025.

Z. Sun and Y. Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2023.

Y. Wang, Y. Li, J. Yan, and Y. Chang. StruDiCO: Structured denoising diffusion with gradient-free inference-stage boosting for memory and time efficient combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2025.

Y. Wang, Y. Li, J. Ma, J. Yan, and Y. Chang. Native adaptive solution expansion for diffusion-based combinatorial optimization. In *International Conference on Learning Representations*, 2026.

H. Ye, H. Lin, J. Han, M. Xu, S. Liu, Y. Liang, J. Ma, J. Zou, and S. Ermon. TFG: Unified training-free guidance for diffusion models. In *Advances in Neural Information Processing Systems*, 2024. Spotlight.

**Appendix overview.** The appendix is organized as follows. Appendix A provides implementation details for reproducibility (Section 4) and applies PISG to the Transformer-based GenSCO solver (Appendix A.5). Appendix B documents full results with standard deviations and statistical tests, the deployment pipeline with objective rewrite and 2-opt, and compute cost with compute-matched comparisons (Sections 6.1–6.3). Appendix C reports configuration selection, sensitivity, within-family transfer, and transfer under distribution shift (Section 6.1). Appendix D presents additional mechanism evidence and the limits of the diagnostics (Section 5).

**Seed convention.** Unless otherwise noted, appendix results report 5-seed means (seeds 41–45). Signal characterization in the main text (Section 5) also uses 5-seed aggregates. Single-seed analyses in ablation, trajectory probes, and cross-solver replication experiments are marked at first occurrence.

# A Method Details and Transfer to a Transformer Solver

This appendix provides implementation details for reproducibility, supporting Section 4. It includes the PISG inference pseudocode, the full set of perturbation operator definitions, and an empirical comparison of perturbation strategies that validates the design rationale. Appendix A.5 applies the same procedure to GenSCO, a Transformer solver without edge gating, and reports its configuration, results, and signal diagnostics.

## A.1 PISG Inference Pseudocode

Algorithm 1 summarizes the complete PISG inference procedure described in Section 4.2.

**Algorithm 1** PISG Inference

**Require:** trained denoiser $F_\theta$; instance graph $G$; perturbation operator $\mathcal{P}$; guidance strength $s$; step schedule $w(\cdot)$; layer set $\mathcal{S}_{\text{pisg}}$; sampling steps $T_s$

**Ensure:** final guided logits $\mathbf{p}_1^{\text{guided}}$, decoded to feasible $\hat{\mathbf{x}}$ by greedy/neighbor-aware decoder

$\mathbf{x}_{T_s} \sim$ initial noise distribution
**for** $t = T_s, T_s{-}1, \ldots, 1$ **do**
  $\mathbf{p}_t^{\text{normal}} \leftarrow F_\theta(\mathbf{x}_t, t, G;\, \Gamma_t)$
  $\mathbf{p}_t^{\text{perturbed}} \leftarrow F_\theta(\mathbf{x}_t, t, G;\, \mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t))$
  $s_t \leftarrow s \cdot w(t)$
  $\mathbf{p}_t^{\text{guided}} \leftarrow \mathbf{p}_t^{\text{normal}} + s_t(\mathbf{p}_t^{\text{normal}} - \mathbf{p}_t^{\text{perturbed}})$
  $\mathbf{x}_{t-1} \leftarrow \text{Sample}(\mathbf{p}_t^{\text{guided}}, t)$
**end for**
**return** $\hat{\mathbf{x}} = \text{Decoder}(\mathbf{p}_1^{\text{guided}})$

## A.2 Perturbation Operator Definitions

Section 4.2 introduces two representative operators (`uniform` and `adaptive`). Here we define the complete set, including one additional strategy used in the ablation study (Table 3).

Table 3: Perturbation strategy comparison on TSP-100 and MIS-ER (FastT2T, $T_s$=5, $T_g$=0, greedy, seed 42). Strategies are grouped into gate-targeting (operating on the gate vector $\mathbf{g}_{ij}^{(\ell)}$, Eq. (7)) and non-gate (operating on the denoiser input, Eq. (8)).

| | | | Gate-targeting | | | Non-gate |
|---|---|---|---|---|---|---|
| Task | Metric | Base | Unif. | Adpt. | Rand-g. | Inp-noise |
| TSP | gap↓ | 0.086 | 0.057 | 0.056 | **0.046** | 0.077 |
| MIS | cost↑ | 37.81 | **39.30** | 33.48 | <u>38.58</u> | 37.81 |

All three gate-targeting operators replace the $d_h$-dimensional gate vector $\mathbf{g}_{ij}^{(\ell)}$ at each target layer $\ell \in \mathcal{S}_{\text{pisg}}$:

$$\mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma): \quad \mathbf{g}_{ij}^{(\ell)} \mapsto \begin{cases} c \cdot \mathbf{1} & (\texttt{uniform}), \\ \mu_t^{(\ell)} \cdot \mathbf{1} & (\texttt{adaptive}), \\ \mathbf{u}_{ij}^{(\ell)},\ \mathbf{u}_{ij}^{(\ell)} \sim \mathcal{U}(0,1)^{d_h} & (\texttt{random-gate}), \end{cases} \tag{7}$$

where $c = \frac{1}{2}$ by default and $\mu_t^{(\ell)} = \frac{1}{|\mathcal{E}|} \sum_{(i',j') \in \mathcal{E}} \text{mean}(\mathbf{g}_{i'j'}^{(\ell)})$ averages over all edges and hidden channels for the current instance. `Uniform` is an instance-agnostic constant that completely severs the link between instance structure and gating weights. `Adaptive` preserves instance-level activation magnitude but removes spatial selectivity. `Random-gate` replaces each gate vector with i.i.d. uniform noise in the gate range $(0, 1)$; it destroys the edge–gate correspondence but does not preserve the learned gate values.

**Non-gate perturbation.** A fourth baseline, `input-noise`, does not target the gating mechanism. Instead, it adds Gaussian noise to the (scaled) noisy solution state $\mathbf{x}_t$ fed to the perturbed branch, while the gates are computed as usual without replacement:

$$\mathcal{P}_{\texttt{inp}}: \quad \mathbf{x}_t \ \mapsto\ \mathbf{x}_t + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0,\, \sigma^2 \mathbf{I}), \tag{8}$$

where $\sigma = 0.5$. Unlike the three gate-targeting operators in Eq. (7), $\mathcal{P}_{\texttt{inp}}$ degrades the input to the denoiser without replacing the structure-sensitive gates. It is included in Table 3 as a control that tests whether effective guidance requires targeting the gate mechanism rather than degrading information generically.

### A.3 Diagnostic Interpretation of the Guidance Signal

View the selected gate vectors as a concatenated continuous variable and define $\Delta\Gamma_t = \Gamma_t - \mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t)$. A local linearization of $F_\theta$ with respect to this gate variable gives the diagnostic approximation

$$\boldsymbol{\delta}_t = F_\theta(\mathbf{x}_t, t, G; \Gamma_t) - F_\theta(\mathbf{x}_t, t, G; \mathcal{P}_{\mathcal{S}_{\text{pisg}}}(\Gamma_t)) \approx J_{\theta,t}^{\Gamma}\, \Delta\Gamma_t,$$

where $J_{\theta,t}^{\Gamma}$ is the denoiser's Jacobian with respect to the selected gate map. This approximation is used only as a diagnostic lens, since $\mathcal{P}_{\mathcal{S}_{\text{pisg}}}$ is a finite intervention rather than an infinitesimal perturbation. Nevertheless, it clarifies that $\boldsymbol{\delta}_t$ probes the denoiser's local sensitivity to gate-mediated structural selection ($J_{\theta,t}^{\Gamma}$), projected along the direction removed by the perturbation ($\Delta\Gamma_t$). If this sensitivity is constraint-aligned—i.e., if the denoiser has learned to route messages preferentially along constraint-relevant edges—then $\boldsymbol{\delta}_t$ should concentrate at task-specific constraint boundaries rather than uniformly rescaling logits. Section 5 tests this prediction empirically.

### A.4 Why Effective Degradation Must Sever Structure-Sensitive Gating

Table 3 groups the four strategies into *gate-targeting* (operating directly on the gate vector) and *non-gate* (operating on the denoiser input). In this two-task ablation, gate-targeting emerges as the relevant search space for effective perturbation: the two best cross-task strategies (`uniform`, `random-gate`) both belong to this family. However, not all gate-targeting strategies are reliable: `adaptive` severely degrades MIS-ER (33.48 vs. baseline 37.81), likely because preserving instance-level activation magnitude produces an impure contrast that amplifies harmful directions. The non-gate baseline (`input-noise`) yields no improvement on MIS-ER. These results support the view that effective

Table 4: GenSCO TSP results with one frozen configuration (seeds 41–45). Entries are the mean paired gap reduction in percentage points ($> 0$: PISG better) with paired wins out of 5. $C$: search cycles; 2-opt: GenSCO's in-loop 2-opt steps.

| Task | 2-opt | $C$=10 | $C$=20 | $C$=40 |
|---|---|---|---|---|
| TSP-500 | 0 | +0.00096 (3/5) | +0.00106 (5/5) | +0.00176 (5/5) |
| TSP-500 | 20 | −0.00003 (2/5) | +0.00049 (4/5) | −0.00012 (2/5) |
| TSP-1000 | 0 | +0.04718 (5/5) | +0.02012 (5/5) | +0.01427 (5/5) |
| TSP-1000 | 20 | +0.00399 (5/5) | +0.00297 (4/5) | +0.00215 (4/5) |

Table 5: GenSCO MIS results with one frozen configuration per distribution (seeds 41–45). Entries are the mean paired change in independent-set size ($> 0$: PISG better) with paired wins out of 5. The baseline is evaluated at both the default and the PISG flip rate, and each row compares against the stronger one (default for ER, PISG flip rate for RB).

| Dataset | $C$=500 | $C$=1000 | $C$=2000 |
|---|---|---|---|
| MIS-ER | +0.047 (4/5) | +0.080 (5/5) | −0.036 (2/5) |
| MIS-RB | −0.003 (2/5) | +0.004 (4/5) | −0.005 (2/5) |

degradation in this setting requires both targeting the gating mechanism *and* breaking edge–instance correspondence. Among the four strategies, `uniform` and `random-gate` satisfy both requirements and are the most reliable cross-task choices in this two-task ablation.

### A.5 Adapting PISG to GenSCO

This subsection applies PISG to GenSCO [Li et al., 2025], a Transformer-based diffusion solver without edge gating. The procedure is the same as for AGNN denoisers: identify where instance structure enters the prediction, construct a structure-degraded branch there, and extrapolate with Equation (2). We report it as preliminary evidence on one Transformer solver with five paired seeds.

**Perturbation target.** GenSCO operates on node embeddings, and instance structure enters through an adjacency bias added to the attention logits. This bias is therefore the perturbation target. For layers $\ell \in \mathcal{S}_{\text{pisg}}$, the perturbation operator scales down the adjacency bias and, for TSP, also replaces each node's attention weights with a less node-specific average. Like the gate surrogates in Equation (3), these operations remove part of the edge-wise structural selectivity, while the structure-degraded branch shares all inputs and parameters with the normal branch.

**Configuration.** For TSP, one configuration (bias and attention perturbation, $s$=0.5, $\mathcal{S}_{\text{pisg}}$=$\{3\}$, `linear` schedule) is fixed for all twelve cells: two scales, three search budgets $C \in \{10, 20, 40\}$ cycles, and two settings of GenSCO's in-loop 2-opt (0 or 20 steps). For MIS, one bias-only configuration per distribution is selected at the reference budget $C$=500 on a subset of the evaluation graphs and frozen for $C \in \{1000, 2000\}$: ER uses $s$=0.93 with layers 3–6, and RB uses $s$=0.25 with layers 7–12, both with a `linear` schedule. Both MIS configurations also raise GenSCO's search flip rate from the default $[0.25, 0.40]$ to $[0.39, 0.54]$ (ER) and $[0.35, 0.50]$ (RB). No cell-wise re-tuning is performed.

**Results.** Table 4 shows positive mean effects in 10 of the 12 TSP cells. TSP-1000 improves in all six cells with 4–5 paired wins each, including after in-loop 2-opt. TSP-500 improves without 2-opt; with 2-opt the effect depends on the budget and is negative at $C$=10 and $C$=40. On MIS (Table 5), MIS-ER improves at $C$=500 and $C$=1000 and reverses at $C$=2000. This requires PISG together with the higher flip rate: at the default flip rate, the ER configuration lowers the independent-set size at all three budgets, and the higher flip rate alone also lowers the unguided baseline. MIS-RB shows no measurable effect at either flip rate. With five seeds, the cell-wise results are descriptive; we do not claim per-cell statistical significance.

**Signal diagnostics.** We repeat the diagnostics of Section 5 on GenSCO (seeds 41–45, 128 instances, three search checkpoints). On TSP-100, sc@2 exceeds its within-instance shuffled null at search cycles 0, 4, and 9 by 0.009–0.010 ($p < 10^{-9}$ at each checkpoint). On MIS-ER, the CMA ratio ($\theta$=0.3) is 6.5×, 9.5×, and 9.7× at cycles 0, 249, and 499 ($p < 10^{-22}$). The degree-2 sign pattern

and conflict-edge asymmetry therefore also appear in this attention-bias architecture. Their temporal profiles differ from FastT2T, and the diagnostics carry the same boundaries as in Appendices D.7–D.8: they characterize the guidance direction rather than predict per-instance benefit.

**Scope.** These results extend the procedure to one solver outside the edge-gated family; broader architectural coverage is left to future work.

# B Full Experimental Results

This appendix provides the complete experimental evidence for Section 6. It documents the solver implementations, benchmark datasets, evaluation metrics, and seed protocol, and extends the main tables with standard deviations and paired statistical tests (Appendices B.2–B.3). These controlled-isolation results constitute the primary empirical evidence for PISG's effectiveness within the edge-gated AGNN family. Appendix B.4 evaluates PISG in the full deployment pipeline with objective rewrite and 2-opt, and Appendix B.5 reports its compute cost and compute-matched comparisons.

## B.1 Experimental Setup

**Solvers.** We evaluate PISG on four AGNN-based solvers, all sharing the same 12-layer anisotropic GNN backbone with edge gating [Bresson and Laurent, 2017, Sun and Yang, 2023] and publicly released checkpoints:

- **DIFUSCO** [Sun and Yang, 2023]: categorical discrete diffusion, $T_s$=100, greedy decoding. Baseline and PISG both feed the denoiser the training-time state encoding ($2x-1$ with small multiplicative jitter) on TSP and MIS, whereas the official inference code feeds raw $\{0, 1\}$ states; with raw MIS inputs, the default configuration (selected under the training encoding) changes MIS-ER cost by $-1.7\%$.
- **T2T** [Li et al., 2023]: discrete diffusion, $T_s$=50, greedy decoding; supports objective-guided refinement (disabled in $T_g$=0 rows).
- **FastT2T** [Li et al., 2024]: consistency-based, $T_s$=5, greedy decoding.
- **StruDiCO** [Wang et al., 2025]: structured diffusion with variable-absorption noising, $T_s$=5; its native decoders are retained in all rows including $T_g$=0. For MIS, neighbor-aware decoding is a single-pass, deterministic variant that reranks candidates by $p_i/(\epsilon + \sum_{j\in\mathcal{N}(i)} p_j)$, distinct from iterative rewrite. For TSP, a threshold-merge decoder first merges edges whose scores exceed a threshold and then stitches the remaining subpaths in a seeded random order. Threshold $m_1$ applies to intermediate denoising samples and the initial decoding, and $m_2$ to rewrite sampling and decoding (active only when $T_g$>0). Baseline and PISG share $m_1$=$m_2$=0.9 on TSP-50 and 0.8 on TSP-100; on TSP-500 and TSP-1000 the baseline uses 0.7 and 0.8, respectively, and PISG uses 0.6 for both, including the deployment pipeline (Appendix B.4).

All main-table results report mean over 5 independent seeds (41–45). No post-hoc local search (no 2-opt) is applied unless stated (Appendix B.4).

**Datasets and metrics.** TSP is evaluated at four scales (50/100/500/1000 nodes; 1,280 instances for 50/100, 128 for 500/1000) against Concorde [Applegate et al., 2006], reporting gap (%). MIS is evaluated on ER (700–800 nodes, 128 graphs) and RB (200–300 nodes, 500 graphs), reporting solved cost directly. All datasets follow prior work's standard splits: the TSP test sets follow DIFUSCO [Sun and Yang, 2023], MIS-ER uses the DIMES evaluation graphs [Qiu et al., 2022], and MIS-RB uses StruDiCO's evaluation graphs [Wang et al., 2025].

**PISG configuration.** For TSP, three of the four AGNN solvers (DIFUSCO, T2T, FastT2T) share a single default: `adaptive`, $s$=2.0, L7, `linear`. For MIS, per-distribution defaults are: ER uses `uniform`/$s$=2.0/L7/`constant`; RB uses `adaptive`/$s$=2.0/L7/`constant`. On ER, these configurations transfer across three of four solvers; on RB, only FastT2T and StruDiCO have checkpoints. StruDiCO is a systematic exception: held-out validation selected substantially lower guidance strength on all tasks. TSP uses `adaptive`/$s$=0.1/L7/`linear`; MIS-ER uses `uniform`/$s$=0.1/L9/`constant`; MIS-RB uses `adaptive`/$s$=0.2/L5/`constant`. The calibration protocol and rationale appear in Appendix C.3.

Table 6: TSP small-scale results with standard deviations (gap%, 5-seed mean±std). Extended version of Table 1 (TSP-50/100). **Bold**: best gap within each solver block and scale.

| | | TSP-50 | | | TSP-100 | | |
|---|---|---|---|---|---|---|---|
| Method | $T_s/T_g$ | Cost | Gap% | Time | Cost | Gap% | Time |
| Concorde | – | 5.688 | 0.000 | – | 7.756 | 0.000 | – |
| DIFUSCO | 100/0 | 5.708 | 0.367±.044 | 6.2m | 7.864 | 1.388±.067 | 6.7m |
| +PISG | 100/0 | 5.699 | **0.203**±.017 | 6.2m | 7.800 | **0.567**±.038 | 10.2m |
| T2T | 50/0 | 5.727 | 0.683±.071 | 3.5m | 7.882 | 1.624±.064 | 3.6m |
| +PISG | 50/0 | 5.719 | **0.543**±.034 | 3.5m | 7.815 | **0.758**±.039 | 5.0m |
| FastT2T | 5/0 | 5.689 | 0.023±.005 | 45s | 7.762 | 0.079±.007 | 52s |
| | 5/5 | 5.688 | **0.0069**±.0011 | 135s | 7.758 | 0.025±.002 | 154s |
| +PISG | 5/0 | 5.689 | 0.017±.001 | 48s | 7.760 | 0.051±.003 | 55s |
| | 5/5 | 5.688 | 0.0071±.0012 | 141s | 7.758 | **0.023**±.001 | 154s |
| StruDiCO | 5/0 | 5.689 | 0.027±.005 | 45s | 7.770 | 0.187±.019 | 46s |
| | 5/5 | 5.688 | 0.0125±.001 | 71s | 7.758 | **0.031**±.002 | 77s |
| +PISG | 5/0 | 5.689 | 0.025±.003 | 48s | 7.770 | 0.179±.015 | 53s |
| | 5/5 | 5.688 | **0.0123**±.001 | 70s | 7.758 | 0.031±.002 | 82s |

Table 7: TSP large-scale results with standard deviations (gap%, 5-seed mean±std). Extended version of Table 1 (TSP-500/1000). **Bold**: best gap within each solver block and scale.

| | | TSP-500 | | | TSP-1000 | | |
|---|---|---|---|---|---|---|---|
| Method | $T_s/T_g$ | Cost | Gap% | Time | Cost | Gap% | Time |
| Concorde | – | 16.55 | 0.00 | – | 23.12 | 0.00 | – |
| DIFUSCO | 100/0 | 18.52 | 11.91±.22 | 3.8m | 26.19 | 13.31±.29 | 9.6m |
| +PISG | 100/0 | 18.11 | **9.44**±.38 | 5.8m | 25.65 | **10.95**±.10 | 16.7m |
| T2T | 50/0 | 18.42 | 11.33±.42 | 76s | 26.13 | 13.05±.12 | 2.4m |
| +PISG | 50/0 | 18.18 | **9.88**±.45 | 96s | 26.09 | **12.87**±.42 | 3.6m |
| FastT2T | 5/0 | 17.51 | 5.84±.41 | 32s | 24.61 | 6.46±.20 | 48s |
| | 5/5 | 16.92 | 2.27±.13 | 68s | 23.99 | 3.78±.05 | 177s |
| +PISG | 5/0 | 17.40 | 5.15±.19 | 33s | 24.49 | 5.93±.21 | 56s |
| | 5/5 | 16.87 | **1.94**±.09 | 69s | 23.86 | **3.22**±.11 | 182s |
| StruDiCO | 5/0 | 17.34 | 4.81±.23 | 33s | 25.83 | 11.75±.20 | 46s |
| | 5/5 | 17.00 | 2.71±.17 | 42s | 25.11 | 8.60±.20 | 74s |
| +PISG | 5/0 | 17.33 | 4.74±.26 | 37s | 25.43 | 9.99±.15 | 58s |
| | 5/5 | 16.95 | **2.45**±.19 | 46s | 24.67 | **6.72**±.34 | 86s |

**Settings by subsection.** Appendices B.2 and B.3 use the controlled-isolation setting ($T_g$=0, greedy decoding, no local search); the follow-up significance test in Appendix B.3 extends six cells to 20 paired seeds (41–60). Appendix B.4 uses the deployment pipeline ($T_g$>0 and 2-opt) with the main-table configurations frozen. Appendix B.5 measures runtime and memory under controlled isolation and compares PISG with baselines that use more denoising steps.

### B.2 Controlled-Isolation Results with Standard Deviations

Tables 6–8 extend the main-text tables (Tables 1–2) with per-seed standard deviations from 5 independent seeds (41–45). DIFUSCO and T2T report $T_g$=0 only. In several cases PISG reduces cross-seed variance alongside improving the mean; for instance, DIFUSCO TSP-100 std drops from .067 to .038, and DIFUSCO MIS-ER std from .19 to .09.

### B.3 Statistical Significance Analysis

Table 9 reports paired statistical tests for each solver–task combination. For TSP (gap%, ↓), we test $H_1$: PISG reduces gap compared to baseline; for MIS (cost, ↑), we test $H_1$: PISG increases cost. All tests use the exact one-sided Wilcoxon signed-rank test with $n$=5 paired seeds (seeds 41–45). With $n$=5, the minimum attainable $p$-value is 0.031; consequently, all $p$<0.05 results in the table achieve this minimum.

Of the 22 comparisons, 16 reach significance at $\alpha$=0.05 and 16 show 5/5 seed consistency. The six non-significant cases (T2T TSP-1000, StruDiCO TSP-50/100/500, StruDiCO MIS-ER, and StruDiCO

Table 8: MIS results with standard deviations (cost↑, 5-seed mean±std). Extended version of Table 2. **Bold**: best result within each solver block. *Significant at $\alpha$=0.05 (Table 9).

| | | ER | | | RB | | |
|---|---|---|---|---|---|---|---|
| Method | $T_s/T_g$ | Cost±std↑ | Gap%↓ | Time | Cost±std↑ | Gap%↓ | Time |
| Reference | – | 41.35 | 0.00 | — | 20.09 | 0.00 | — |
| DIFUSCO | 100/0 | 40.63±.19 | 1.73 | 4.0m | — | — | — |
| +PISG | 100/0 | 42.55±.09 | **−2.91*** | 6.5m | — | — | — |
| T2T | 50/0 | 36.50±.16 | 11.74 | 1.5m | — | — | — |
| +PISG | 50/0 | 38.25±.16 | **7.50*** | 2.3m | — | — | — |
| FastT2T | 5/0 | 37.65±.17 | 8.92 | 33s | 18.81±.02 | 6.38 | 38s |
| | 5/5 | 40.98±.12 | 0.87 | 62s | 19.47±.00 | 3.07 | 93s |
| +PISG | 5/0 | 38.89±.30 | 5.92* | 38s | 18.89±.05 | 5.98* | 41s |
| | 5/5 | 41.86±.07 | **−1.25** | 67s | **19.49**±.02 | **2.98** | 96s |
| StruDiCO | 5/0 | 39.89±.16 | 3.51 | 35s | 19.30±.03 | 3.93 | 48s |
| | 5/5 | 42.46±.06 | −2.72 | 49s | 19.74±.02 | 1.77 | 73s |
| +PISG | 5/0 | 40.05±.13 | 3.11 | 41s | 19.31±.03 | 3.90 | 50s |
| | 5/5 | 42.55±.11 | **−2.93** | 58s | 19.75±.01 | **1.72** | 79s |

MIS-RB) all show positive mean effects (PISG direction) over five seeds but with high variance relative to effect size. StruDiCO's lower guidance strength ($s$=0.1–0.2) produces smaller per-seed differences, requiring more seeds to resolve; the 20-seed follow-up below addresses these cells. Overall, PISG yields substantial, statistically significant gains on most settings, with marginal effects where the base solver is already near-optimal or the guidance strength is conservatively set. All reported $p$-values are uncorrected for multiple comparisons; conclusions rely on effect consistency across solvers, tasks, and scales rather than individual significance thresholds.

Table 9: Statistical significance of PISG improvements. **Win**: seeds (out of 5) where PISG improves. $p$: one-sided Wilcoxon signed-rank. $\Delta$: mean improvement ($> 0$ = PISG better; TSP: baseline gap% minus PISG gap%; MIS: PISG cost minus baseline cost). Bold $p$-values: significant at $\alpha$=0.05.

| Solver | Task | Win | $p$ | $\Delta$ (mean ± std) |
|---|---|---|---|---|
| DIFUSCO | TSP-50 | 5/5 | **0.031** | 0.1642 ± 0.0477 |
| DIFUSCO | TSP-100 | 5/5 | **0.031** | 0.8211 ± 0.0696 |
| DIFUSCO | TSP-500 | 5/5 | **0.031** | 2.4651 ± 0.2394 |
| DIFUSCO | TSP-1000 | 5/5 | **0.031** | 2.3601 ± 0.2495 |
| T2T | TSP-50 | 5/5 | **0.031** | 0.1397 ± 0.0886 |
| T2T | TSP-100 | 5/5 | **0.031** | 0.8668 ± 0.0880 |
| T2T | TSP-500 | 5/5 | **0.031** | 1.4450 ± 0.7704 |
| T2T | TSP-1000 | 4/5 | 0.219 | 0.1803 ± 0.4110 |
| FastT2T | TSP-50 | 5/5 | **0.031** | 0.0055 ± 0.0051 |
| FastT2T | TSP-100 | 5/5 | **0.031** | 0.0284 ± 0.0063 |
| FastT2T | TSP-500 | 5/5 | **0.031** | 0.6906 ± 0.3685 |
| FastT2T | TSP-1000 | 5/5 | **0.031** | 0.5264 ± 0.3562 |
| StruDiCO | TSP-50 | 3/5 | 0.219 | 0.0014 ± 0.0034 |
| StruDiCO | TSP-100 | 4/5 | 0.312 | 0.0078 ± 0.0195 |
| StruDiCO | TSP-500 | 3/5 | 0.406 | 0.0677 ± 0.3971 |
| StruDiCO | TSP-1000 | 5/5 | **0.031** | 1.7573 ± 0.3436 |
| *MIS* | | | | |
| DIFUSCO | MIS-ER | 5/5 | **0.031** | 1.9172 ± 0.2664 |
| T2T | MIS-ER | 5/5 | **0.031** | 1.7500 ± 0.2196 |
| FastT2T | MIS-ER | 5/5 | **0.031** | 1.2391 ± 0.2892 |
| StruDiCO | MIS-ER | 4/5 | 0.156 | 0.1594 ± 0.2649 |
| FastT2T | MIS-RB | 5/5 | **0.031** | 0.0776 ± 0.0390 |
| StruDiCO | MIS-RB | 2/5 | 0.406 | 0.0032 ± 0.0105 |

**Follow-up with 20 paired seeds.** Five seeds limit the attainable significance. We therefore extended the six non-significant cells to 20 paired seeds (41–60, including the original five) under the same controlled setting and the same one-sided Wilcoxon test (Table 10). This is a follow-up test of these six cells, not a re-test of all 22 comparisons. Four cells reach $p<0.05$: T2T TSP-1000 and StruDiCO TSP-100, TSP-500, and MIS-ER. All four remain significant after Holm correction across the six follow-up tests (largest adjusted $p$=0.036). StruDiCO TSP-50 remains inconclusive ($p$=0.062, 13/20

wins) with a positive mean, and StruDiCO MIS-RB is effectively null (9/20 wins, mean $\Delta \approx 0$). Taken together with Table 9, 15 of the 16 TSP solver–scale pairs and all four MIS-ER solvers reach $p<0.05$ in either the original or the follow-up test.

Table 10: Follow-up test of the six cells that were not significant with five seeds (controlled isolation, seeds 41–60, one-sided Wilcoxon signed-rank). $\Delta$ as in Table 9, computed over 20 seeds. Bold: $p<0.05$.

| Solver | Task | Win ($n$=20) | $p$ ($n$=20) | $p$ ($n$=5) | $\Delta$ (mean $\pm$ std) |
|---|---|---|---|---|---|
| T2T | TSP-1000 | 17/20 | **0.0032** | 0.219 | $0.2648 \pm 0.4110$ |
| StruDiCO | TSP-50 | 13/20 | 0.0615 | 0.219 | $0.0030 \pm 0.0072$ |
| StruDiCO | TSP-100 | 17/20 | **0.0008** | 0.312 | $0.0248 \pm 0.0310$ |
| StruDiCO | TSP-500 | 14/20 | **0.0120** | 0.406 | $0.1417 \pm 0.2539$ |
| StruDiCO | MIS-ER | 16/20 | **0.0047** | 0.156 | $0.1680 \pm 0.2265$ |
| StruDiCO | MIS-RB | 9/20 | 0.5298 | 0.406 | $-0.0005 \pm 0.0288$ |

### B.4 Deployment Pipeline: Objective Rewrite and 2-opt

The $T_g>0$ rows of Tables 1–2 combine PISG with objective-side rewrite for FastT2T and StruDiCO but still omit local search. To test whether the pre-decoding benefit survives a complete deployment pipeline, we evaluate the two strongest solvers at the two largest TSP scales with $T_s$=5, $T_g$=5, and 2-opt applied after both the initial decode and the rewrite. 2-opt runs to convergence with a cap of 1,000 iterations; in a seed-42 diagnostic, both branches converged after about 30 iterations without reaching the cap. The main-table PISG configurations are frozen without re-tuning, and we use five paired seeds (41–45).

Table 11: Full deployment pipeline ($T_s$=5, $T_g$=5, 2-opt to convergence; seeds 41–45). Gap% after the final rewrite and 2-opt. $\Delta$: baseline gap minus PISG gap ($>0$: PISG better).

| Solver | Task | Baseline | PISG | $\Delta$ | Win |
|---|---|---|---|---|---|
| FastT2T | TSP-500 | 0.429 | 0.403 | $+0.026$ | 4/5 |
| FastT2T | TSP-1000 | 0.627 | 0.599 | $+0.028$ | 4/5 |
| StruDiCO | TSP-500 | 0.323 | 0.314 | $+0.009$ | 4/5 |
| StruDiCO | TSP-1000 | 0.518 | 0.595 | $-0.076$ | 0/5 |

Table 11 shows that PISG retains a positive mean effect in three of the four cells. On FastT2T, it removes $6.1\%$ (TSP-500) and $4.5\%$ (TSP-1000) of the gap remaining after rewrite and 2-opt, with 4/5 paired wins in each. StruDiCO TSP-500 shows a smaller positive effect ($+0.009$ points, 4/5 wins). StruDiCO TSP-1000 reverses: PISG is worse than the baseline on all five seeds at the 2-opt endpoint. Gains are much smaller than under controlled isolation, which is consistent with downstream refinement absorbing part of the residual headroom.

Table 12: StruDiCO TSP-1000 after rewrite under increasing 2-opt budgets (seeds 41–45). The cap-0 row is the StruDiCO TSP-1000 $T_g$=5 row of Table 1; the cap-1,000 row is the endpoint in Table 11.

| 2-opt cap | Baseline | PISG | $\Delta$ | Win |
|---|---|---|---|---|
| 0 | 8.60 | 6.72 | $+1.88$ | – |
| 10 | 2.048 | 1.678 | $+0.370$ | 5/5 |
| 20 | 0.864 | 0.783 | $+0.081$ | 4/5 |
| 1,000 (converged) | 0.518 | 0.595 | $-0.076$ | 0/5 |

Table 12 traces the StruDiCO TSP-1000 reversal. The PISG advantage after rewrite persists through the first 20 2-opt iterations and reverses only at convergence. The frozen StruDiCO configuration therefore does not carry its pre-2-opt benefit to the converged endpoint at this scale.

### B.5 Compute Cost and Compute-Matched Comparison

**Runtime and memory.** PISG adds one perturbed forward pass per denoising step; the two branches are batched. Table 13 reports the overhead for both FastT2T and StruDiCO at $T_s$=5, $T_g$=0 across all six tasks. Wall-clock overhead is 3–17% on FastT2T and 4–26% on StruDiCO (5-seed mean). Memory overhead is negligible on TSP-50 ($<1\%$) and modest on TSP-100 ($\approx 8\%$), rising to 27–112%

on larger-scale tasks where the second branch's intermediate activations dominate. StruDiCO's higher time overhead on TSP-1000 (+26% vs. FastT2T's +17%) reflects limited batching headroom in its objective-aware pipeline. In all cases the absolute overhead is small: the largest wall-clock increase is 12 s (StruDiCO TSP-1000).

Table 13: PISG overhead (FastT2T and StruDiCO, $T_s$=5, $T_g$=0, batched, 4×A100-40GB). Batch size follows each solver's default; DIFUSCO/T2T overheads are proportional and can be read from Tables 1–2 time columns.

| | FastT2T | | | | StruDiCO | | | |
|---|---|---|---|---|---|---|---|---|
| | Time (s) | | Mem (GB) | | Time (s) | | Mem (GB) | |
| Task | Base | +PISG | Base | +PISG | Base | +PISG | Base | +PISG |
| TSP-50 | 45 | 48 | 0.71 | 0.71 | 45 | 48 | 0.71 | 0.71 |
| TSP-100 | 52 | 55 | 0.73 | 0.79 | 46 | 53 | 0.73 | 0.79 |
| TSP-500 | 32 | 33 | 0.92 | 1.32 | 33 | 37 | 0.93 | 1.32 |
| TSP-1000 | 48 | 56 | 1.57 | 3.10 | 46 | 58 | 1.57 | 3.10 |
| MIS-ER | 33 | 38 | 2.06 | 4.14 | 35 | 41 | 1.85 | 3.93 |
| MIS-RB | 38 | 41 | 0.88 | 1.48 | 48 | 50 | 1.21 | 1.54 |

**Compute-matched comparison.** Table 14 asks whether the time PISG spends on the extra branch would be better spent on more denoising steps. For each task we compare PISG@$T_s$=5 with (a) the *closest-time baseline*, the baseline $T_s$ whose wall-clock time most closely matches PISG@$T_s$=5, and (b) the baseline at $T_s$=10, which tests whether PISG can replace a doubled step budget. All six tasks are covered: TSP-100/500/1000 and MIS-ER come from the original dense $T_s$ sweeps, and TSP-50 and MIS-RB from a five-seed quality rerun that reproduces the historical sweep values, with times taken from that sweep.

Table 14: Compute-matched comparison (FastT2T, $T_g$=0, 5-seed mean). Closest-time: baseline $T_s$ whose wall-clock ≈ PISG@$T_s$=5. Runtimes come from the dense $T_s$ sweeps (independent measurements); absolute times can differ from Table 13, so comparisons are made within each sweep. TSP: gap% (↓); MIS: solved cost (↑).

| Task | PISG@5 | Time | Closest base ($T_s$) | Base@10 | Outcome |
|---|---|---|---|---|---|
| TSP-50 | 0.0171% | 52 s | $T_s$=5: 0.0225%, 52 s | 0.0152%, 71 s | Beats closest-time; base@10 better |
| TSP-100 | 0.051% | 54 s | $T_s$=6: 0.055%, 56 s | 0.042%, 72 s | Beats closest-time; base@10 better |
| TSP-500 | 5.15% | 35 s | $T_s$=8: 5.48%, 34 s | 5.74%, 38 s | Beats both |
| TSP-1000 | 5.93% | 58 s | $T_s$=8: 6.07%, 57 s | 6.15%, 63 s | Beats both |
| MIS-ER | 38.89 | 37 s | $T_s$=9: 39.73, 37 s | 39.79, 37 s | Both baselines better |
| MIS-RB | 18.89 | 49 s | $T_s$=6: 18.92, 53 s | 19.16, 68 s | Both baselines better |

On TSP, PISG@$T_s$=5 outperforms the closest-time baseline at all four scales. Against the baseline at $T_s$=10, it wins only on TSP-500 and TSP-1000, at comparable or lower wall-clock cost; on TSP-50 and TSP-100 the extra steps are better. On both MIS distributions, adding denoising steps is more effective than PISG at $T_s$=5 under either comparison. The compute advantage of PISG is therefore limited to large-scale TSP in this comparison. PISG can still be combined with $T_g$ scaling (Figure 4).

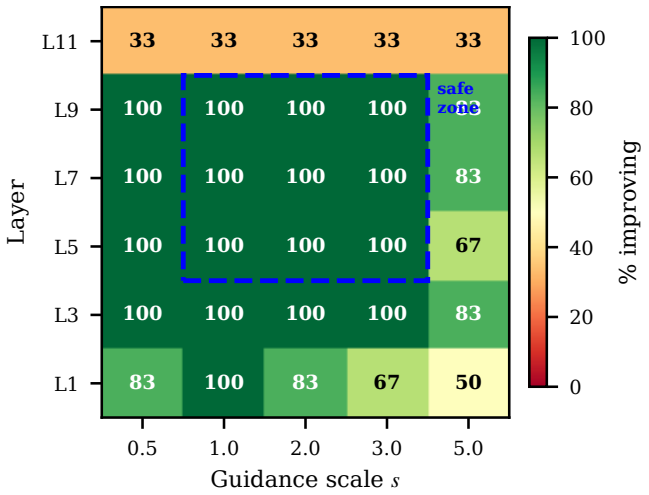


Figure 5: TSP-100 sensitivity heatmap (FastT2T, seed 42). The main text (Section 6.1) summarizes this sweep. 82% of 180 gate-level configurations improve over the baseline; 100% succeed in the safe zone (dashed box: scale $\in [1, 3]$, layers $\in [5, 9]$).

# C Configuration Selection and Transfer

This appendix documents the three-level configuration strategy summarized in Section 6.1: (1) a full sensitivity sweep on FastT2T that identifies family defaults and a broad safe zone; (2) direct transfer validation showing that these defaults work for DIFUSCO and T2T without re-tuning; (3) a systematic held-out calibration for StruDiCO, whose redesigned noise type requires solver-specific parameters. Appendix C.4 then tests how frozen configurations transfer under density, scale, and real-world distribution shift.

PISG has four configuration axes per task–distribution pair:

- **Perturbation operator** $\mathcal{P}$: one of the three strategies defined in Appendix A.2.
- **Guidance scale** $s$: global extrapolation strength. Family default $s$=2.0; StruDiCO uses lower solver-specific values (§C.3).
- **Layer set** $\mathcal{S}_{\text{pisg}}$: which backbone layers are perturbed; the shorthand "L$k$" perturbs a single layer at (0-indexed) position $k$. Default: L7 (layer index 7 in a 12-layer backbone).
- **Schedule** $w(\cdot)$: temporal weighting (`constant` or `linear`); default varies by task.

## C.1 FastT2T: Full Sweep and Family Defaults

The family defaults reported in Section 6.1 emerge from cross-task consistency patterns observed in a full sensitivity analysis on FastT2T. This analysis serves dual purposes: it maps the configuration landscape (Figure 5) and identifies robust default axes. A held-out validation search (below) selects the same default on MIS-ER and places the TSP default within the improving region on TSP-100.

**Setup.** We evaluate all six tasks (TSP-50/100/500/1000, MIS-ER, MIS-RB) with a factorial grid of 3 gate-level perturbation operators (uniform, adaptive, random-gate) $\times$ 5 scales ($s \in \{0.5, 1.0, 2.0, 3.0, 5.0\}$) $\times$ 6 layer positions (L $\in \{1, 3, 5, 7, 9, 11\}$) $\times$ 2 schedules (constant, linear), yielding 180 configurations per task, plus a 10-configuration input-noise control (Appendix A.2) for a total of 190 configurations. All configurations are evaluated on the standard test set with seed 42.

**Default selection.** Table 15 reports the per-task optimal configuration. Family defaults are chosen by favoring axes with the strongest cross-task consistency: Layer 7 is optimal on 4 of 6 subtasks; $s$=2.0 provides a safe cross-task default (the per-task best scale is task-sensitive, but $s$=2.0 always falls within the improving region); TSP tasks generally prefer `linear` schedules while MIS tasks prefer `constant`.

Table 15: Best FastT2T sweep configuration for each subtask. These are sweep optima, not necessarily the unified defaults used in the main tables (adap = adaptive, rand = random-gate, uni = uniform, const = constant).

| Subtask | Baseline | Best PISG | Rel. imp. | Best config |
|---|---|---|---|---|
| TSP-50 | 0.0196 gap% | 0.0143 | $-27.3\%$ | adap / s1.0 / L5 / linear |
| TSP-100 | 0.0860 gap% | 0.0414 | $-51.9\%$ | rand / s3.0 / L7 / linear |
| TSP-500 | 5.6087 gap% | 4.8040 | $-14.3\%$ | uni / s3.0 / L7 / const |
| TSP-1000 | 6.3131 gap% | 5.4532 | $-13.6\%$ | adap / s1.0 / L7 / linear |
| MIS-ER | 37.8125 | 39.2969 | $+3.9\%$ | uni / s2.0 / L7 / const |
| MIS-RB | 18.7740 | 19.1120 | $+1.8\%$ | adap / s2.0 / L9 / const |

**Per-task optima and safe zone.** Three sweep-level patterns are robust.

- **Layer 7 is the best default perturbation location.** It gives the global optimum on 4 of 6 subtasks. Layer 11 yields the fewest improving configurations on every task (none on MIS-ER and TSP-50/500/1000); on TSP-500/1000, layer 9 also never improves and degrades more severely.
- **TSP generally prefers linear schedules, but not without exception.** Three of four TSP subtasks favor `linear`; the exception is TSP-500, whose global optimum uses `constant`.
- **Scale is task-sensitive rather than monotonic.** The safe default is $s{=}2.0$ for both TSP and MIS, but the per-task best value is not a monotonic function of problem size.

**Robustness across the configuration landscape.** Of all 190 tested configurations (180 gate-level plus 10 input-noise), $83\%$ improve over the baseline on TSP-100, $68\%$ on MIS-RB, and $35\%$ on TSP-50 (where the baseline is already near-optimal). Within the central region (scale $\in [1, 3]$, layers $\in [5, 9]$), all 54 configurations improve on TSP-100 but only 24 of 54 on TSP-500, where layer 9 degrades in every case; the safe range is therefore task-dependent.

**Confirmatory held-out validation.** To verify that the family defaults above are not artifacts of test-set selection, we repeat the configuration search on a held-out validation split. TSP-100 uses a 256/1024 (val/test) partition of the standard 1280-instance set; MIS-ER uses 32/96 out of 128 graphs. A 50-config grid (5 scales $\times$ 5 layer positions $\times$ 2 schedules, with perturbation family fixed per task; TSP minimizes gap%, MIS maximizes solved cost) is swept on the val split with seed 42; the paper defaults are then evaluated on the test split with 5 seeds (41–45).

On MIS-ER, the validation optimum is the paper default. On TSP-100, the validation optimum is `adaptive`/$s{=}3.0$/L9/`constant`; the paper default ranks 15th of 50 and still improves over the baseline. The family default favors cross-task consistency over per-task optima (Default selection, above), and this optimum illustrates why: in the full sweep, layer 9 degrades every configuration on TSP-500 and TSP-1000. Applied unchanged to TSP-50, TSP-500, and TSP-1000, the TSP default consistently improves over the baseline without per-scale re-tuning (Table 16).

Table 16: Held-out validation (5-seed mean$\pm$std on the test split). On MIS-ER the paper default coincides with the validation optimum. TSP-50/500/1000 rows apply the TSP default without re-tuning.

| Task | Config | Gap% or Cost | Time (s) |
|---|---|---|---|
| TSP-100 | Baseline | $0.081 \pm .016$ gap% | 43 |
| TSP-100 | Paper default | $0.051 \pm .007$ gap% | 46 |
| MIS-ER | Baseline | $37.49 \pm 0.30$ cost | 28 |
| MIS-ER | Paper default (= val optimum) | $38.56 \pm 0.19$ cost | 32 |
| TSP-50 | Baseline | $0.023 \pm .005$ gap% | 44 |
| TSP-50 | Paper default (transfer) | $0.017 \pm .001$ gap% | 46 |
| TSP-500 | Baseline | $5.84 \pm .41$ gap% | 31 |
| TSP-500 | Paper default (transfer) | $5.15 \pm .19$ gap% | 33 |
| TSP-1000 | Baseline | $6.46 \pm .20$ gap% | 47 |
| TSP-1000 | Paper default (transfer) | $5.93 \pm .21$ gap% | 56 |

### C.2 DIFUSCO and T2T: Direct Transfer Validation

The family defaults derived from FastT2T transfer directly to DIFUSCO and T2T without re-tuning. Tables 17 and 18 verify this by applying the shared default and single-axis variants to each solver (seed 42).

Table 17: TSP-100 cross-solver transfer (gap%, ↓, seed 42). The shared default from FastT2T (adaptive/$s$=2.0/L7/linear) is applied to DIFUSCO and T2T. Each row varies one axis; bold indicates improvement over baseline. Scale variants: ↓ =1.0, ↑ =3.0.

| | DIFUSCO | T2T |
|---|---|---|
| Baseline | 1.393 | 1.559 |
| Default | **0.561** | **0.812** |
| Layer → 5 | 1.698 | 1.665 |
| Layer → 9 | **0.845** | **1.437** |
| Scale ↓ | **0.698** | **0.978** |
| Scale ↑ | **0.472** | **0.741** |
| → constant | **0.483** | **0.805** |
| → uniform | **0.846** | **1.084** |
| → random-gate | **0.728** | **0.962** |

Table 18: MIS-ER cross-solver transfer (cost, ↑, seed 42). The shared default from FastT2T (uniform/$s$=2.0/L7/constant) is applied to DIFUSCO and T2T. Each row varies one axis; bold indicates improvement over baseline. Scale variants: ↓ =1.0, ↑ =3.0.

| | DIFUSCO | T2T |
|---|---|---|
| Baseline | 40.38 | 36.76 |
| Default | **42.59** | **38.20** |
| Layer → 5 | **41.56** | **38.80** |
| Layer → 9 | 39.78 | **39.98** |
| Scale ↓ | **42.00** | **38.24** |
| Scale ↑ | **42.86** | **38.47** |
| → linear | **41.91** | **38.14** |
| → adaptive | **41.84** | **41.63** |
| → random-gate | **42.59** | **38.50** |

**Transfer patterns.** On TSP-100, the shared default improves both DIFUSCO and T2T; 14 of 16 solver–configuration cells (88%) improve over the respective baseline, the only exceptions being Layer 5. On MIS-ER, 15 of 16 cells improve, the exception being DIFUSCO with Layer 9. The shared default (Layer 7) improves every cell on both tasks, consistent with the single-solver landscape analysis.

On MIS-ER, `uniform` remains the family default rather than `adaptive` because `adaptive` severely degrades FastT2T on MIS-ER (Table 3), making it an unreliable cross-solver choice.

### C.3 StruDiCO: Solver-Specific Calibration

StruDiCO employs variable-absorption noising and a redesigned edge-gating architecture, making it qualitatively different from the other three solvers in the family. The shared family default ($s$=2.0) severely degrades StruDiCO on MIS-ER, and even moderate scales ($s \geq 1.0$) cause instability on TSP (Table 19). We therefore perform a systematic coarse-to-fine held-out calibration to identify solver-specific configurations.

**Stage 1: Coarse transfer check.** We evaluate the family default scale and three alternatives ($s \in \{0.5, 1.0, 2.0, 3.0\}$) on held-out validation splits for TSP-100, MIS-ER, and MIS-RB (3 seeds each). Table 19 shows that higher scales monotonically degrade all three tasks; $s$=0.5 is the best coarse candidate but still underperforms the baseline on MIS, motivating a finer search in the low-scale region.

**Stages 2–3: Low-scale refinement and single-axis sweep.** Stage 2 explores the low-scale region ($s \in \{0.1, 0.2, 0.3, 0.7\}$) and identifies per-task scale optima: $s$=0.1 for TSP-100 and MIS-ER, $s$=0.2

Table 19: StruDiCO coarse transfer check (Stage 1, 3-seed mean on validation splits). Family default scale ($s$=2.0) degrades all three tasks. Arrows indicate metric direction (↓ gap%, ↑ cost).

| Scale | TSP-100 gap% (↓) | MIS-ER cost (↑) | MIS-RB cost (↑) |
|---|---|---|---|
| Baseline | 0.172 | 39.85 | 19.30 |
| $s$=0.5 | **0.178** | 39.20 | 19.30 |
| $s$=1.0 | 0.248 | 38.73 | 19.14 |
| $s$=2.0 | 0.518 | 34.90 | 18.78 |
| $s$=3.0 | 2.448 | 33.35 | 18.68 |

for MIS-RB. Stage 3 fixes the selected scale and varies one axis at a time to check whether departing from the transferred defaults yields improvement. The axes tested are layer ({5, 7, 9}), perturbation operator, schedule ({constant, linear, reverse-linear}), fusion space ({logit, probability}; probability fusion was explored but not adopted in any final configuration), and active steps ({all, last}). For TSP-100, the transferred default (adaptive/L7/linear) remains optimal; for MIS-RB, calibration selected L5 over the transferred L7 (Table 20); for MIS-ER, Layer 9 improves over Layer 7 and is adopted. A small composed-candidate check (Stage 3b) verifies that the per-axis winners compose safely; for MIS-ER, the single-axis Layer 9 winner outperforms multi-axis combinations.

**Frozen configurations.** Table 20 summarizes the calibrated configurations used in all main-table StruDiCO results. These are frozen before test evaluation; the held-out test split is accessed only once with the final configuration.

Table 20: StruDiCO frozen configurations from held-out calibration. "Transferred default" shows the family default axis value; boldface marks axes where calibration selected a different value.

| Axis | TSP | MIS-ER | MIS-RB |
|---|---|---|---|
| Perturbation | adaptive | uniform | adaptive |
| Scale ($s$) | **0.1** | **0.1** | **0.2** |
| Layers | L7 | **L9** | **L5** |
| Schedule | linear | constant | constant |
| Fusion | logit | logit | logit |
| *Transferred default: adaptive (TSP) / uniform (MIS-ER) / adaptive (MIS-RB) / $s$=2.0 / L7 / task-default schedule / logit* | | | |

**Calibration cost.** The entire calibration pipeline evaluates 54 configurations × 3 seeds = 162 runs and completes in ∼80 minutes on 4 GPUs. This modest cost is comparable to one full 5-seed test evaluation and is performed once per solver.

### C.4 Frozen-Configuration Transfer under Distribution Shift

The preceding subsections select configurations for each task–distribution pair. This subsection evaluates how a selected configuration behaves when it is frozen and the test distribution moves away from the one it was selected on. We probe three axes separately: graph density, problem scale, and real-world TSP instances. All runs use controlled isolation ($T_g$=0, greedy decoding, no local search) and paired seeds 41–45.

**Density.** We evaluate the FastT2T MIS-ER checkpoint, trained at edge probability $p$=0.15, on ER graphs with $n$=750 and $p \in \{0.10, 0.125, 0.15, 0.175, 0.20\}$ (128 graphs per density), using the MIS-ER family default (`uniform`/$s$=2.0) for all densities. Table 21 reports relative gains with 95% intervals from a crossed paired graph-and-seed bootstrap. Transfer is positive from $p$=0.10 to $p$=0.175, with all intervals excluding zero and 5/5 wins, and negative at $p$=0.20. The boundary is asymmetric: the shift to sparser graphs ($p$=0.10) succeeds, while the shift of equal size to denser graphs ($p$=0.20) fails.

**Scale.** We freeze the TSP-100-selected setting ($s$=2.0, `linear` schedule) and evaluate FastT2T checkpoints trained at four scales on all four test scales (Table 22). When the test scale does not exceed the training scale, the frozen configuration improves 9 of 10 cells; the exception is the TSP-1000 checkpoint tested at TSP-500. Every cell is unanimous across seeds (5/5 or 0/5). All other negative cells test a checkpoint on instances larger than those it was trained on.

Table 21: Density transfer on MIS-ER (FastT2T trained at $p$=0.15, $n$=750, frozen `uniform`/$s$=2.0). Relative gain = $100 \times$ (PISG $-$ baseline)/baseline.

| $p$ | Baseline | PISG | Relative gain (95% CI) | Win |
|---|---|---|---|---|
| 0.100 | 50.85 | 52.17 | +2.67% [0.98, 4.38] | 5/5 |
| 0.125 | 44.84 | 46.50 | +3.93% [2.99, 4.88] | 5/5 |
| 0.150 | 37.59 | 38.64 | +3.08% [1.90, 4.33] | 5/5 |
| 0.175 | 31.47 | 32.03 | +2.13% [0.86, 3.38] | 5/5 |
| 0.200 | 25.35 | 24.69 | −2.28% [−3.49, −0.99] | 0/5 |

Table 22: Scale transfer on FastT2T with a frozen TSP-100-selected configuration. Entries are mean gap reductions in percentage points ($> 0$: PISG better) with paired wins. $^{\dagger}$: test scale larger than training scale.

| Train \ Test | 50 | 100 | 500 | 1000 |
|---|---|---|---|---|
| 50 | +0.01 (5/5) | +0.43 (5/5)$^{\dagger}$ | −7.18 (0/5)$^{\dagger}$ | −7.45 (0/5)$^{\dagger}$ |
| 100 | +0.01 (5/5) | +0.03 (5/5) | −6.76 (0/5)$^{\dagger}$ | −2.78 (0/5)$^{\dagger}$ |
| 500 | +0.89 (5/5) | +1.38 (5/5) | +0.69 (5/5) | +0.94 (5/5)$^{\dagger}$ |
| 1000 | +0.71 (5/5) | +0.62 (5/5) | −0.87 (0/5) | +0.53 (5/5) |

**TSPLIB.** TSPLIB [Reinelt, 1991] instances combine a non-uniform spatial distribution with a scale mismatch. We evaluate DIFUSCO, T2T, and FastT2T with their frozen TSP defaults on two bands: 50–200 nodes (29 instances, TSP-100 checkpoints) and 200–1,000 nodes (19 instances, TSP-500 checkpoints). Table 23 shows that in the 50–200 band, DIFUSCO and FastT2T improve with 4/5 wins each, while T2T degrades. In the 200–1,000 band, all three solvers are neutral or negative. Because this shift changes several factors at once, it does not isolate which factor causes the failures.

Table 23: TSPLIB transfer with frozen TSP defaults (controlled isolation, seeds 41–45). Gap% and mean gap reduction $\Delta$ in percentage points ($> 0$: PISG better).

| Band | Solver | Baseline | PISG | $\Delta$ | Win |
|---|---|---|---|---|---|
| 50–200 | DIFUSCO | 4.51 | 4.05 | +0.46 | 4/5 |
| 50–200 | T2T | 4.66 | 5.12 | −0.46 | 2/5 |
| 50–200 | FastT2T | 1.73 | 1.68 | +0.05 | 4/5 |
| 200–1,000 | DIFUSCO | 19.77 | 19.91 | −0.14 | 3/5 |
| 200–1,000 | T2T | 38.77 | 40.62 | −1.85 | 1/5 |
| 200–1,000 | FastT2T | 30.87 | 32.11 | −1.24 | 2/5 |

**Summary.** A frozen configuration transfers within the coverage of its training distribution: across moderate density changes, to test scales no larger than the training scale, and partly to small real-world instances. Failures concentrate at the high-density endpoint, at most extrapolations to larger scales, and at large TSPLIB instances. These results delimit the claim in Section 6.1 that good configuration regions transfer within a task–distribution pair.

# D Additional Mechanism Evidence

This appendix provides additional evidence for the signal characterization claims in Section 5: the signal concentrates on structurally contested decisions (not isotropic rescaling), corrections compose along the trajectory (not step-by-step improvement), and these signatures are consistent across the edge-gated AGNN family. Appendices D.7 and D.8 delimit what the diagnostics show: their association with final quality, and the absence of subtour-level signal in TSP.

## D.1 Raw-output Constraint-Boundary Alignment

**Purpose.** This experiment tests whether the trained denoiser's raw output—before any greedy decoding or post-processing—already exhibits constraint-aligned responses, and whether PISG amplifies them.

**Three branches.** We run three *independent*, complete denoising trajectories from the same trained FastT2T checkpoint and the same initial noise $\mathbf{x}_{T_s}$, differing only in the PISG configuration:

- **Normal**: standard inference without perturbation or guidance.
- **Perturbed**: structure-degraded gating is applied at every step, but *no* extrapolation is performed; the model's raw prediction under weakened gating is used directly.
- **Guided**: full PISG inference (Algorithm 1).

Each branch produces an independent set of raw edge/node probabilities, which are then passed to the standard greedy decoder for evaluation.

**Metrics.** For TSP, we measure (i) the optimality gap (%) to the Concorde reference, and (ii) the total number of edges the greedy decoder skips due to degree-2 violations during tour construction:

$$\text{TotalSkips} = \sum_{i=1}^{n} \big|\{e \in \text{candidates}(i) : e \text{ skipped due to degree-2 violation}\}\big|, \tag{9}$$

where candidates$(i)$ are the edges considered at step $i$ of greedy tour construction. For MIS, we measure (i) the independent-set size and (ii) the probability separation between selected and excluded nodes:

$$\text{ProbSep} = \frac{1}{|S|}\sum_{i\in S} p_i \;-\; \frac{1}{|V\setminus S|}\sum_{j\notin S} p_j, \tag{10}$$

where $S$ is the set of nodes selected by the greedy decoder and $p_i$ is the raw predicted probability for node $i$.

**Procedure.** We use a trained FastT2T checkpoint on 128 instances per TSP scale (TSP-50/100/500/1000) and 128 graphs for MIS-ER (seed 42). All three branches (normal, guided, perturbed) share the same initial noise per instance to ensure a fair comparison; the noise seed is reset per instance, so absolute gap values are not directly comparable to the main-table results (which use independent noise across instances and 5-seed aggregation). The relative ordering among branches remains valid. Tables 24 and 25 report the underlying data for Figure 1(b).

*Note:* Figure 1(a) uses a 10-node TSP instance constructed to illustrate the degree-2 constraint pattern; it is a schematic diagram, not the output of an actual solver.

## D.2 TSP: Degree-2 Sign Compliance Details

**Purpose.** This experiment tests whether the probability-space guidance signal $\boldsymbol{\delta} = \psi(\mathbf{p}^{\text{normal}}) - \psi(\mathbf{p}^{\text{perturbed}})$ (denoted $\tilde{\boldsymbol{\delta}}_t$ in Section 5; $\boldsymbol{\delta}$ in this subsection and Appendix D.3) exhibits a non-random sign structure aligned with the TSP degree-2 tour constraint, rather than acting as isotropic rescaling.

**Metric definition.** *Sign Compliance at degree 2* (sc@2) measures, for each node $i$, whether $\boldsymbol{\delta}$ simultaneously boosts the top-2 candidate edges and suppresses the third-ranked edge:

$$\text{sc@2}(i) = \mathbb{1}[\delta_{e_0(i)} > 0] \cdot \mathbb{1}[\delta_{e_1(i)} > 0] \cdot \mathbb{1}[\delta_{e_2(i)} < 0], \tag{11}$$

Table 24: TSP raw-output metrics underlying Figure 1(b) (128 instances per scale, single seed).

| Scale | Branch | gap (%) | total skips |
|---|---|---|---|
| TSP-50 | Perturbed | 1.586 | 32.4 |
| | Normal | 0.127 | 4.6 |
| | Guided | 0.040 | 0.08 |
| TSP-100 | Perturbed | 5.711 | 788.4 |
| | Normal | 0.339 | 43.3 |
| | Guided | 0.239 | 30.9 |
| TSP-500 | Perturbed | 15.308 | 68277 |
| | Normal | 6.103 | 47437 |
| | Guided | 5.434 | 34258 |
| TSP-1000 | Perturbed | 16.364 | 279783 |
| | Normal | 8.927 | 240013 |
| | Guided | 5.153 | 149813 |

Table 25: MIS-ER raw-output metrics underlying Figure 1(b) (128 graphs, single seed).

| Branch | IS size | prob separation |
|---|---|---|
| Perturbed | 34.52 | 0.448 |
| Normal | 37.54 | 0.787 |
| Guided | 39.49 | 0.886 |

where $e_k(i)$ denotes the edge ranked $k$-th by normal-branch probability among all edges incident to node $i$ (rank 0 = highest). The global sc@2 is the fraction of nodes satisfying this joint condition. As a null baseline, we compute a *shuffled* sc@2 by randomly permuting $\boldsymbol{\delta}$ across edges within each instance and averaging over 100 permutations.

**Procedure.** We compute sc@2 at each of the $T_s$=5 denoising steps on TSP-100 using FastT2T over five seeds (41–45), with 1280 instances per seed and 6400 instance-seed pairs in total. Statistical significance is assessed via 100-permutation tests, one-sided Wilcoxon signed-rank tests, and bootstrap 95% confidence intervals (1000 resamples).

**Results.** Table 26 shows that sc@2 is 76–78% at steps 0–2 and decays sharply afterward. Table 27 decomposes sc@2 into its three constituent conditions: at steps 0–2, all three conditions exceed 85%. At steps 3–4, the positive conditions for rank 0 and rank 1 drop sharply (reflecting prediction convergence), while the rank-2 negative condition remains stable at $\geq$85%, indicating that suppression of the third-ranked edge is the most persistent component of the degree-2 signal.

As a complementary view, Table 28 decomposes $\boldsymbol{\delta}$ by optimal-edge membership. At steps 0–2, the preference ratio is 0.86–0.88, the optimal-edge direction is positive for about 90% of instances, and $|\delta_{\text{opt}}|$ exceeds $|\delta_{\text{nonopt}}|$ in more than 99.9% of instances, consistent with the sc@2 sign pattern.

### D.3 MIS: Conflict-Edge Discrimination Details

**Purpose.** This experiment tests whether the probability-space signal $\boldsymbol{\delta}$ (Appendix D.2) encodes constraint-relevant magnitude differences on conflict edges in MIS. Unlike TSP, MIS $\boldsymbol{\delta}$ is predominantly positive ($> 99\%$ for high-probability nodes); the constraint signal is expected to appear as magnitude asymmetry rather than sign reversal.

**Metric definition.** *Conflict Magnitude Asymmetry* (CMA) quantifies how much $\boldsymbol{\delta}$ differentiates the two endpoints of an edge. Given a probability threshold $\theta$, an edge $(i, j) \in E$ is a *conflict edge* if both endpoints have high predicted selection probability: $\sigma(p_i^{\text{normal}}) > \theta$ and $\sigma(p_j^{\text{normal}}) > \theta$. CMA on a set of edges $\mathcal{C}$ is defined as:

$$\text{CMA}_{\mathcal{C}} = \frac{1}{|\mathcal{C}|} \sum_{(i,j)\in\mathcal{C}} |\delta_i - \delta_j|. \tag{12}$$

The CMA ratio compares conflict edges to non-conflict edges: Ratio $= \text{CMA}_{\text{conflict}}/\text{CMA}_{\text{non-conflict}}$. A shuffled baseline randomly permutes $\boldsymbol{\delta}$ across nodes within each instance (100 permutations).

Table 26: sc@2 summary across denoising steps (TSP-100, 5 seeds, 6400 instance-seed pairs). Ratio = sc@2 / shuffled baseline. All permutation $p < 0.01$ and Wilcoxon $p < 10^{-95}$.

| Step | $t$ | sc@2 | Shuffled | Ratio | sc@2 95% CI | Ratio 95% CI |
|---|---|---|---|---|---|---|
| 0 | 1000 | 0.764 | 0.041 | $18.6\times$ | [0.761, 0.767] | [18.3, 18.9] |
| 1 | 691 | 0.764 | 0.041 | $18.6\times$ | [0.761, 0.768] | [18.3, 18.9] |
| 2 | 413 | **0.782** | 0.036 | $\mathbf{22.0}\times$ | [0.779, 0.785] | [21.7, 22.4] |
| 3 | 191 | 0.425 | 0.020 | $21.1\times$ | [0.420, 0.429] | [20.6, 21.5] |
| 4 | 49 | 0.021 | 0.007 | $3.0\times$ | [0.021, 0.022] | [2.9, 3.2] |

Table 27: Three-condition decomposition of sc@2 (TSP-100). Each column shows the fraction of nodes satisfying that individual condition. sc@2 is the fraction satisfying all three conditions jointly.

| Step | $\delta_{\text{rank 0}}>0$ | $\delta_{\text{rank 1}}>0$ | $\delta_{\text{rank 2}}<0$ | $\delta_{\text{rest}}<0$ | sc@2 |
|---|---|---|---|---|---|
| 0 | 91.8% | 90.5% | 85.5% | 78.9% | 76.4% |
| 1 | 91.8% | 90.5% | 85.5% | 78.9% | 76.4% |
| 2 | 92.0% | 92.0% | 86.8% | 81.2% | 78.2% |
| 3 | 59.0% | 72.9% | 86.5% | 86.5% | 42.5% |
| 4 | 9.8% | 11.2% | 85.1% | 92.1% | 2.1% |

**Procedure.** We compute CMA at each denoising step on MIS-ER using FastT2T over five seeds (41–45), with 128 graphs per seed and 640 instance-seed pairs in total. We evaluate thresholds $\theta \in \{0.3, 0.5, 0.7\}$. Statistical significance is assessed via permutation tests, one-sided Wilcoxon signed-rank tests (per-instance $\text{CMA}_{\text{conflict}}$ vs. $\text{CMA}_{\text{non-conflict}}$), and bootstrap 95% CIs (1000 resamples).

**Results.** Table 29 reports CMA across steps and thresholds; Table 30 lists the corresponding hypothesis tests. At step 0, all three tests indicate a significant but mild effect ($p = 0.044$, Wilcoxon $p < 10^{-17}$, CI $[1.39, 1.59]$). From step 2 onward, the CMA ratio is strongly above 1, reaching $4.2\times$ at step 4 ($\theta$=0.3, CI $[4.11, 4.38]$). Higher thresholds ($\theta$=0.5, 0.7) yield similar ratios at steps 3–4 but with fewer conflict edges, confirming robustness.

The shuffled baseline (Table 29, last column) closely tracks non-conflict CMA, confirming that the elevated CMA on conflict edges reflects genuine constraint structure rather than random variation in $\boldsymbol{\delta}$ magnitudes.

### D.4 Directional rather than Temperature Scaling

**Purpose.** This experiment rules out the hypothesis that PISG's effect can be replicated by tuning the softmax temperature—a standard technique for adjusting distribution sharpness.

**Definition.** Temperature scaling adjusts the sharpness of a probability distribution by exponentiating and renormalizing:

$$p_e^{(T)} = \frac{p_e^{1/T}}{\sum_{e'} p_{e'}^{1/T}}, \tag{13}$$

where $T < 1$ sharpens the distribution, $T > 1$ smooths it, and $T = 1$ is the identity. This is equivalent to dividing pre-softmax logits by $T$.

**Procedure.** We sweep $T \in \{0.3, 0.5, 0.8, 1.0, 1.5, 2.0\}$ on FastT2T for TSP-100 and MIS-ER (5-seed mean, seeds 41–45) and compare against PISG under default configuration.

**Key argument.** Temperature scaling is a rank-preserving scalar transformation: it cannot introduce instance-specific, edge-dependent directional corrections. The constraint-aligned signal characterized in §D.2 (rank-dependent sign reversal captured by sc@2) and §D.3 (conflict-specific magnitude asymmetry captured by CMA) cannot arise from any scalar $T$. Rank-dependent sign flips require non-monotonic transformations, and conflict-specific magnitude inflation requires per-element factors; neither is achievable by a single temperature parameter.

Table 28: Direction alignment on matched-subset TSP-100 trajectories (5 seeds, 6400 instance-seed pairs). Preference ratio is $\delta_{\text{opt}}/(|\delta_{\text{opt}}| + |\delta_{\text{nonopt}}|)$ averaged across instances.

| Step | Pref. ratio | 95% CI | $\lvert\delta_{\text{opt}}\rvert > \lvert\delta_{\text{nonopt}}\rvert$ frac. | opt. positive frac. |
|---|---|---|---|---|
| 0 | 0.884 | [0.873, 0.893] | 99.97% | 89.8% |
| 1 | 0.883 | [0.874, 0.892] | 99.95% | 89.8% |
| 2 | 0.862 | [0.851, 0.873] | 99.92% | 90.2% |
| 3 | 0.501 | [0.482, 0.521] | 99.50% | 63.1% |
| 4 | 0.524 | [0.506, 0.542] | 94.12% | 8.8% |

Table 29: CMA across denoising steps and thresholds (MIS-ER, 5 seeds, 640 instance-seed pairs). CMA = mean $|\delta_i - \delta_j|$. Ratio = conflict / non-conflict. Rows for $\theta$=0.5 and 0.7 at steps 0–2 are omitted because conflict edges occur in at most 10 of the 640 instance-seed pairs.

| Step | $\theta$ | $n_{\text{conflict}}$ | $\text{CMA}_{\text{conflict}}$ | $\text{CMA}_{\text{non-conf}}$ | $\text{CMA}_{\text{shuffled}}$ | Ratio |
|---|---|---|---|---|---|---|
| 0 | 0.3 | 1823 | 0.056 | 0.037 | 0.037 | 1.5× |
| 1 | 0.3 | 1883 | 0.056 | 0.038 | 0.037 | 1.5× |
| 2 | 0.3 | 1934 | 0.086 | 0.029 | 0.029 | 2.9× |
| 3 | 0.3 | 50639 | 0.189 | 0.070 | 0.070 | 2.7× |
| 3 | 0.5 | 7382 | 0.200 | 0.072 | 0.072 | 2.8× |
| 3 | 0.7 | 548 | 0.249 | 0.076 | 0.076 | 3.3× |
| 4 | 0.3 | 39861 | **0.233** | 0.055 | 0.056 | **4.2×** |
| 4 | 0.5 | 9981 | **0.232** | 0.057 | 0.057 | 4.1× |
| 4 | 0.7 | 1449 | **0.241** | 0.060 | 0.060 | 4.0× |

**Results.** No temperature improves either task, and PISG surpasses every temperature setting (TSP-100 gap: 0.051% vs. best-$T$ 0.079%; MIS-ER cost: 38.89 vs. best-$T$ 37.65, i.e. gap 3.2% below PISG).

### D.5 Mechanism Consistency Across Solvers

**Purpose.** The signal characterization experiments in §D.2–D.3 use FastT2T ($T_s$=5, consistency model) as the primary analysis vehicle. To verify that the constraint-aligned signatures are not solver-specific, we repeat the full sc@2 and CMA analyses on StruDiCO ($T_s$=5, variable-absorption diffusion with neighbor-aware decoding), which uses substantially different PISG configurations: adaptive perturbation with $s$=0.1, $L$=7, linear schedule for TSP; uniform perturbation with $s$=0.1, $L$=9, constant schedule for MIS-ER. All results use seed 42.

**TSP: sc@2 on StruDiCO.** Table 31 shows that StruDiCO reproduces the same qualitative degree-2 sign pattern as FastT2T: sc@2 is highly significant at early steps (step 1: 0.568, 12.9× shuffle, permutation $p < 0.01$; step 2: 0.277, 6.2×, $p < 0.01$). The per-rank sign decomposition preserves the same structure: at step 1, rank 0 positive rate is 79.0%, rank 1 is 81.5%, and rank 2 negative rate is 73.4%, all well above chance. The signal decays more rapidly than in FastT2T: by step 3, sc@2 falls below the shuffled baseline. This faster decay is consistent with StruDiCO's lower guidance scale ($s$=0.1 vs. FastT2T's $s$=2.0) and its variable-absorption noise schedule.

**MIS: CMA on StruDiCO.** Table 32 shows that the CMA ratio grows monotonically from 2.9× at step 0 to 6.5× at step 4 ($\theta$=0.3, CI [5.79, 7.40]), exceeding FastT2T's peak of 4.2× at the same step. At higher thresholds, the pattern narrows: $\theta$=0.5 reaches 4.7× ($p < 0.015$) at step 4, but $\theta$=0.7 is not significant (ratio ≈ 1.0), whereas FastT2T maintains 4.0× at $\theta$=0.7. This indicates that StruDiCO concentrates its conflict-discrimination signal in the moderate-confidence boundary region rather than distributing it across all confidence levels.

**Cross-solver summary.** Both the degree-2 sign structure (TSP) and conflict-edge magnitude asymmetry (MIS) are confirmed on StruDiCO with full statistical tests, supporting that these constraint-aligned signatures arise from the task structure rather than from solver-specific artifacts. Appendix A.5 reports the same two signatures on GenSCO, indicating that they are not unique to edge gating. The quantitative differences (faster sc@2 decay on TSP, stronger but narrower CMA on MIS) are consistent with the substantially different guidance scales and perturbation strategies used across the two solvers.

Table 30: Hypothesis tests for CMA ratio (MIS-ER, 5-seed aggregate). Permutation: 100 shuffles of $\delta$ across nodes; Wilcoxon: one-sided paired test on per-instance $\text{CMA}_{\text{conflict}}$ vs. $\text{CMA}_{\text{non-conf}}$; Bootstrap: 1000 resamples of the ratio.

| Step | $\theta$ | Perm. $p$ | Wilcoxon $p$ | Ratio 95% CI |
|---|---|---|---|---|
| 0 | 0.3 | 0.044 | $4.9\times10^{-18}$ | [1.39, 1.59] |
| 1 | 0.3 | 0.043 | $1.1\times10^{-19}$ | [1.40, 1.60] |
| 2 | 0.3 | $<0.003$ | $6.8\times10^{-55}$ | [2.76, 3.13] |
| 3 | 0.3 | $<0.002$ | $8.9\times10^{-107}$ | [2.64, 2.80] |
| 3 | 0.5 | $<0.002$ | $4.8\times10^{-95}$ | [2.67, 2.87] |
| 3 | 0.7 | $<0.005$ | $1.4\times10^{-31}$ | [3.01, 3.55] |
| 4 | 0.3 | $<0.002$ | $5.8\times10^{-106}$ | **[4.11, 4.38]** |
| 4 | 0.5 | $<0.002$ | $1.5\times10^{-98}$ | [3.93, 4.25] |
| 4 | 0.7 | $<0.003$ | $1.0\times10^{-67}$ | [3.81, 4.24] |

Table 31: StruDiCO sc@2 across denoising steps (TSP-100, 1280 instances, seed 42). Step indices start from 1 due to solver-specific trajectory conventions. Bootstrap 95% CI over instances.

| Step | $t$ | sc@2 | Shuffled | Ratio | sc@2 95% CI |
|---|---|---|---|---|---|
| 1 | 691 | **0.568** | 0.044 | $\mathbf{12.9}\times$ | [0.559, 0.578] |
| 2 | 413 | 0.277 | 0.045 | $6.2\times$ | [0.269, 0.285] |
| 3 | 191 | 0.021 | 0.024 | $0.9\times$ | [0.019, 0.023] |
| 4 | 49 | 0.003 | 0.016 | $0.2\times$ | [0.002, 0.003] |

### D.6 Trajectory-Level Evidence

This subsection provides two trajectory-level probes that together support a schedule-sensitive trajectory interpretation in Section 5: stage-wise activation ablation compares early, late, and full-trajectory guidance, and stepwise greedy decoding shows guided intermediates can be individually worse while the final endpoint improves.

**Stage-wise activation ablation.** In the schedule-ablation protocol, *Early* activates PISG on steps $\{0, 1, 2\}$, *Late* activates steps $\{2, 3, 4\}$, *All steps* activates the full trajectory, and *Baseline* disables PISG.

Together with the signal characterization (Section 5) and the step-wise decode analysis below (Appendix D.6), these data support a trajectory-level interpretation: the benefit is schedule-sensitive and not explained by uniformly better intermediate greedy decodes.

**Stepwise decode analysis.** To test whether PISG uniformly improves the decoded solution at every intermediate step, we greedily decode $\mathbf{x}_0$ predictions from all three branches (guided, normal, perturbed) at each of the $T_s{=}5$ denoising steps, using matched $\mathbf{x}_t$ inputs. All results are from FastT2T with seed 42.

**TSP-100.** Table 34 shows that at step 0 guided $\equiv$ normal (the linear schedule sets scale$= 0$). At steps 1–3, guided intermediate solutions are *worse* than normal: gap increases by $+0.19$–$+0.26$ points, and degree violations are $1.4$–$19\times$ higher ($28\times$ at step 4). Only at the final step 4 does guided marginally improve ($-0.03$ gap), though win rate is only $3\%$. The consistently higher degree violations confirm that PISG's extrapolation produces intermediate predictions that are less structurally feasible when decoded greedily.

**MIS-ER.** Table 35 reveals a quality–feasibility trade-off. Guided predictions yield larger independent sets at every step, but with $1.5$–$12\times$ more conflicts and higher skip rates during greedy decoding ($3$–$4\times$ at steps 3–4; over $380\times$ at steps 0–2, where the normal branch almost never skips, $< 0.001$). The guided branch aggressively pushes probabilities above the selection threshold, inflating raw set size at the cost of feasibility.

**Implications.** Both tasks confirm that the final benefit of PISG does not arise from per-step decoded-solution improvement. On TSP, guided intermediate decodes are strictly worse; on MIS, the improvement in raw set size comes at the cost of much worse feasibility. The trajectory-level

Table 32: StruDiCO CMA ratio across denoising steps (MIS-ER, $\theta$=0.3, 128 instances, seed 42). Bootstrap 95% CI over instances.

| Step | $n_{\text{conflict}}$ | $\text{CMA}_{\text{conf}}$ | $\text{CMA}_{\text{non-conf}}$ | Ratio | Perm. $p$ | 95% CI |
|---|---|---|---|---|---|---|
| 0 | 11 | 0.061 | 0.021 | $2.9\times$ | 0.09 | [2.01, 4.05] |
| 1 | 8 | 0.078 | 0.024 | $3.3\times$ | 0.12 | [2.04, 4.72] |
| 2 | 28 | 0.103 | 0.029 | $3.6\times$ | $<0.036$ | [2.60, 4.63] |
| 3 | 70 | 0.161 | 0.031 | $5.2\times$ | $<0.014$ | [4.62, 5.74] |
| 4 | 105 | 0.140 | 0.022 | $\mathbf{6.5}\times$ | $<0.010$ | **[5.79, 7.40]** |

Table 33: Stage-wise activation ablation under the schedule-ablation protocol (FastT2T, seed 42). Lower is better for TSP gap; higher is better for MIS solved cost.

| Task | Baseline | Early | Late | All steps |
|---|---|---|---|---|
| TSP-100 gap↓ | 0.0860 | 0.0640 | 0.0643 | 0.0560 |
| MIS-ER cost↑ | 37.81 | 39.17 | 38.80 | 39.30 |

mechanism is the parsimonious explanation: PISG's per-step directional corrections reshape the denoising path in a way that benefits the final decode, even though intermediate decodes may appear worse in isolation.

**Relation to the signal analysis.** Section 5 and this subsection address different levels of the same question. Section 5 characterizes what the guidance direction encodes: $\boldsymbol{\delta}_t$ concentrates at constraint boundaries rather than rescaling predictions isotropically. The stepwise decode tests a stronger hypothesis, namely that applying this direction improves the feasibility of each intermediate decode, and finds that it does not. The guidance direction is therefore focused on constraint-relevant positions, while step-wise feasibility improvement is not the route by which PISG improves the endpoint. The stage-wise ablation (Table 33) is consistent with a trajectory-level effect: early-only and late-only guidance each improve over the baseline, and all-step guidance performs best. At the endpoint, the matched-noise probe (Table 24) shows fewer decoder skips with guidance, for example $43.3 \rightarrow 30.9$ on TSP-100.

### D.7 Association between Diagnostics and Final Quality

**Purpose.** sc@2 and CMA characterize the guidance direction. This subsection tests whether they also carry information about the final solution quality of PISG beyond instance difficulty. Easier instances tend to produce both stronger diagnostics and better solutions, so a raw correlation can reflect shared difficulty rather than an effect of the diagnosed signal.

**Procedure.** On FastT2T (seeds 41–45, aggregated per instance), we compute the Spearman correlation between each diagnostic and the final PISG quality, and the partial Spearman correlation that controls for the baseline's final quality on the same instance. TSP uses 1,280 TSP-100 instances and sc@2 at steps 0–2. MIS uses 127 MIS-ER instances and CMA at steps 3–4, the steps at which CMA is defined for nearly all instances. Intervals are 95% bootstrap intervals over instances.

**Results.** Table 36 shows that the relationship depends on the task. On MIS, CMA keeps an association with the final independent-set size after controlling for baseline quality (partial $\rho = 0.33$–$0.36$, $p < 0.001$): among instances where the baseline reaches similar set sizes, those with stronger conflict-edge asymmetry still tend to produce larger guided sets. CMA is therefore an outcome-linked diagnostic, but this association does not establish it as a causal mediator. On TSP, the raw correlation of sc@2 with final quality ($\rho \approx 0.45$) nearly vanishes once baseline quality is controlled (partial $\rho \approx 0.03$–$0.05$). PISG and baseline quality track each other closely across instances ($\rho = 0.93$), so the raw association reflects shared instance difficulty. sc@2 remains a structural diagnostic of the guidance direction but does not predict per-instance benefit.

Table 34: TSP-100 stepwise decode: solution quality (gap%, ↓) and structural feasibility (degree violation, ↓) at each step. Win%: percentage of instances where guided gap is lower than normal gap. Steps 1–3: guided is worse in both metrics; only at step 4 does guided narrowly improve.

| | Gap% (↓) | | | Degree violation (↓) | | |
|---|---|---|---|---|---|---|
| Step | Guided | Normal | Δ(G−N) | Guided | Normal | Win% |
| 0 | 2.023 | 2.023 | 0.000 | 0.0378 | 0.0378 | — |
| 1 | 2.170 | 1.982 | +0.188 | 0.0514 | 0.0378 | 24% |
| 2 | 1.994 | 1.731 | +0.263 | 0.0808 | 0.0307 | 22% |
| 3 | 0.696 | 0.448 | +0.249 | 0.1551 | 0.0083 | 9% |
| 4 | 0.176 | 0.206 | −0.030 | 0.0476 | 0.0017 | 3% |

Table 35: MIS-ER stepwise decode: independent set size (↑), conflict rate (↓), and greedy skip ratio (↓) at each step. Win%: percentage of instances where guided IS size exceeds normal IS size. Guided achieves larger sets but with substantially worse feasibility.

| | IS size (↑) | | | Conflict (↓) | | Skip ratio (↓) | | |
|---|---|---|---|---|---|---|---|---|
| Step | Guided | Normal | Δ | Guided | Normal | Guided | Normal | Win% |
| 0 | 24.03 | 1.78 | +22.25 | 0.0197 | 0.0048 | 0.286 | 0.001 | 98% |
| 1 | 23.99 | 1.88 | +22.11 | 0.0182 | 0.0047 | 0.262 | 0.001 | 98% |
| 2 | 19.95 | 4.71 | +15.24 | 0.0049 | 0.0032 | 0.088 | 0.000 | 92% |
| 3 | 36.54 | 31.99 | +4.55 | 0.0349 | 0.0028 | 0.691 | 0.156 | 67% |
| 4 | 38.35 | 37.80 | +0.55 | 0.0165 | 0.0017 | 0.514 | 0.167 | 40% |

### D.8 No Subtour-Level Signal in TSP

**Purpose.** The TSP diagnostic in Section 5.1 measures the local degree-2 boundary. This subsection tests whether $\boldsymbol{\delta}_t$ also carries subtour-level information, that is, whether guided predictions help the greedy decoder avoid closing cycles before all nodes are connected.

**Procedure.** During greedy decoding, a candidate edge is rejected as fragment-closing when it would close a cycle among a proper subset of the nodes. If $\boldsymbol{\delta}_t$ carried subtour information, decoding the guided prediction should produce fewer such rejections $R$ than decoding the normal prediction. We decode the normal, guided, and shuffled-$\boldsymbol{\delta}_t$ predictions at each step on FastT2T TSP-100 (seeds 41–45, 1,280 instances per seed, ten within-instance shuffles) and report $\Delta R_t = R_t^{\text{normal}} - R_t^{\text{guided}}$ ($> 0$: fewer rejections with guidance) with 95% hierarchical bootstrap intervals. Step 0 is omitted because the three predictions coincide there.

**Results.** Table 37 provides no evidence of subtour-level signal. Three of the four intervals include zero, and at step 3 guidance increases fragment-closing rejections. The TSP mechanism evidence in this paper therefore concerns local degree-2 boundary sensitivity, not subtour awareness.

Table 36: Association between diagnostics and final PISG quality (FastT2T, seeds 41–45). Partial correlations control for the baseline's final quality on the same instance.

| Diagnostic | Raw $\rho$ | Partial $\rho$ (controlling baseline) | 95% CI of partial $\rho$ |
|---|---|---|---|
| TSP sc@2, steps 0–2 ($n$=1,280) | 0.44–0.45 | 0.03–0.05 | includes 0 |
| MIS CMA, steps 3–4 ($n$=127) | 0.34–0.39 | 0.33–0.36 | excludes 0 |

Table 37: Fragment-closing rejections per instance during greedy decoding (FastT2T TSP-100, seeds 41–45). $\Delta R_t > 0$ would indicate that guidance reduces premature cycle closure.

| Step | Normal $R$ | Guided $R$ | Shuffled $R$ | $\Delta R_t$ (95% CI) |
|---|---|---|---|---|
| 1 | 0.5670 | 0.5345 | 0.5623 | $+0.033$ [$-0.007$, 0.075] |
| 2 | 0.4169 | 0.4406 | 0.4116 | $-0.024$ [$-0.060$, 0.015] |
| 3 | 0.0973 | 0.1259 | 0.1022 | $-0.029$ [$-0.046$, $-0.012$] |
| 4 | 0.0325 | 0.0311 | 0.0335 | $+0.001$ [$-0.007$, 0.011] |

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction claim that PISG improves solution quality within a controlled family of edge-gated AGNN solvers on TSP and MIS, and that its constraint-aligned signal is mechanistically distinct from objective-side refinement. These claims are supported by Tables 1–2 and the signal characterization in Section 5. The scope is explicitly limited to solvers sharing edge-gated message passing; the deployment-pipeline results (Section 6.2) and the transfer to the Transformer-based GenSCO (Appendix A.5) are reported as conditional and preliminary evidence, respectively.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 discusses limitations including: (1) the extra forward pass raises peak GPU memory by 27–112% on larger tasks, PISG offers limited value at small $T_s$ (it improves over the baseline from $T_s$=5 onward, from $T_s$=3 on MIS-ER), and additional denoising steps are more effective at matched compute on MIS; (2) it is training-free but not tuning-free, and frozen configurations can fail under larger distribution shifts or after converged 2-opt; (3) the main evidence is within the controlled family of edge-gated AGNN denoisers and two problem families (TSP, MIS), with only preliminary transfer to one Transformer solver; (4) the constraint-aligned interpretation is empirical and partly correlational; and (5) TSP mechanism evidence covers the degree-2 local constraint, with no subtour-level signal.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This is an empirical paper. PISG is motivated by analogy to PAG in vision diffusion models; no formal theorems are stated or proved.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 6.1 gives the family defaults (perturbation type, scale, layers, schedule) for each task, and Appendix B (Experimental Setup) lists the configuration used for every solver, including StruDiCO's calibrated settings. Appendix A gives inference pseudocode and perturbation definitions; Appendix B documents solvers, benchmarks, metrics, seed protocol, standard deviations, and statistical tests; Appendix C reports the full configuration-selection protocol including sweep grids, held-out validation splits, and transfer rules. Appendix A.5 specifies the GenSCO perturbation and its configurations. Main-table benchmarks follow standard datasets from prior work; the distribution-shift sets are described in Appendix C.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is available at `https://github.com/Ming-321/PISG`. It contains the PISG implementation integrated into DIFUSCO, T2T, FastT2T, and StruDiCO and a patch for GenSCO, together with environment files, configuration files, data acquisition instructions, and reproduction commands. The main-table datasets are synthetic benchmark test sets released by prior work; the repository documents where to obtain them and provides checksums for verification.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: PISG is a test-time method requiring no training. All inference hyperparameters and selection protocols are specified in Section 6.1, Appendix C.1, and Appendix C. Solver implementations, benchmark sizes, evaluation metrics, and seed protocol are summarized in Appendix B; data splits use the standard benchmarks from prior work.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All main results are averaged over 5 independent random seeds (seeds 41–45); variability is reported as mean $\pm$ standard deviation across seeds. Tables with full standard deviations appear in Appendix B, and paired exact one-sided Wilcoxon signed-rank tests are reported in Appendix B.3, together with a 20-seed follow-up test of the six cells that were not significant with five seeds. Mechanism analyses in Appendix D additionally report bootstrap confidence intervals and permutation or Wilcoxon tests where applicable.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 6.3 reports wall-clock time and overhead analysis. Appendix B.5 provides the full efficiency table including per-task runtime and memory. All experiments were conducted on NVIDIA A100 GPUs (40 GB). We estimate the total compute for all reported main experiments and validation sweeps at approximately 50 A100 GPU-hours (inference only; no training is performed): main-table 5-seed evaluations account for roughly 25 hours, and validation sweeps plus StruDiCO calibration account for the remaining 25 hours. The additional experiments for the 20-seed follow-up, the deployment pipeline with 2-opt, the GenSCO adaptation, the distribution-shift transfer, and the diagnostic analyses (Appendices A.5, B.3–B.5, C.4, D.7–D.8) required roughly 100 further A100 GPU-hours. Preliminary exploratory sweeps and failed configurations outside the reported validation grids required additional compute and are not included in this estimate.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: This work proposes a test-time inference technique for combinatorial optimization solvers. It does not involve human subjects, personal data, or dual-use concerns.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: PISG is a benchmark-level method for improving the quality–efficiency trade-off of diffusion-based CO solvers. Such techniques may benefit routing, logistics, scheduling, and resource allocation applications. It is evaluated only on standard TSP/MIS benchmarks and is not a deployed decision system. In real deployments, optimization objectives may omit fairness, robustness, safety, or domain constraints, so task-specific validation and human oversight are required before use in high-stakes settings.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: PISG is a lightweight inference-time module for combinatorial optimization. It does not produce generative content and poses no misuse risk requiring safeguards.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that

users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All base solvers (DIFUSCO, T2T, FastT2T, StruDiCO, and GenSCO), the Concorde reference solver, the sources of the test sets, and TSPLIB are cited (Appendices B and C.4). The four edge-gated solvers are released under the MIT license. GenSCO's public code release does not include a repository-level license, so our repository distributes only a patch that is applied to a user-fetched copy of its upstream source. The released code retains the licenses and copyright notices of all third-party components and documents their provenance.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the PISG implementation integrated into the native code of DIFUSCO, T2T, FastT2T, and StruDiCO, and a patch for GenSCO, at `https://github.com/Ming-321/PISG`. The repository documentation includes installation instructions, environment specifications, configuration files, reproduction scripts, license and provenance information, and the scope of the provided reproduction. We release code only; no new dataset or pretrained model checkpoint is introduced.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This work does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This work does not involve human subjects research.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as a component of the proposed method or experimental pipeline.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.