TRAINING PROBABILISTIC GRAPH PREDICTORS WITHOUT GRAPH MATCHING VIA FLOW MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised graph prediction (SGP) maps any kind of input, such as images or molecular fingerprints, to graph outputs. The main challenge is that graphs do not have an intrinsic node ordering, making many different tensor representations valid and thus complicating coordinate-wise supervision. Existing SGP methods often handle this issue by matching predicted and target graphs. However, matching typically introduces a cubic-time alignment step and is hard to extend to settings with multiple valid, non-isomorphic outputs. We instead cast SGP as conditional generation over graph isomorphism classes and introduce PSGP (*probabilistic supervised graph predictor*), which instantiates this formulation using permutation-equivariant conditional flow matching. PSGP supports graph outputs with discrete and continuous graph features, and its training using standard coordinate-wise losses without graph matching scales quadratically. We prove that its empirical risk is invariant to the arbitrary indexing of each target graph. On six established SGP benchmarks, a single sample from PSGP outperforms the strongest baseline on graph-isomorphism accuracy, and modal decoding from multiple samples improves these results further. We also introduce a probabilistic graph prediction benchmark from genotype data with ambiguous targets, where PSGP achieves lower discrepancy from the target distribution over graph isomorphism classes. These results support conditional flow matching as a scalable, probabilistic approach to SGP.

1 INTRODUCTION

Many prediction problems have graph outputs. A satellite image might determine a road network, a molecular fingerprint might specify a molecular structure, and genetic sequences might be consistent with multiple family graphs. In *supervised graph prediction* (SGP), the input can be of any modality and the goal is to predict an entire attributed graph: its nodes, edges, and their associated features (Krzakala et al., 2024). This differs from many graph learning settings, where an input graph, or at least the set of nodes over which predictions are made, is already given (Xia et al., 2021).

A central difficulty is that graphs have no intrinsic node ordering, whereas their tensor representations require assigning nodes to specific indices. The same graph can admit many equivalent node and adjacency orderings. A point predictor trained with coordinate-wise losses against arbitrarily ordered targets can therefore receive incompatible supervision for the same underlying graph. Prior work avoids this problem in several ways. When a domain-specific convention is available, graphs can be assigned an ordering, for example through canonical atom sorting in molecular generation (Bresson & Laurent, 2019) or spatially defined traversals for road graphs (Belli & Kipf, 2019). Other approaches formulate graph prediction indirectly using Fused Gromov–Wasserstein (FGW) geometry (Peyré & Cuturi, 2019; Mémoli, 2011; Vayer et al., 2020), for example, through FGW barycenter regression (Brogat-Motte et al., 2022). These methods can be effective, but rely on a suitable representation, ordering convention, or intermediate predictions, which limits their general applicability.

An alternative is to account explicitly for permutations when comparing a prediction with its target. Relationformer (Shit et al., 2022) uses Hungarian matching (Kuhn, 1955; Carion et al., 2020) to assign predicted objects to targets before evaluating their relations. Similarly, Grapher (Melnyk et al., 2022) first aligns generated and target nodes and then constructs edges. In both cases, graph structure does not determine the correspondence itself. Any2Graph (Krzakala et al., 2024) addresses this limitation with a partially masked Fused Gromov–Wasserstein (PMFGW) objective that jointly aligns

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

and compares node and edge information through optimal transport. GRALE (Krzakala et al., 2025) builds on this structure-aware approach by training a graph autoencoder using a reconstruction loss also based on optimal transport and a learned differentiable matcher, and then predicting through the resulting graph-level latent space. These methods remove dependence on the dataset's arbitrary node ordering, but keep an explicit or learned notion of correspondence between graph representations. Matching also carries substantial computational cost: Hungarian assignment is cubic in the number of matched elements, Any2Graph's dense optimal transport solver is cubic in graph size per iteration, and GRALE reports cubic overall complexity as well.

A related line of work studies graph generative models, which learn distributions over graph structures (Zhu et al., 2022). Most graph generative models learn an unconditional distribution over graphs, although conditional generation has also been studied. For example, DeFoG (Qin et al., 2025a) uses permutation-equivariant graph flows and supports both unconditional generation and generation conditioned on graph-level labels through classifier-free guidance (Ho & Salimans, 2021).

We formulate SGP as conditional graph generation from paired input–graph examples (x, g), with the supervised input x as the condition. Here x can be of any modality, such as an image or a molecular fingerprint, and the objective is to predict its associated graph output. We model the conditional distribution over graph isomorphism classes, that is, equivalence classes of attributed graphs under node reindexing. For deterministic tasks, the target distribution given x is concentrated on one class. When the input is ambiguous, it can assign probability to several non-isomorphic graphs. This formulation treats equivalent indexings as representations of the same output, while retaining uncertainty over genuinely different graph structures. We introduce PSGP (*probabilistic supervised graph predictor*, Figure 1) to learn this input-conditioned distribution directly from the dataset's arbitrarily indexed targets, without needing graph matching or a canonical node ordering.

We instantiate this formulation using conditional flow matching, building on flow models for discrete and mixed discrete–continuous data (Campbell et al., 2024) and their permutation-equivariant adaptation to graph generation (Qin et al., 2025a; Carballo-Castro et al., 2026). Starting from a permutation-invariant source distribution, PSGP uses a permutation-equivariant denoiser conditioned on x to generate graphs with categorical and continuous node and edge features. During training, the denoiser predicts the clean target from an intermediate graph state, with the supervised input x provided throughout. Training uses standard coordinate-wise cross-entropy and squared-error losses against the indexed graph representation supplied by the dataset. No node correspondence, transport plan, or graph matching is computed.

We show that the expected training objective of PSGP is unchanged by independently reindexing each target, provided that the conditional probability path and denoiser are permutation equivariant and the endpoint loss is invariant to simultaneous reindexing of its prediction and target. This result covers categorical and continuous graph features and holds for every model parameter setting, and therefore does not depend on successful optimization. Reindexing a target reindexes the distribution of intermediate graph states, and the equivariant denoiser follows the same permutation, leaving the expected coordinate-wise loss unchanged. Training can thus use the dataset's indexed targets directly, without a matching stage. For fixed model width, depth, and number of condition tokens, the dense graph computations and coordinate-wise losses require $O(n^2)$ time and memory per training example, where n is the number of output nodes. Scaling experiments confirm this difference in objective cost and show that prediction quality degrades slowly with output graph size.

Our main contributions are:

- We formulate SGP as conditional generation from paired input–graph examples, modeling distributions over graph isomorphism classes, and instantiate this formulation with PSGP.
- We prove that the expected training objective is invariant to independently reindexing each target, for every model parameter setting and for categorical and continuous graph features.
- On each of six established SGP benchmarks with deterministic targets, a single PSGP sample outperforms the strongest reported baseline under graph-isomorphism accuracy. We introduce a sample-based modal decoder for point prediction that further improves accuracy on every task.
- We introduce a probabilistic SGP benchmark from genotype data with non-identifiable target family graphs. PSGP achieves lower squared maximum mean discrepancy than Any2Graph, nearly halving it on the non-identifiable subset.

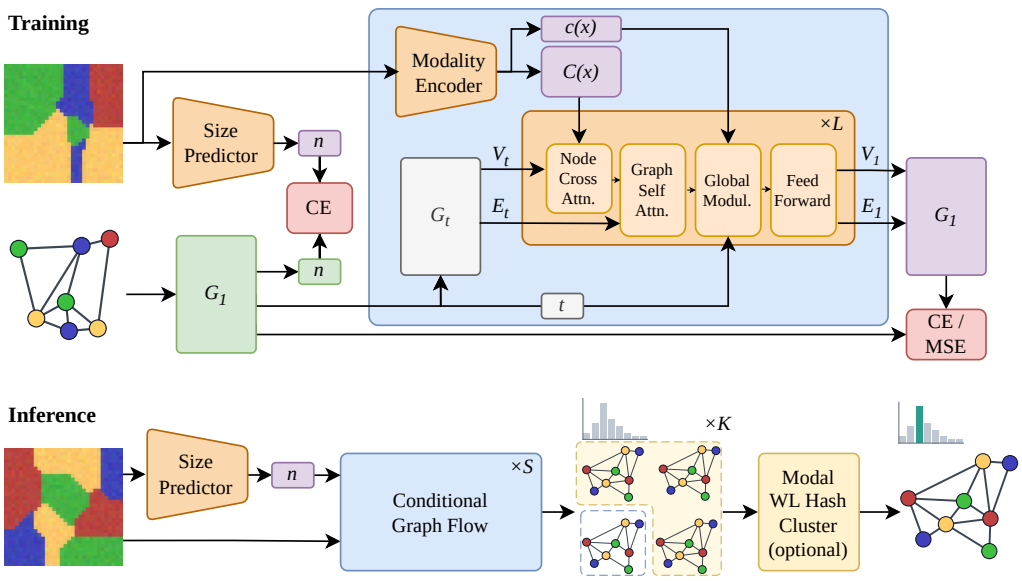


Figure 1: **Overview of PSGP.** *Training:* a task-specific modality encoder maps the input x to a global representation $c(x)$ and condition tokens $C(x)$, while a separate size predictor is trained to predict the target number of nodes n. Given the target graph G_1, we sample an intermediate state $G_t \sim q_t(\cdot \mid G_1)$ from the fixed flow path. A permutation-equivariant graph transformer conditions on $C(x)$ through node cross-attention, while $c(x)$ and time t initialize a global state that modulates node and edge updates, and predicts the clean node and edge features of G_1, which are trained directly against the arbitrarily indexed target with coordinate-wise cross-entropy or squared-error losses, without graph matching. *Inference:* the predicted size n determines the source graph size, and the learned conditional graph flow is integrated for S steps to generate graph samples. Multiple samples represent the predictive distribution. When a single prediction is required, K samples are grouped by their WL hashes and the largest cluster is returned as an approximation to the empirical mode.

2 METHOD

2.1 PROBABILISTIC FORMULATION OF SGP

We begin by formulating SGP probabilistically, including tasks conventionally treated as deterministic. Rather than viewing SGP as learning a point-valued map from an input to a single graph, we model a conditional distribution over graphs up to isomorphism. The usual deterministic setting is recovered when this distribution is concentrated on a single graph isomorphism class, while a non-degenerate distribution can represent non-uniqueness among graph outputs.

Indexed attributed graphs. Let $X \in \mathcal{X}$ denote the SGP input, with x a realization. The output is an attributed graph whose nodes are not assumed to correspond to components of X.

For a graph with n nodes, let $[n] = \{1, \dots, n\}$ denote its set of node indices and let $\mathfrak{S}_n$ be the *symmetric group* on $[n]$, i.e., the set of all $n!$ permutations of these indices. We call a graph *indexed* when its nodes have been assigned indices in $[n]$. These indices are used only to represent the graph as tensors and they have no semantic meaning and are distinct from the node features themselves.

We write an indexed graph as $g = (V, E)$, where V contains node features and E contains edge features. Nodes and edges may each carry multiple categorical and continuous feature channels. Following discrete graph generative formulations such as DeFoG (Qin et al., 2025a), connectivity is represented by a categorical edge channel with a binary edge / no-edge state. Additional edge channels may encode categorical or continuous edge attributes.

For undirected graphs, $E_{ij} = E_{ji}$ and an edge is represented once for each unordered pair $i < j$. For directed graphs, E_{ij} and E_{ji} are distinct and an edge is represented for each ordered pair $i \neq j$. In either case, let $\mathcal{G}_n$ denote the resulting space of indexed attributed graphs with n nodes.

A permutation $\pi \in \mathfrak{S}_n$ changes the node indexing while carrying all node and edge features with it. If P_π denotes its permutation matrix, we write

$$\pi \cdot g = \left(P_\pi V,\ P_\pi E P_\pi^\top\right), \tag{1}$$

where the same permutation is applied to every node- and edge-feature channel. Indexed graphs differing only by such a permutation represent the same graph. The *graph isomorphism class* of g is

$$[g] = \{\pi \cdot g : \pi \in \mathfrak{S}_n\}, \tag{2}$$

and we define the space of unindexed graphs with n nodes as

$$\mathcal{U}_n = \mathcal{G}_n/\mathfrak{S}_n := \{[g] : g \in \mathcal{G}_n\}, \qquad \mathcal{U} = \bigcup_{n \geq 1} \mathcal{U}_n,$$

where $\mathcal{G}_n/\mathfrak{S}_n$ denotes the quotient of $\mathcal{G}_n$ under the permutation action in Eq. 1. All node and edge features are included in the isomorphism relation.

Probabilistic prediction. Our formulation takes the target of SGP to be a conditional distribution

$$p^*(u \mid x), \qquad u \in \mathcal{U}, \tag{3}$$

over graph isomorphism classes. A training example, however, provides an indexed graph $g \in \mathcal{G}_n$ representing its realized target class $u = [g]$. Its node indexing may be arbitrary: we do not assume that it is canonical, random, or consistent across examples. Thus the modeled quantity is invariant to node indexing, while the training data expose only one indexed representative of each realized target.

For deterministic SGP, $p^*(u \mid x)$ is concentrated on a single class. More generally, it may assign probability to several non-isomorphic graphs that are compatible with the same input. The probabilistic formulation therefore includes conventional deterministic SGP as a special case while also allowing uncertainty over graph structure.

When the class distribution is discrete and a single graph is required, we consider graph-level 0–1 loss

$$\ell_{\mathrm{GI}}(\hat{u}, u) = \mathbf{1}\{\hat{u} \neq u\}. \tag{4}$$

Proposition 1 (Bayes-optimal SGP)**.** *For a fixed input x, any maximum a posteriori (MAP) class*

$$u_{\mathrm{MAP}}(x) \in \arg\max_{u \in \mathcal{U}} p^*(u \mid x) \tag{5}$$

minimizes the conditional expected graph-level zero–one loss.

Proof. Let U denote the target graph isomorphism class, with conditional law $p^*(\cdot \mid x)$. For any predicted class $\hat{u}$,

$$\mathbb{E}[\ell_{\mathrm{GI}}(\hat{u}, U) \mid X = x] = \Pr(U \neq \hat{u} \mid X = x) = 1 - p^*(\hat{u} \mid x).$$

Minimizing this risk is therefore equivalent to maximizing $p^*(\hat{u} \mid x)$, as in Eq. 5. □

Thus modeling a distribution does not preclude conventional point prediction: when one graph is required, its class mode is the Bayes-optimal decision under graph-isomorphism accuracy. Proposition 1 later motivates our sample-based modal decoder.

Variable graph size. The number of output nodes is itself part of the prediction problem. Since all representatives of an isomorphism class have the same size, $p^*(u \mid x)$ induces a size distribution $p^*(n \mid x)$ and a conditional class distribution $p^*(u \mid n, x)$ on $\mathcal{U}_n$. We factor the model, for $u \in \mathcal{U}_n$, as

$$p_{\theta,\psi}(u \mid x) = p_\theta(u \mid n, x)\, p_\psi(n \mid x), \tag{6}$$

where $p_\psi(n \mid x)$ is a categorical distribution over graph size and $p_\theta(u \mid n, x)$ models the graph conditional on that size. During training, n is observed from the target graph and p_ψ is trained with cross-entropy. At sampling time, we first draw $n \sim p_\psi(\cdot \mid x)$ and then draw a graph conditional on (n, x). When a deterministic size prediction is desired, we instead use the mode of p_ψ.

2.2 CONDITIONAL GRAPH FLOW PREDICTOR

We instantiate the probabilistic formulation above using conditional flow matching on indexed graph representations. For a training example (x, g) with n nodes, we write $G_1 = g$ for the clean endpoint and G_t for an intermediate graph at time $t \in [0, 1]$. The source distribution $q_0(\cdot \mid n)$ is permutation invariant and therefore assigns no special meaning to any node index.

For each categorical node or edge feature coordinate a, we use the linear discrete flow matching path

$$q_t(G_t^a = j \mid G_1^a = k) = (1 - t)q_0^a(j) + t\,\mathbf{1}\{j = k\}, \qquad t \in [0, 1], \tag{7}$$

following Campbell et al. (2024) and DeFoG (Qin et al., 2025a). During training, we sample $t \sim \mathrm{Unif}[0, 1]$ and train a time-conditioned denoiser to predict the clean endpoint. Continuous node or edge features are handled using a continuous space linear interpolant flow model (Lipman et al., 2023), combined with discrete flow matching following the mixed discrete and continuous construction of Campbell et al. (2024), with an endpoint squared error objective. The full formulation and sampling equations are given in Appendix E.

Let $\mathcal{A}_{\mathrm{cat}}$ and $\mathcal{A}_{\mathrm{cont}}$ index the categorical and continuous graph variables, respectively, with G^a and G^b denoting the corresponding categorical and continuous variables. We train with

$$\min_\theta \mathcal{L}_{\mathrm{flow}}(\theta) = \mathbb{E}\left[- \sum_{a \in \mathcal{A}_{\mathrm{cat}}} w_a \alpha_a(G_1^a) \log \widehat{p}_\theta^a\,(G_1^a \mid G_t, t, x, n) + \sum_{b \in \mathcal{A}_{\mathrm{cont}}} \lambda_b \left\| \widehat{G}_{1,\theta}^b - G_1^b \right\|_2^2 \right], \tag{8}$$

where the expectation is over $(x, G_1) \sim \mathcal{D}$, $t \sim \mathrm{Unif}[0, 1]$, and $G_t \sim q_t(\cdot \mid G_1, n)$; $\widehat{G}_{1,\theta}^b$ abbreviates $\widehat{G}_{1,\theta}^b(G_t, t, x, n)$. Positive weights w_a, λ_b balance feature losses, while $\alpha_a(k) > 0$ weights categories.

For undirected outputs, we use the permutation-equivariant graph transformer of DeFoG (Qin et al., 2025a), and for directed outputs, we use the directed architecture of DIRECTO (Carballo-Castro et al., 2026). We condition either denoiser on x by cross-attending graph states to tokens produced by a modality-specific input encoder. Further details about the architecture are given in Appendix C.

Crucially, Eq. 8 is evaluated directly against the indexed target G_1 provided by the dataset. Training requires no permutation search, node correspondence, transport plan, or graph matching loss.

2.3 WHY GRAPH MATCHING IS UNNECESSARY

The apparent difficulty with the preceding objective is that it compares coordinates against an arbitrary indexing of the target graph. We show that this indexing does not affect the expected objective. Let $\mathcal{R}_\theta(x, g)$ denote the expected training loss for input x and clean indexed target g, where the expectation is over the sampled flow time and intermediate graph state.

Proposition 2 (Representative invariance)**.** *Suppose the conditional probability path and denoiser are permutation equivariant, and the endpoint loss is invariant when the same node permutation is applied to both its prediction and target. Then, for every parameter setting θ, input $x \in \mathcal{X}$, graph $g \in \mathcal{G}_n$, and permutation $\pi \in \mathfrak{S}_n$,*

$$\mathcal{R}_\theta(x, \pi \cdot g) = \mathcal{R}_\theta(x, g). \tag{9}$$

Thus replacing a target by any other indexing of the same attributed graph leaves its expected training loss unchanged. For any training set, the objective obtained by averaging $\mathcal{R}_\theta(x, g)$ over examples is therefore invariant to independently reindexing each target. This holds for every model parameter setting and does not depend on successful optimization. The argument follows from applying the same permutation to the clean target, its intermediate graph state, and the denoiser prediction. Equivariance ensures that reindexing the target correspondingly reindexes the corruption distribution, while simultaneously reindexing the prediction and target merely reorders the coordinate-wise loss terms. The full proof and assumptions verification are in Appendix F.

2.4 PREDICTION

At inference, we first choose a size n, initialize a source graph from $q_0(\cdot \mid n)$, and generate an indexed graph conditional on (n, x) using the sampling dynamics. For *single-sample* predictions, we use the

most probable size, $\hat{n}(x) \in \arg\max_n p_\psi(n \mid x)$, and generate one graph conditional on $(\hat{n}(x), x)$. Categorical features evolve according to the corresponding CTMC, while continuous features are integrated jointly with their conditional velocity field. More sampling details are in Appendix E.

When one graph must be returned, Proposition 1 motivates selecting the most probable isomorphism class. We therefore introduce a sample-based *modal decoder* that approximates this class mode. We generate K independent and identically distributed (IID) candidates conditional on x: for each candidate, we independently draw $n \sim p_\psi(\cdot \mid x)$ and generate a graph conditional on (n, x). We group the resulting candidates according to their attribute-aware Weisfeiler–Lehman (WL) hashes (Shervashidze et al., 2011) and return a representative from the largest group. Thus, the frequency of each hash group provides an empirical estimate of the probability mass assigned to the corresponding graph class, and the largest group approximates the predictive class mode. We use WL hashes rather than exact graph isomorphism tests to avoid introducing an additional costly matching step. As a heuristic for continuous attributes, quantization is used to construct the WL hashes used for clustering, but the returned graph maintains its original continuous values. Because WL hashes are not complete isomorphism invariants, non-isomorphic graphs can occasionally collide. Together with finite sampling, this makes the procedure an approximation of the mode.

Complexity. For fixed model width, depth, and number of condition tokens, the dense graph denoiser has $O(n^2)$ graph-side time and memory complexity per training example because it maintains and updates representations for all n nodes and all $O(n^2)$ potential node pairs. Training evaluates the denoiser at one sampled time and introduces no graph alignment stage.

Generating one graph with S integration steps costs $O(Sn^2)$, and generating K candidates costs $O(KSn^2)$. For a fixed number of WL iterations, computing the WL hashes costs $O(K(n+m))$, where m is the number of edges in a generated graph (Shervashidze et al., 2011); in the worst case this is $O(Kn^2)$. Thus modal decoding, including both generation and WL clustering, has overall complexity $O(KSn^2)$. However, note that the K candidates are independent and can therefore be generated in parallel. The task-specific input encoder is separate from this graph-side complexity.

3 Experiments

We evaluate PSGP on six established SGP benchmarks, study how runtime and predictive performance scale with the number of output nodes, and introduce a new graph prediction task with known non-identifiable target families to test whether PSGP can recover a non-degenerate conditional distribution. Additional results and full experimental details are provided in the Appendices.

3.1 Deterministic supervised graph prediction benchmarks

We evaluate on six established SGP tasks from the Any2Graph benchmark (Krzakala et al., 2024). *Coloring 10* and *Coloring 15* map images of colored regions to their adjacency graphs; *QM9* (Wu et al., 2018) and *GDB13* (Blum & Reymond, 2009) reconstruct molecular graphs from molecular fingerprints; and *Toulouse* (Belli & Kipf, 2019) and *USCities* (He et al., 2020) map satellite segmentation images to road graphs. The first four tasks have categorical node features, whereas in the road tasks they are continuous. Representative input–target pairs for these tasks are shown in Figure 7.

Our primary metric is graph-isomorphism accuracy (GI-Acc), the fraction of point predictions that recover the target attributed graph up to node reindexing. On categorical graphs, this is one minus the empirical graph-level zero–one risk ℓ_{GI} introduced in Section 2.1. Proposition 1 therefore identifies the class MAP as the Bayes-optimal predictor for this metric. For the road tasks, we use the benchmark's tolerance relaxation, which also accepts matched node coordinates within Euclidean distance δ (using the same values as Any2Graph). Full metric definitions are given in Appendix D.

We compare against FGWBary (Brogat-Motte et al., 2022), Relationformer (Shit et al., 2022), Any2Graph (Krzakala et al., 2024), and GRALE (Krzakala et al., 2025). For Coloring 10, Coloring 15, QM9, and GDB13, we use the published results collected in Table 4 of the GRALE paper, and Toulouse and USCities use the published results from the Any2Graph paper. FGWBary is reported only where available and is given the true output graph size at inference, making its results an oracle-size comparison. For PSGP, *single* denotes one generated graph candidate, without aggregation or selection among multiple candidates. We use the highest-probability predicted graph size and generate the graph conditional on that size using the finite sampler. *modal@32* applies the introduced modal decoder with 32 samples.

Table 1: **GI-Acc (%) on established SGP benchmarks.** Best and second-best results are shown in bold and underlined, respectively. * denotes that the method was given the true output graph size at inference. "–" indicates that the method was not reported on that task.

Method	Coloring 10	Coloring 15	QM9	GDB13	Toulouse	USCities
FGWBary*	1.00	–	28.95	–	0.00	–
Relationformer	18.14	21.99	9.95	0.05	93.28	55.00
Any2Graph	84.44	43.77	29.85	16.25	93.62	58.10
GRALE	89.66	86.02	30.77	32.02	–	–
GRALE + finetuning	90.04	88.87	35.27	53.22	–	–
PSGP single	<u>96.21</u>	<u>98.92</u>	<u>50.83</u>	<u>68.63</u>	<u>95.77</u>	<u>77.68</u>
PSGP modal@32	**99.00**	**99.53**	**71.91**	**86.71**	**96.97**	**84.00**

Table 1 shows that a single PSGP sample already outperforms the strongest previously reported result on all six tasks. This comparison separates the performance of the single-candidate predictor from the additional benefit of multi-candidate modal decoding. Modal decoding then improves every task, with the largest gains on QM9 and GDB13, and almost saturating the Coloring tasks. Additional analyses are reported in Appendix A.

3.2 Scaling with output size

Avoiding graph matching is particularly relevant as the number of output nodes grows. We study this from two complementary perspectives: the computational cost of the training objective and predictive performance as the maximum output size increases. We use Any2Graph for this scaling study because its PMFGW objective directly instantiates the matching stage that PSGP removes.[1]

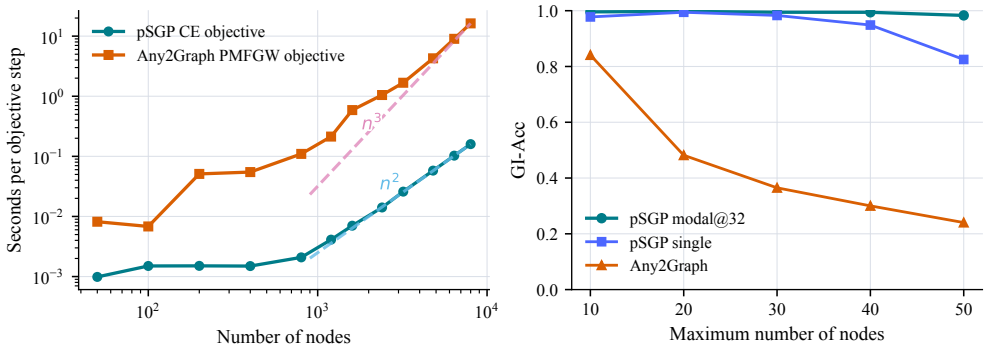


Figure 2: **Scaling with output graph size. Left:** runtime per optimization step (loss evaluation, backpropagation, and optimizer update) on synthetic dense graphs, comparing PSGP's coordinate-wise cross-entropy with Any2Graph's PMFGW objective. Dashed lines show quadratic and cubic reference scaling. **Right:** GI-Acc on Coloring tasks as the maximum number of nodes increases. Both single-sample and modal PSGP predictions degrade substantially more slowly than Any2Graph.

Figure 2 (left) isolates the cost of the output objective. The coordinate-wise node and edge losses used by PSGP follow the expected quadratic scaling of the dense graph representation, whereas the PMFGW alignment exhibits the cubic scaling of its optimal transport computation. The gap therefore grows rapidly with the number of nodes. This experiment excludes the predictors themselves and isolates the cost of making supervision permutation-aware through graph matching.

The right panel tests how prediction quality changes as the output graph grows. We fix the Coloring input resolution at 32×32 and vary the maximum number of output nodes up to 50. Any2Graph degrades quickly with graph size, whereas a single PSGP sample and modal@32 remain substantially more stable. Together, the two experiments support the intended scaling advantage of training without graph matching: PSGP avoids the cubic alignment stage while conserving strong prediction quality as the output space grows.

[1]Our attempts to reproduce GRALE did not consistently recover the performance reported in the original paper, so we use its published results for the main benchmark comparison and exclude it from other analyses.

3.3 PROBABILISTIC SGP WITH NON-IDENTIFIABLE FAMILY STRUCTURES

The benchmarks above are designed primarily for point prediction: for each input, evaluation asks whether a model recovers one target graph. They do not test if a predictor can represent a non-degenerate distribution over graph structures. We introduce *Genotypes*, a controlled synthetic task with genotypes paired with explicitly specified target distributions over family graphs, including structures motivated by established non-identifiability results. The task also evaluates PSGP on directed acyclic graph outputs.

Genotypes task. The input consists of genotypes at single nucleotide polymorphism (SNP) sites for two or three observed individuals, represented as allele counts (0, 1, or 2). The target is a directed family graph with edges from parent to child. The graph nodes are the observed individuals, their unobserved lowest common ancestors, and the intermediate ancestors needed to connect them. Observed individuals have distinct node labels (a, b, or c), while unobserved ancestors share a common node label. We simulate genetic data for each family template using msprime (Baumdicker et al., 2022; Anderson-Trocmé et al., 2023) and retain the genotypes of the observed individuals. The dataset contains 100,000 training, 10,000 validation, and 10,000 test examples, with target graphs of at most eight nodes. Full simulation and target construction details are given in Appendix B.

The task is motivated by a fundamental identifiability limitation of family graph reconstruction. Donnelly (1983) showed that certain distinct cousin-type relationships cannot be distinguished from genetic information. Kirkpatrick (2012) expressed this ambiguity in terms of equal numbers of edges, and showed that this ambiguity can persist with three observed individuals, with two types of different family graphs having identical likelihoods.

Figure 3 shows representative family templates used in the benchmark. For each input, we specify a target distribution over family graph isomorphism classes. Generic examples place all probability on one family class. In the non-identifiable two-person family, the two observed individuals a and b descend from one shared ancestor with four total edges along the two descent paths. Following the notation from Kirkpatrick (2012), let d_a and d_b denote the numbers of edges from the shared ancestor to a and b, respectively. We define the target distribution to be uniform over the three possible ordered pairs $(d_a, d_b) \in \{(1,3),(2,2),(3,1)\}$, assigning probability $1/3$ to each corresponding class. In the three-person case with two non-identifiable classes, each receives probability $1/2$.

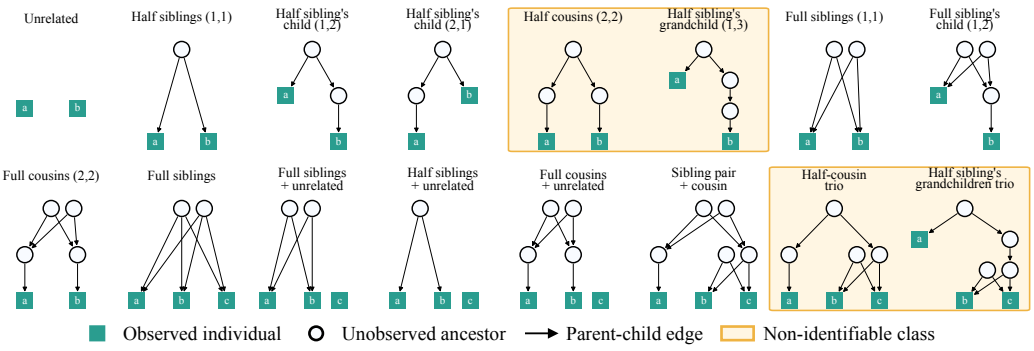


Figure 3: **Representative templates used in Genotypes.** The benchmark includes generic and non-identifiable templates for two and three observed individuals. Within each non-identifiable family, the evaluation reference distribution assigns probability to multiple non-isomorphic structures.

Distributional evaluation. GI-Acc against a single target does not assess a predictive distribution. We instead use squared maximum mean discrepancy (MMD^2) (Gretton et al., 2012). For each condition, outputs outside the reference family are first mapped to a shared invalid class with zero reference probability. Let $\bar{p}_{\theta,\psi}$ and r denote the predictive and benchmark reference distributions on this collapsed class space. The reference is not the simulator's full posterior over family classes, which is generally intractable to evaluate exactly. Using the equality kernel $k_{\text{GI}}(u,v) = \mathbf{1}\{u = v\}$ on this space gives

$$\text{MMD}^2_{\text{GI}}(\bar{p}_{\theta,\psi}, r) = \mathbb{E}_{U,U' \sim \bar{p}_{\theta,\psi}} k_{\text{GI}}(U,U') - 2\mathbb{E}_{U \sim \bar{p}_{\theta,\psi}, V \sim r} k_{\text{GI}}(U,V) + \mathbb{E}_{V,V' \sim r} k_{\text{GI}}(V,V'). \quad (10)$$

This MMD^2 compares the probabilities of each valid graph class and the shared invalid class, and is zero exactly when the collapsed distributions agree.

For PSGP, we estimate Eq. 10 using the unbiased MMD^2 estimator from $K = 16$ IID generated samples per condition. For each sample, graph size is independently sampled from the size predictor and the graph is generated with independent sampling randomness. Modal decoding is not used for this distributional evaluation. Any2Graph produces a single deterministic prediction, which we interpret as a point-mass predictive distribution.

Table 2: **Distributional prediction on Genotypes.** MMD^2 uses equality on the collapsed class space; lower is better. PSGP is evaluated with $K = 16$ IID samples per condition.

Method	Overall	Generic	Non-identifiable
Any2Graph	0.383	0.313	0.774
PSGP	**0.225**	**0.196**	**0.388**

Table 2 shows that PSGP more closely matches the reference distributions over family structures. The improvement is largest on the non-identifiable subset, where representing multiple graph structures is important: a deterministic predictor places all probability on a single graph, whereas PSGP can distribute probability across multiple non-isomorphic family classes appropriately. For both models 100% of the predicted graphs are valid directed acyclic graphs.

4 Conclusion, Limitations and Broader Impact

We introduced PSGP, a probabilistic formulation of supervised graph prediction that learns conditional distributions over graph-isomorphism classes without aligning predicted and target nodes. By combining a permutation-invariant source distribution with permutation-equivariant conditional flow matching, PSGP can be trained directly on arbitrarily indexed graph targets using standard coordinate-wise losses. We established that the expected training objective is invariant to the indexed representative chosen for each target, for every model parameter setting and for categorical and continuous graph features. In contrast to matching approaches with $O(n^3)$ training, the dense implementation used here has $O(n^2)$ training complexity.

Across six established SGP benchmarks, a single PSGP sample outperforms the strongest previously reported result on every task, with further gains from sample-based modal decoding. Besides, controlled experiments show that prediction quality degrades substantially more slowly than Any2Graph as the maximum output size increases, while the coordinate-wise training objective exhibits quadratic rather than cubic scaling. On our family graph reconstruction task, PSGP substantially reduces $\mathrm{MMD}^2_{\mathrm{GI}}$, demonstrating the benefit of modeling a distribution when multiple non-isomorphic outputs are compatible with the same observations.

The dense graph transformer used here processes all potential node pairs. This quadratic scaling is a substantial improvement over the cubic matching used by prior SGP methods, but dense all-pairs processing may still become a bottleneck for substantially larger graphs. This limitation is specific to the present instantiation rather than of the probabilistic SGP formulation itself. Future work could combine the same formulation with sparsity-aware (Qin et al., 2025b) or hierarchical (Karami, 2024) graph generative models, whose denoising architecture can reduce per-pass graph computation toward subquadratic complexity. More generally, advances in scalable graph generation could be incorporated into the same formulation without reintroducing graph matching.

SGP arises in several settings where the output structure may itself be uncertain. A matching-free probabilistic approach can make such problems more tractable while representing multiple plausible graphs rather than forcing a single prediction. Graph predictions can, however, encode consequential relationships, and samples from a generative model should not be treated as calibrated posteriors. Our family graph reconstruction benchmark uses synthetic genotypes and controlled family templates to instantiate established non-identifiability phenomena with a known target distribution, without using real human data. Performance on this benchmark is not by itself evidence of suitability for real-world family graph inference, which would require richer population models, calibration, domain validation, and appropriate privacy considerations. Overall, matching-free probabilistic SGP provides a general approach toward modeling graph outputs across a broader range of structured prediction problems.

AI USE STATEMENT

In this work, generative AI was used to assist with proof development, provide feedback on research methodology and experiments, implement methods and create or edit code, support qualitative and thematic analysis, and interpret results. It was also used for manuscript drafting and editing, brainstorming, information search, and literature discovery.

The main idea and mathematical claims were formulated by the authors. Generative AI was not used to develop theoretical or conceptual frameworks, propose or refine hypotheses, or clean or reformat datasets. Translation was not applicable. Generative AI did not directly generate synthetic data, but assisted in writing code used by the authors to generate it.

All AI-assisted work was reviewed by the authors. Proofs were independently checked for mathematical correctness, code was reviewed and tested, and AI-assisted research suggestions were checked against prior work. The authors take responsibility for the final content and all AI-assisted outputs.

ETHICS STATEMENT

The family graph reconstruction benchmark contains only synthetic genotypes generated from predefined templates. It does not contain patient, consumer genetics, or other human subject data, and its purpose is only to create controlled ambiguity with a known target distribution. Performance on this benchmark should not be taken as evidence that the model is ready for forensic, medical, genealogical, or reproductive decisions. Real deployment would require substantially richer population models, calibrated uncertainty and privacy safeguards.

REPRODUCIBILITY STATEMENT

The appendices detail the data, model, sampling, and evaluation settings used in the reported experiments. An accompanying repository will be made open source upon acceptance of the paper for publication and will include the code for data generation, training, and evaluation, as well as configurations for all tasks, datasets, checkpoints, and full test results.

REFERENCES

Luke Anderson-Trocmé, Dominic Nelson, Shadi Zabad, Alex Diaz-Papkovich, Ivan Kryukov, Nikolas Baya, Mathilde Touvier, Ben Jeffery, Christian Dina, Hélène Vézina, et al. On the genes, genealogies, and geographies of Quebec. *Science*, 380(6647):849–855, 2023. doi: 10.1126/science.add5300. URL https://www.science.org/doi/10.1126/science.add5300.

Franz Baumdicker, Gertjan Bisschop, Daniel Goldstein, Graham Gower, Aaron P Ragsdale, Georgia Tsambos, Sha Zhu, Bjarki Eldon, E Castedo Ellerman, Jared G Galloway, Ariella L Gladstein, Gregor Gorjanc, Bing Guo, Ben Jeffery, Warren W Kretzschumar, Konrad Lohse, Michael Matschiner, Dominic Nelson, Nathaniel S Pope, Consuelo D Quinto-Cortés, Murillo F Rodrigues, Kumar Saunack, Thibaut Sellinger, Kevin Thornton, Hugo van Kemenade, Anthony W Wohns, Yan Wong, Simon Gravel, Andrew D Kern, Jere Koskela, Peter L Ralph, and Jerome Kelleher. Efficient ancestry and mutation simulation with msprime 1.0. *Genetics*, 220(3):iyab229, 2022. URL https://doi.org/10.1093/genetics/iyab229.

Davide Belli and Thomas Kipf. Image-conditioned graph generation for road network extraction. In *Graph Representation Learning Workshop at NeurIPS*, 2019. URL https://grlearning.github.io/papers/23.pdf.

Lorenz C. Blum and Jean-Louis Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *Journal of the American Chemical Society*, 131(25): 8732–8733, June 2009. ISSN 0002-7863. doi: 10.1021/ja902302h. URL https://doi.org/10.1021/ja902302h.

Xavier Bresson and Thomas Laurent. A two-step graph convolutional decoder for molecule generation. In *Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019)*, 2019. URL https://ml4physicalsciences.github.io/2019/files/NeurIPS_ML4PS_2019_93.pdf.

Luc Brogat-Motte, Rémi Flamary, Celine Brouard, Juho Rousu, and Florence D'Alché-Buc. Learning to predict graphs with fused Gromov-Wasserstein barycenters. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2321–2335. PMLR, 2022. URL https://proceedings.mlr.press/v162/brogat-motte22a.html.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=kQwSbv0BR4.

Alba Carballo-Castro, Manuel Madeira, Yiming Qin, Dorina Thanou, and Pascal Frossard. Generating directed graphs with dual attention and asymmetric encoding. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=s2s5xGQCKM.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision – ECCV 2020*, pp. 213–229, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58452-8. URL https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123460205.pdf.

Kevin P. Donnelly. The probability that related individuals share some section of genome identical by descent. *Theoretical Population Biology*, 23(1):34–63, 1983. ISSN 0040-5809. doi: 10.1016/0040-5809(83)90004-7. URL https://www.sciencedirect.com/science/article/pii/0040580983900047.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL http://jmlr.org/papers/v13/gretton12a.html.

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, pp. 11 – 15, 2008. URL https://networkx.org.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

Songtao He, Favyen Bastani, Satvat Jagwani, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Mohamed M. Elshrif, Samuel Madden, and Mohammad Amin Sadeghi. Sat2Graph: Road graph extraction through graph-tensor encoding. In *Computer Vision – ECCV 2020*, pp. 51–67, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58586-0. URL https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123690052.pdf.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. URL https://doi.org/10.48550/arXiv.1606.08415.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Mahdi Karami. Higen: Hierarchical graph generative networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=KNvubydSB5.

Bonnie Kirkpatrick. Non-identifiable pedigrees and a bayesian solution. In *Bioinformatics Research and Applications*, pp. 139–152, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-30191-9. URL https://dl.acm.org/doi/10.1007/978-3-642-30191-9_14.

Paul Krzakala, Junjie Yang, Rémi Flamary, Florence d'Alché Buc, Charlotte Laclau, and Matthieu Labeau. Any2Graph: Deep end-to-end supervised graph prediction with an optimal transport loss. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=tPgagXpvcV.

Paul Krzakala, Gabriel Melo, Charlotte Laclau, Florence d'Alché Buc, and Rémi Flamary. The quest for the GRAph Level autoEncoder (GRALE). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=V8HhrBSsEn.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. URL https://doi.org/10.1002/nav.3800020109.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Igor Melnyk, Pierre Dognin, and Payel Das. Knowledge graph generation from text. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 1610–1622, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.116/.

Facundo Mémoli. Gromov–Wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011. URL https://doi.org/10.1007/s10208-011-9093-5.

Gabriel Peyré and Marco Cuturi. Computational optimal transport with applications to data sciences. *Foundations and Trends in Machine Learning*, 11(5-6):355–607, February 2019. ISSN 1935-8237. URL https://doi.org/10.1561/2200000073.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *Forty-second International Conference on Machine Learning (ICML)*, 2025a. URL https://openreview.net/forum?id=KPRIwWhqAZ.

Yiming Qin, Clement Vignac, and Pascal Frossard. SparseDiff: Sparse discrete diffusion for scalable graph generation. *Transactions on Machine Learning Research*, 2025b. ISSN 2835-8856. URL https://openreview.net/forum?id=kuJ3lpxnVC.

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-Lehman graph kernels. *Journal of Machine Learning Research*, 12(77): 2539–2561, 2011. URL http://jmlr.org/papers/v12/shervashidze11a.html.

Suprosanna Shit, Rajat Koner, Bastian Wittmann, Johannes Paetzold, Ivan Ezhov, Hongwei Li, Jiazhen Pan, Sahand Sharifzadeh, Georgios Kaissis, Volker Tresp, et al. Relationformer: A unified framework for image-to-graph generation. In *Computer Vision – ECCV 2022*, pp. 422–439. Springer, 2022. URL https://www.ecva.net/papers/eccv_2022/papers_ECCV/papers/136970414.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Titouan Vayer, Laetitia Chapel, Remi Flamary, Romain Tavenard, and Nicolas Courty. Fused Gromov-Wasserstein distance for structured objects. *Algorithms*, 13(9), 2020. ISSN 1999-4893. URL https://doi.org/10.3390/a13090212.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UaAD-Nu86WX.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yuxin Wu and Kaiming He. Group normalization. In *Computer Vision – ECCV 2018*, September 2018. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Yuxin_Wu_Group_Normalization_ECCV_2018_paper.html.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: a benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, January 2018. ISSN 2041-6520. doi: 10.1039/c7sc02664a. URL https://doi.org/10.1039/c7sc02664a.

Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence*, 2(2):109–127, 2021. URL https://doi.org/10.1109/TAI.2021.3076021.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/xiong20b.html.

Yanqiao Zhu, Yuanqi Du, Yinkai Wang, Yichen Xu, Jieyu Zhang, Qiang Liu, and Shu Wu. A survey on deep graph generation: Methods and applications. In *The First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pp. 47:1–47:21. PMLR, 2022. URL https://openreview.net/forum?id=Im8G9R1boQi.

A ADDITIONAL RESULTS

A.1 A QUALITATIVE EXAMPLE OF MODAL DECODING

In order to illustrate how modal decoding works, Figure 4 shows the pool of 32 predictions for one Coloring 10 input. Several competing structures appear, but one cluster clearly dominates. This example displays how variation among individual samples can coexist with a clear modal prediction. The following experiments examine how often this behavior leads to the correct graph and which decoding choices affect its reliability.

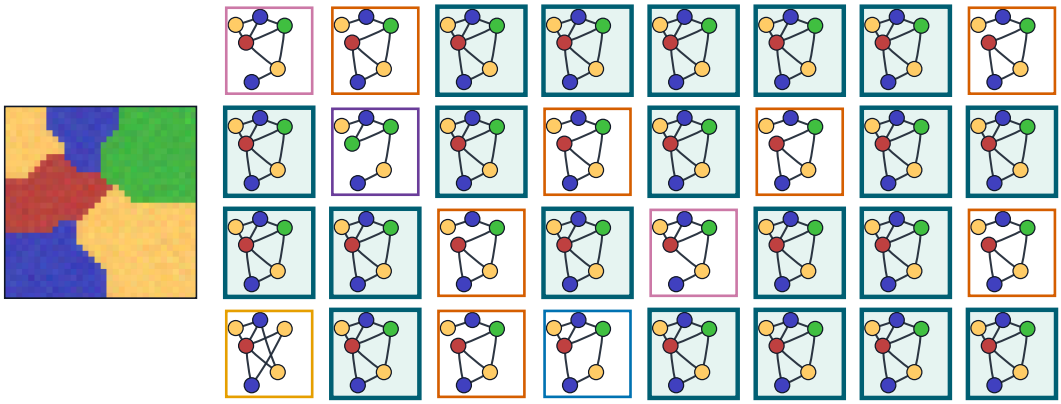


Figure 4: **An example of modal decoding on Coloring 10.** The input image appears on the left and the 32 generated candidates on the right. Node colors denote categorical attributes. Matching border colors indicate the same WL cluster. The highlighted cluster contains 20 candidates, compared with seven in the second largest cluster, and supplies the modal prediction.

A.2 SIZE PREDICTION, CLUSTERING, AND POOL COVERAGE

The table ablates three aspects of prediction: the accuracy of one draw (*single*), the accuracy of modal selection from 32 candidates (*modal@32*), and the fraction of pools containing at least one exact target match (*any@32*). The last quantity is a diagnostic to measure what an oracle selector could achieve using the same pool, but is not a prediction rule available in practice. We show results computed on validation samples from the Coloring 15, QM9 and GDB13 tasks.

Note that by default, including for the results in Table 1, PSGP *single* uses MAP size prediction and *modal@32* uses IID size prediction for all candidates.

Predicting the number of nodes. *MAP size* fixes the most probable predicted size for every candidate. *IID size* independently samples each candidate's size from $p_\psi(n \mid x)$. *Oracle size* supplies the true target size, which is generally unavailable at inference. An incorrect MAP size prevents the entire pool from containing the target, whereas IID sampling can preserve alternative sizes.

Using WL clustering, Coloring 15 is already near saturation and is largely insensitive to the size policy. On GDB13, IID size improves modal accuracy and comes within 0.5 percentage points of oracle size. On QM9, IID size improves pool coverage but slightly reduces modal accuracy. Thus, covering more possible sizes does not necessarily make the target the most frequent candidate at a fixed pool size. Providing the true size improves QM9 modal accuracy by 8.4 percentage points relative to IID size, identifying size prediction as an important remaining source of error on this task.

WL clustering. WL grouping and exact attributed graph isomorphism give nearly identical modal accuracies, differing by at most 0.3 percentage points across the reported settings. This suggests that WL collisions have little effect on aggregate accuracy in these experiments, supporting WL grouping as an efficient approximation for modal decoding.

Target coverage versus modal selection. Any@32 is consistently higher than modal@32 on the molecular tasks. This means that the correct graph is often present in the sampled pool even when modal decoding does not return it. For example, under IID size sampling on GDB13, the target appears in 98.1% of pools while modal accuracy is 88.5%. The remaining gap therefore reflects cases where another graph class receives more samples than the target class, rather than cases where the target is never generated. This highlights the quality of PSGP's candidate generation and the opportunity to improve the selection of a single prediction.

Table 3: **Modal decoding ablation on validation examples.** All values are GI-Acc percentages, with correctness checked by exact attributed graph isomorphism. "Cluster" specifies how candidates are grouped for selection, not how correctness is evaluated. Single uses one graph. Modal@32 and any@32 use pools of 32 graphs. Any@32 measures exact target presence and requires an oracle to select a matching candidate. Single and any@32 do not depend on the clustering rule and are shown only once where reported. Bold marks the best modal result using WL clustering and predicted (MAP or IID) rather than oracle size for each task to highlight only results obtainable in practice.

Task	Size decoder	Cluster	Single ↑	Modal@32 ↑	Any@32 ↑
Coloring 15					
	MAP size	WL	99.4	99.7	99.8
		Exact GI		99.7	
	IID size	WL		**99.8**	99.8
		Exact GI		99.8	
	Oracle size	WL	99.4	99.8	100.0
		Exact GI		99.8	
QM9					
	MAP size	WL	53.4	**71.1**	86.4
		Exact GI		71.0	
	IID size	WL		70.6	89.0
		Exact GI		70.4	
	Oracle size	WL	56.4	79.0	96.9
		Exact GI		78.9	
GDB13					
	MAP size	WL	68.3	86.7	97.6
		Exact GI		86.8	
	IID size	WL		**88.5**	98.1
		Exact GI		88.6	
	Oracle size	WL	70.4	89.0	99.8
		Exact GI		89.3	

A.3 Inference budget and computational cost

In order to understand the effects of K and S on prediction accuracy, and how these effects vary with graph size, we study inference behavior on the model checkpoint trained for Figure 2 (right) with graphs of up to 50 nodes. We use this setting because the scaling results show that prediction accuracy is not yet saturated for larger graphs, leaving room to study how additional inference computation is used. At the same time, Coloring provides a controlled synthetic task in which graph size can be varied directly and the relation between the input image and output graph is easy to interpret.

Both sweeps evaluate the same 1,000 validation examples. The K sweep fixes $S = 250$ and uses MAP predicted size for all candidates of a condition. For each K, we evaluate the first K candidates from a pool of 128 samples. The S sweep instead generates one fresh sample for each value of S. The two sweeps therefore use the same checkpoint and conditions, but do not reuse the same generated samples.

More steps versus more candidates. We stratify the validation examples by their true graph size in Figure 5. The left panel shows that smaller graphs reach high accuracy with relatively few integration steps, whereas larger graphs remain difficult for an individual draw. For graphs with 41–50 nodes, single sample GI-Acc is 50.8% at $S = 250$ and 54.1% at $S = 1000$. Thus, a fourfold increase in integration steps alone leaves substantial errors within the tested range.

In Figure 5 (right), increasing the pool size has a much larger effect on the largest graphs. For graphs with 41–50 nodes, modal accuracy rises from 38.9% at $K = 1$ to 91.9% at $K = 32$ and 97.3% at $K = 128$. These gains occur with the model parameters, integration steps, and predicted graph size

held fixed. The correct structure is therefore generated often enough for repeated sampling to recover it reliably, even when a single draw is unreliable.

Connection to complexity. The $O(KSn^2)$ cost in Section 2.4 clarifies the distinction between these two uses of computation. Increasing S refines each sampling trajectory, whereas increasing K supplies more complete predictions for aggregation. Both increase the dominant generation cost linearly. For fixed S and K, the dependence on graph size remains quadratic, including WL grouping.

The black curves in Figure 5 allow a comparison at the same number of denoiser evaluations. The $K = 4$ point in the right panel uses four candidates with $S = 250$ and achieves 91.0% overall GI-Acc. The $S = 1000$ point in the left panel uses one candidate and achieves 87.6%. Both use $KS = 1000$ denoiser evaluations per input. This suggests that allocating part of the generation budget to repeated sampling can be more effective than spending it entirely on one trajectory in this setting. It is not a comparison of wall clock runtime, which also depends on batching and clustering overhead. The higher accuracies at $K = 32$ and $K = 128$ require correspondingly more total computation. Nonetheless, increasing K also admits parallel generation across candidates, whereas increasing S lengthens each sequential sampling trajectory. Thus, the two allocations need not have the same wall-clock cost, even when they use the same number of denoiser evaluations.

This use of additional computation at inference parallels self consistency in language models, where aggregating answers from multiple sampled reasoning paths improves reasoning accuracy (Wang et al., 2023).

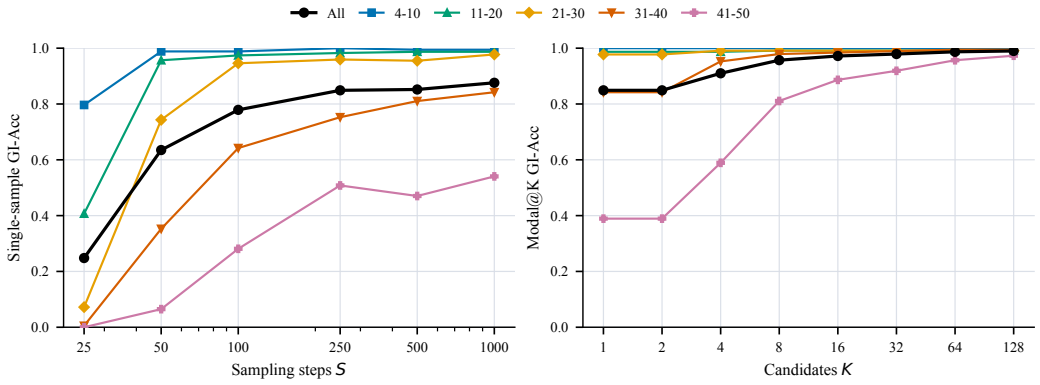


Figure 5: **Inference budget and graph size on Coloring. Left:** single sample GI-Acc as the number of integration steps S increases. **Right:** modal@K GI-Acc as the number of candidates K increases at fixed $S = 250$. Both panels use the same checkpoint and 1,000 validation conditions with MAP size prediction, but separately generated samples. Curves average accuracy within bins of true graph size. Black curves average over all conditions. Repeated sampling particularly benefits the largest graphs.

A.4 Understanding the modal vote

Figure 6 examines the same pools to distinguish target coverage, concentration, and ambiguity in the vote. Each statistic is computed for each condition and then averaged over the 1,000 conditions.

Pool coverage and selected accuracy. The left panel of Figure 6 compares target presence in the pool with modal accuracy. At $K = 32$, a candidate with the target's WL hash appears in 99.2% of pools, while modal GI-Acc is 97.9%. At $K = 128$, these values reach 99.4% and 99.0%. Thus, modal selection converts most of the observed coverage into correct predictions. The coverage curve is an optimistic proxy for exact target presence because graphs that are not isomorphic can share a WL hash. Its gap from modal GI-Acc can therefore reflect both selection errors and WL collisions.

How decisive is the winning cluster? Let c_1 and c_2 be the sizes of the largest and second largest WL clusters, with $c_2 = 0$ when only one cluster exists. The middle panel reports the mean largest

cluster share c_1/K and the mean normalized gap $(c_1 - c_2)/K$. The share measures how many samples support the selected prediction. The gap measures its lead over the strongest alternative.

For example, at $K = 32$, cluster sizes $c_1 = 27$ and $c_2 = 3$ give $\frac{c_1 - c_2}{K} = \frac{27-3}{32} = 0.75$, indicating a decisive vote. In contrast, $c_1 = 12$ and $c_2 = 11$ give $\frac{c_1 - c_2}{K} = \frac{12-11}{32} \approx 0.031$, so changing only a few samples could change the selected cluster.

At $K = 32$, the average largest cluster share is 0.841 and the average gap is 0.804. These values indicate substantial agreement and separation from competing predictions across the evaluation set. The gap measures how decisive the modal vote is, but not whether the winning cluster is correct; modal GI-Acc is reported separately in the left panel of Figure 6.

Modal decoding also does not require the winning cluster to contain a majority of the samples, only to be larger than every competing cluster. For the 41–50 node subset at $K = 32$, the average largest cluster contains 43.4% of the samples, while the corresponding modal accuracy in Figure 5 (right) is 91.9%. Thus, the target class can be selected reliably even when the remaining probability mass is spread across several alternative graph classes.

Unique samples and ties. For $K \geq 2$, if all candidates belong to distinct WL clusters, then all clusters have size one and are therefore tied for largest. However, a tie can also occur when some candidates repeat. For example, cluster sizes $(3, 3, 1, 1)$ contain repeated samples but still have two clusters tied for largest size.

At $K = 1$, uniqueness is automatic and there is no tie between clusters. At $K = 2$, the rule that resolves ties by earliest sample index always returns the first candidate, explaining why modal accuracy is unchanged from $K = 1$.

The right panel of Figure 6 shows that, across all conditions, the fractions of pools with all samples unique and with a tied largest cluster fall to 2.2% and 3.0% at $K = 8$, and to 0.1% and 0.6% at $K = 32$. Thus, with a moderate pool, selection is usually determined by repeated agreement rather than the rule for breaking ties. Together with the accuracy results, these diagnostics show why PSGP can make reliable predictions from a pool even when individual samples remain imperfect.

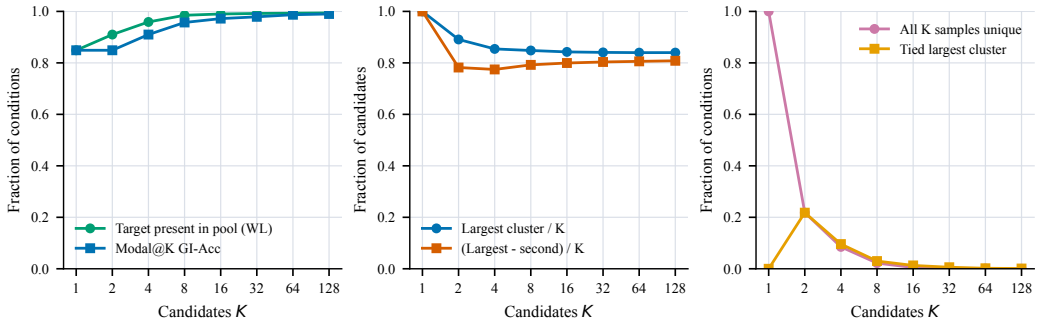


Figure 6: **Candidate pool diagnostics on Coloring.** All panels use $S = 250$, MAP size prediction, and the same 1,000 validation conditions as Figure 5. **Left:** exact modal GI-Acc and the fraction of pools containing a candidate with the target's WL hash. **Middle:** the average largest cluster share c_1/K and normalized gap $(c_1 - c_2)/K$, with $c_2 = 0$ for a single cluster. **Right:** the fractions of pools with all candidates in distinct clusters or with multiple clusters tied for the largest size. All statistics are averaged over conditions.

B DATASETS

We use six SGP benchmarks from the Any2Graph benchmark family (Krzakala et al., 2024) and introduce the Genotypes benchmark for probabilistic SGP. Representative input–target pairs for all seven tasks are shown in Figure 7. We retain the arbitrarily indexed graph representation supplied by each dataset.

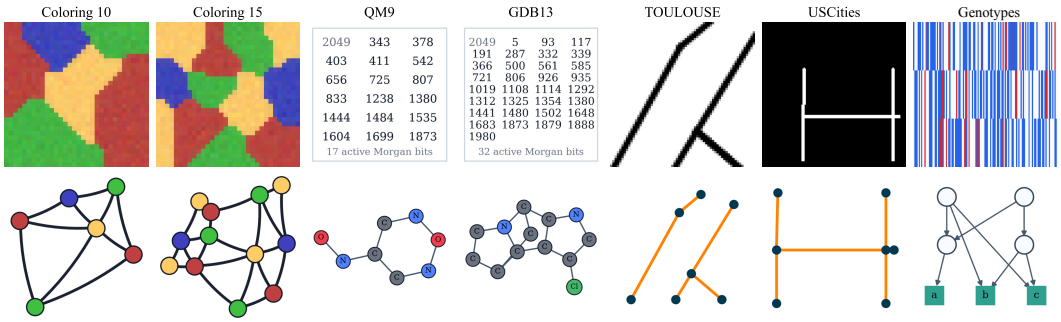


Figure 7: Representative input–target pairs for the supervised graph prediction tasks. The upper row shows the supervised input x, and the lower row shows the corresponding target graph. Coloring 10 and Coloring 15 map images of colored regions to region-adjacency graphs, with node colors indicating the region labels. QM9 and GDB13 map radius-2 Morgan fingerprints to molecular graphs. The fingerprint input is shown as the sorted indices of the active bins in the 2,048-bin fingerprint, preceded by a start token, and the number of active Morgan bits is indicated below each example. Toulouse and USCities map grayscale road segmentation images to undirected road graphs with continuous two-dimensional node coordinates. For Genotypes, columns correspond to SNP sites in genomic order and rows correspond to observed individuals, with white, blue, and red indicating genotype dosages 0, 1, and 2, respectively. Only a window of the SNP sequence is shown because the full input can contain up to 2,047 SNP sites. The target is a directed family graph in which square nodes denote observed individuals, circular nodes denote unobserved ancestors, and arrows point from parent to child.

B.1 Coloring

The Coloring task maps a rendered image of colored regions to the region-adjacency graph introduced in the Any2Graph benchmark (Krzakala et al., 2024). Each graph node carries one of four categorical color labels and each edge records whether two regions are adjacent. The generator uses Gaussian filtering scale 0.4, noise scale 0.02, and adjacency thresholds of 3.

Coloring 10 uses the Any2Graph Coloring Small data comprised of 32×32 images with at most 10 nodes, and 100,000 training, 10,000 validation, and 10,000 test examples. As in GRALE (Krzakala et al., 2025), Coloring 15 uses 64×64 images with 5–15 nodes, and includes 300,000 training, 10,000 validation, and 10,000 test examples.

For the performance scaling experiment in Subsection 3.2, we use the same Coloring data generator as in the main benchmark tasks, and generate datasets with maximum node counts $M_{\max} \in \{10, 20, 30, 40, 50\}$. In all cases, graphs contain between 4 and $M_{\max}$ nodes, images have resolution 32×32, and the train/validation/test splits contain 300,000/10,000/10,000 examples. We keep the image resolution fixed at 32×32 so that the experiment isolates the effect of increasing graph size rather than conflating graph size with input resolution.

B.2 Fingerprint-to-molecule benchmarks

For QM9 (Wu et al., 2018) and GDB13 (Blum & Reymond, 2009), we use the Fingerprint2Graph train, validation, and test splits from the Any2Graph benchmark (Krzakala et al., 2024).

To obtain the input, the Morgan fingerprint with radius 2 and 2,048 hash bins is computed and the nonzero bin indices are represented as a token sequence. The active indices are sorted, preceded by a start token, and padded to a maximum sequence length of 100. If necessary, at most the first 99 active fingerprint indices are retained in addition to the start token.

The output graph encodes molecular adjacency and contains atom types as node features. QM9 uses the atom alphabet $\{\mathrm{C}, \mathrm{N}, \mathrm{O}, \mathrm{F}\}$, while GDB13 uses $\{\mathrm{C}, \mathrm{N}, \mathrm{O}, \mathrm{S}, \mathrm{Cl}\}$. Molecules exceeding the configured maximum of 10 atoms for QM9 or 15 atoms for GDB13 are excluded.

B.3 ROAD GRAPH BENCHMARKS

We use the Toulouse (Belli & Kipf, 2019) and USCities (He et al., 2020) road graph data distributed with the Any2Graph Sat2Graph benchmark (Krzakala et al., 2024). In both tasks, the target graph has continuous two-dimensional node coordinates and undirected road connectivity.

We load the preprocessed grayscale road segmentation images and graph data splits used by Any2Graph, where node features represent two-dimensional node coordinates, and graphs exceeding 11 nodes for Toulouse or 17 nodes for USCities are excluded.

B.4 GENOTYPES

The Genotypes benchmark is designed to evaluate prediction of a non-degenerate conditional distribution over directed graph structures. Each condition contains unphased diploid genotypes for two or three observed individuals, and the output is a directed family graph with edges from parent to child. Dataset generation uses msprime (Baumdicker et al., 2022; Anderson-Trocmé et al., 2023) and controlled family graph templates motivated by known non-identifiability results (Donnelly, 1983; Kirkpatrick, 2012).

Family graph templates. For each example, we first choose a two-person case with probability 0.35 or a three-person case with probability 0.65. We then sample from the generic families with probability 0.85 or from the non-identifiable families with probability 0.15. We use *generic* for examples sampled without explicitly imposing one of the non-identifiable constructions described below. The term does not imply that every generic family graph is statistically identifiable. The non-identifiable families are the specific ambiguity constructions described below. Figure 3 illustrates representative templates.

For a generic two-person case, the pair is unrelated with probability 0.40. Otherwise, it has one or two shared ancestors with probabilities 0.55 and 0.45, respectively. The total branch depth is 2, 3, or 4 edges with probabilities 0.80, 0.15, and 0.05, and this total is split between the two descendant branches. The total branch depth is assigned uniformly across the two observed individuals: for a total depth D, we sample (d_a, d_b) uniformly from the assignments with $d_a + d_b = D$ and $d_a, d_b \geq 1$, where d_a and d_b are the numbers of edges from the shared ancestor to observed individuals a and b. Generic three-person examples are drawn uniformly from five templates: full siblings plus an unrelated individual, half siblings plus an unrelated individual, three full siblings, full cousins plus an unrelated individual, and a sibling pair plus a full cousin.

The non-identifiable two-person case contains family graphs with one shared ancestor and total branch depth four. The realized member uses one of the three assignments $(1, 3)$, $(2, 2)$, or $(3, 1)$. These structures have the same total number of edges connecting the two observed individuals through the shared ancestor, giving the cousin-type non-identifiability described by Donnelly (1983). For distributional evaluation, the target law is uniform over the three corresponding graph isomorphism classes. The non-identifiable three-person family classes described by Kirkpatrick (2012) each have target probability $1/2$. All other examples have a point-mass target law at the realized family graph.

Genetic simulation. We use msprime's FixedPedigree simulator (Baumdicker et al., 2022; Anderson-Trocmé et al., 2023) in order to generate the synthetic data given each family graph (also known as pedigree graph). For each chromosome, we first simulate recombination through the family graph, then extend the founder ancestry using the Hudson coalescent with population size $2{,}000$. We use 22 chromosomes of 15 Mb each, recombination rate 10^{-8} per base per generation, and mutation rate 2×10^{-8} per base with the msprime binary mutation model. Only the two or three designated observed individuals are retained as genetic observations. Their two haplotypes are summed to obtain unphased genotype dosages in $\{0, 1, 2\}$. An example is resimulated if fewer than 30 segregating sites are produced.

For the reported experiments, in both Any2Graph and PSGP, each SNP is represented by a categorical token encoding the joint dosage pattern across at most three observed individuals. Tokens remain in genomic order. When more than $2{,}047$ SNPs are present, sites are evenly subsampled to $2{,}047$ positions and a start token is prepended.

Prediction target. The SGP target contains the observed individuals, the lowest common ancestors of observed nodes, and the intermediate ancestors on paths connecting those common ancestors to the observations. Simulation-only side parents that are not part of the target relationship skeleton are removed.

Target edges retain their parent-to-child orientation, therefore the graphs are directed. Node labels are encoded as 0 for an unobserved ancestor, 1 for observed individual a, 2 for observed individual b, and 3 for observed individual c. Thus all unobserved ancestors share one node category, while the observed individuals have distinct categories. Graph isomorphism may therefore reindex unobserved ancestors but must preserve the identities of the observed individuals. For two-person examples, node category 3 is absent. Target graphs contain at most eight nodes.

We generate $100{,}000$ training, $10{,}000$ validation, and $10{,}000$ test examples using independent seeds.

C Implementation and experimental details

This section describes the pSGP architecture and the training and sampling configurations. The implementation builds on the permutation-equivariant graph-transformer design of DiGress (Vignac et al., 2023) and its discrete flow matching adaptation in DeFoG (Qin et al., 2025a). Directed family graph outputs use the directed extension introduced in DIRECTO (Carballo-Castro et al., 2026). Figure 1 shows an overview of the architecture of pSGP.

C.1 Graph denoiser

Our denoiser retains the node–edge–global graph-transformer backbone of DiGress (Vignac et al., 2023) and its flow matching adaptation in DeFoG (Qin et al., 2025a). For directed outputs, we use the directed extension of DIRECTO (Carballo-Castro et al., 2026). We do not use any of the structural encodings such as RRWP or MagLap used in these papers. We extend their backbones with conditioning mechanisms for the arbitrary non-graph input x required by SGP. A modality-specific encoder produces a sequence of condition tokens

$$C(x) \in \mathbb{R}^{m \times d_c}$$

and a pooled condition representation

$$c(x) \in \mathbb{R}^{d_c},$$

where m depends on the input modality. The former provides token-level conditioning through cross-attention, while the latter provides global conditioning together with the flow time t.

For an intermediate graph $G_t = (V_t, E_t)$, input projections initialize hidden node and edge states

$$H_X^{(0)} \in \mathbb{R}^{n \times d_X}, \qquad H_E^{(0)} \in \mathbb{R}^{n \times n \times d_E},$$

and the pooled condition and flow time initialize the graph-level state,

$$y^{(0)} = \phi_y([c(x), t]) \in \mathbb{R}^{d_y}.$$

Thus $c(x)$ and t initialize the global stream once per denoiser evaluation, after which $y^{(\ell)}$ evolves through the L graph-transformer layers.

At every layer, the current node representations first cross-attend to the condition tokens $C(x)$, with graph nodes as queries and condition tokens as keys and values. This node-to-condition cross-attention is our conditioning extension to the DiGress/DeFoG graph-transformer backbone and allows different graph nodes to retrieve different information from the supervised input.

The resulting node states then enter the graph self-attention update together with the current edge and graph-level states. Following DiGress (Vignac et al., 2023) and DeFoG (Qin et al., 2025a), the edge representation $H_{E,ij}^{(\ell)}$ modulates the pairwise interaction between nodes i and j through learned multiplicative and additive terms. The graph-level state similarly provides feature-wise affine modulation of the resulting node and edge updates. If z denotes an unmodulated node or edge update, this operation has the form

$$z \longmapsto \beta\Big(y^{(\ell)}\Big) + \Big(1 + \gamma\Big(y^{(\ell)}\Big)\Big) \odot z,$$

where γ and β are learned projections from the current graph-level state to feature-wise scale and shift vectors, $\odot$ denotes element-wise multiplication, and separate projections are used for the node and edge streams. The graph-level state is itself updated from the current node and edge representations.

Separate residual feed-forward blocks and layer normalization are then applied to the node, edge, and graph-level streams, yielding

$$\left(H_X^{(\ell+1)}, H_E^{(\ell+1)}, y^{(\ell+1)}\right).$$

Variable-size graphs are padded for batching, with masks preventing padded node and node-pair positions from contributing to the computation.

After the final layer, separate node and edge heads produce clean endpoint predictions. Categorical heads output logits over the clean endpoint categories, while continuous heads directly predict the corresponding clean continuous values. We additionally condition the categorical edge prediction head on the hidden representations of both endpoint nodes,

$$\left[H_{E,ij}^{(L)}, H_{X,i}^{(L)}, H_{X,j}^{(L)}\right].$$

The node representations are detached before this concatenation, so the edge-head loss does not backpropagate through this additional node-to-edge path. Optionally, we further condition the edge head directly on $C(x)$ through pairwise cross-attention in which each candidate node pair queries the condition tokens. For undirected outputs, we retain the final edge-logit symmetrization of the DiGress/DeFoG backbone,

$$\mathbf{z}_{ij} = \frac{1}{2}\left(\widetilde{\mathbf{z}}_{ij} + \widetilde{\mathbf{z}}_{ji}\right),$$

where $\widetilde{\mathbf{z}}_{ij}$ denotes the edge logits before symmetrization. This preserves the symmetric edge predictions required for undirected graphs despite the ordered endpoint representation used by the edge head.

For directed outputs, this final symmetrization is omitted and the undirected graph self-attention is replaced by the directed graph transformer of DIRECTO (Carballo-Castro et al., 2026). In particular, $H_{E,ij}^{(\ell)}$ and $H_{E,ji}^{(\ell)}$ remain distinct throughout the network, allowing the denoiser to represent parent-to-child orientation. The external conditioning interface and clean endpoint prediction heads otherwise follow the same structure described above.

The same denoiser interface supports categorical and continuous graph outputs; only the endpoint heads and corresponding flow dynamics differ. Table 4 summarizes the architecture and conditioning configurations. The hidden dimensions give the node, edge, and graph-level widths (d_X, d_E, d_y).

Table 4: **pSGP architecture and conditioning configurations.** "Edge ctx." indicates the additional optional pairwise edge-to-condition cross-attention block.

Task	Layers	(d_X, d_E, d_y)	Cond. dim.	Condition encoder	Edge ctx.
Coloring 10	5	$(256, 64, 64)$	128	8 residual blocks, patch 2	yes
Coloring 15	5	$(256, 64, 64)$	128	8 residual blocks, patch 8	yes
QM9	5	$(256, 64, 64)$	128	3-layer, 4-head token transformer	yes
GDB13	5	$(512, 128, 128)$	512	5-layer, 8-head token transformer	yes
Toulouse	4	$(256, 64, 64)$	256	truncated ResNet-18	no
USCities	4	$(256, 64, 64)$	256	truncated ResNet-18	no
Genotypes	5	$(256, 64, 64)$	256	4-layer, 8-head token transformer	yes

C.2 Condition encoders and graph size prediction

The condition encoder provides the two representations used by the denoiser, that is, the token sequence $C(x)$ used for node-to-condition cross-attention and the pooled representation $c(x)$ used to initialize the global conditioning stream. The encoder architecture is task-dependent, but this interface is shared across modalities.

Image conditions. The Coloring models use a stride-one convolutional residual encoder (He et al., 2016). A 3×3 convolution first maps the RGB input to the condition dimension, followed by eight residual convolutional blocks with GroupNorm (Wu & He, 2018) and ReLU activations. Coloring 10 then uses a stride-2 patch projection, producing a 16×16 token grid from the 32×32 input. Coloring 15 uses an 8×8 patch projection, producing an 8×8 token grid from the 64×64 input. Learned row and column positional embeddings are added before the spatial features are flattened to form $C(x)$, and spatial mean pooling produces the global representation $c(x)$.

The road tasks use a truncated ResNet-18 (He et al., 2016), following the image conditioning setup used for the Any2Graph benchmarks (Krzakala et al., 2024). We remove the initial max-pooling layer and retain the convolutional stem and the first two ResNet stages, giving an overall spatial stride of four. The resulting feature map is projected to dimension 256 and augmented with learned two-dimensional positional embeddings. Thus the 64×64 Toulouse inputs produce a 16×16 token grid, while the 128×128 USCities inputs produce a 32×32 grid. The flattened spatial representations form $C(x)$, and their spatial mean provides $c(x)$.

Token conditions. QM9 and GDB13 use the same standard Transformer encoder over input tokens (Vaswani et al., 2017) used in Any2Graph (Krzakala et al., 2024). Genotypes uses the same encoder family. Each token receives a learned token embedding and learned positional embedding, followed by pre-normalized Transformer layers (Xiong et al., 2020) with GELU feed-forward blocks (Hendrycks & Gimpel, 2016). The output token sequence forms $C(x)$, while the masked mean of the non-padding token representations produces $c(x)$.

Graph size. Standard graph generators may draw the number of nodes from an unconditional dataset-level size distribution. In SGP, however, the output size is itself a property to be predicted directly from the input. We therefore model it separately via the conditional distribution $p_\psi(n \mid x)$.

For each task, the graph size predictor uses the same encoder architecture as the corresponding condition encoder followed by a categorical classification head. It is pretrained with cross-entropy against the observed target size, and its parameters are frozen during graph flow training. During graph flow training, the observed target size n determines the dimensions and valid positions of G_t. The output of the size predictor is not supplied to the graph denoiser as an additional scalar feature.

At inference, the size selection rule depends on the prediction setting. For the "single" predictions, we use the most probable graph size, $\hat{n}(x) \in \arg\max_n p_\psi(n \mid x)$, and generate one graph conditional on $(\hat{n}(x), x)$ using the finite sampler.

For modal decoding, each of the K candidates independently draws its size from $p_\psi(\cdot \mid x)$ and uses independent graph-generation randomness conditional on that size and the input. The candidates are therefore IID conditional on x, retaining uncertainty over graph size as part of the predictive distribution. The Genotypes distributional evaluation uses the same sampling rule with $K = 16$, but retains all generated graphs without applying modal decoding.

C.3 Training objective and optimization

Categorical features are trained with cross-entropy and continuous ones with squared error, as in Eq. 8. For the fully categorical tasks, the node cross-entropy has unit weight and the edge cross-entropy is multiplied by 4 on Coloring and Genotypes and by 5 on QM9 and GDB13. All configurations additionally use binary edge category weights

$$(\alpha_{\text{no-edge}}, \alpha_{\text{edge}}) = (0.2, 1.0).$$

As described in Appendix E.2, these category weights are removed from the logits before the edge probabilities are used in the sampling rates. For Toulouse and USCities, the mixed loss uses coordinate MSE weight 5 and edge cross-entropy weight 1.

All models are optimized with AdamW (Loshchilov & Hutter, 2019) with weight decay 10^{-12}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. We use a cosine learning rate schedule with linear warmup and train with bfloat16 precision. Flow times used for training are sampled from $\mathrm{Unif}(0, 1)$.

For each deterministic task, we select the checkpoint with the highest GI-Acc on the validation set, while for Genotypes, we use validation MMD^2. We evaluate validation GI-Acc every 50 epochs, except for GDB13 and Genotypes, for which validation is performed every 20 epochs.

Table 5 gives the optimization settings and selected checkpoint epochs. Hyperparameters were chosen based on performance on the validation sets and recommended defaults for DeFoG (Qin et al., 2025a) and DIRECTO (Carballo-Castro et al., 2026).

Table 5: **Training configurations used for the reported PSGP results.** "Epochs" is the configured training duration and "Ckpt." is the checkpoint epoch used for the paper result.

Task	Epochs	Ckpt.	Batch	Learning rate	Warmup
Coloring 10	500	350	512	4×10^{-4}	1000
Coloring 15	250	200	512	4×10^{-4}	1000
QM9	650	650	512	3×10^{-4}	8000
GDB13	100	80	512	1×10^{-4}	8000
Toulouse	650	650	256	2×10^{-4}	8000
USCities	650	650	256	2×10^{-4}	8000
Genotypes	100	20	128	3×10^{-4}	1000

C.4 SAMPLING CONFIGURATIONS

Table 6 summarizes the sampler settings used for the reported checkpoints. All experiments use the finite sampler of Appendix E and deterministic categorical argmax on the endpoint prediction step.

Table 6: **Sampling configurations used for the reported results.** The polynomially decelerating schedule is $f(t) = 2t - t^2$ as in DeFoG (Qin et al., 2025a).

Task	Steps S	η	Time schedule
Coloring 10	250	100	polynomially decelerating
Coloring 15	250	100	polynomially decelerating
QM9	1000	0	identity
GDB13	1000	0	identity
Toulouse	1000	0	identity
USCities	1000	0	identity
Genotypes	250	0	identity

Sample-based modal decoding. The underlying graph flow produces samples from the conditional model rather than a canonical point prediction. For SGP evaluations that require one graph output, we therefore use the sample-based modal decoder introduced in Section 2.4. This decoding procedure is specific to PSGP's point-prediction setting and is applied only after complete graph samples have been generated; it does not modify the underlying DeFoG/DIRECTO sampling dynamics.

For the deterministic benchmarks, the "single" prediction uses one graph generated conditional on the most probable size under $p_\psi(\cdot \mid x)$. The "modal@32" prediction instead generates $K = 32$ candidates and groups them using attribute-aware Weisfeiler–Lehman (WL) hashes (Shervashidze et al., 2011). Candidates with identical hashes are assigned to the same cluster, and the earliest generated candidate in the largest cluster is returned. The resulting cluster frequencies provide an empirical approximation to the distribution over graph isomorphism classes, and the largest cluster approximates its empirical mode without requiring pairwise exact graph-isomorphism tests. WL hashes are computed with NetworkX (Hagberg et al., 2008) using 3 iterations.

Categorical node and edge attributes are included in the WL hashes. For graphs with continuous attributes, continuous values are quantized only when constructing the hashes used for clustering, but the returned graph maintains its original continuous values. Because WL hashes are not complete graph-isomorphism invariants, non-isomorphic graphs can in principle receive the same hash. Thus modal@32 approximates rather than exactly computes the empirical modal isomorphism class.

Distributional sampling. The Genotypes distributional evaluation does not apply modal decoding. Instead, it retains $K = 16$ independently generated graphs per condition as samples from the model's finite sampling procedure, including independent graph size draws and generation randomness. These samples are used directly to evaluate the predictive distribution as described in Appendix D.

D EVALUATION DETAILS

We evaluate predictions at the level of attributed graphs rather than their indexed tensor representations. For the established SGP benchmarks, our primary metric is graph-isomorphism accuracy (GI-Acc), following the evaluation protocol used by Any2Graph (Krzakala et al., 2024) and GRALE (Krzakala et al., 2025). For Genotypes, where several non-isomorphic outputs may be simultaneously valid, we instead compare the predicted and target distributions over graph isomorphism classes.

D.1 GRAPH-ISOMORPHISM ACCURACY

Categorical graphs. For Coloring 10, Coloring 15, QM9, and GDB13, a prediction is counted as correct if it is exactly isomorphic to the target as an attributed graph. Equivalently, for a predicted graph $\widehat{g}$ and target g, define the per-example graph-isomorphism indicator

$$\mathrm{GI}(\widehat{g}, g) = \mathbf{1}\{[\widehat{g}] = [g]\}. \tag{11}$$

Thus the predicted and target graphs must have the same number of nodes, and there must exist a node permutation that simultaneously matches all nodes and edges, as well as their features.

For an evaluation set $(\widehat{g}_i, g_i)_{i=1}^N$, graph-isomorphism accuracy is the empirical mean of this indicator,

$$\text{GI-Acc} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{GI}(\widehat{g}_i, g_i). \tag{12}$$

Equivalently, on these categorical benchmarks, GI-Acc is one minus the empirical graph-level zero–one risk defined in Section 2.1. Exact graph isomorphism is used for scoring both the single and modal@32 predictions. In particular, the WL hashes used by the modal decoder are used only to choose a prediction from the generated candidate pool, and the selected graph is subsequently evaluated by exact attributed graph isomorphism.

Extension to graphs with continuous features. For Toulouse and USCities, the node attributes are continuous two-dimensional coordinates, so exact equality of attributed graphs is not appropriate. Following the benchmark evaluation, we use a tolerance-relaxed graph-isomorphism criterion. Let $\widehat{x}_i \in \mathbb{R}^2$ and $x_i \in \mathbb{R}^2$ denote predicted and target node coordinates in the normalized coordinate system used by the dataset. A prediction is correct if the predicted and target graphs have the same number of nodes and there exists a graph isomorphism π such that

$$\left\|\widehat{x}_i - x_{\pi(i)}\right\|_2 < \delta \qquad \text{for every predicted node } i. \tag{13}$$

The GI-Acc is the fraction of test examples satisfying this criterion.

D.2 DISTRIBUTIONAL EVALUATION FOR GENOTYPES

GI-Acc against a single realized target is not an appropriate distributional metric for Genotypes because some genetic conditions admit several non-isomorphic valid family graphs. We therefore evaluate the predictive distribution using squared maximum mean discrepancy (MMD^2) (Gretton et al., 2012) with the equality kernel on the collapsed class space defined in Section 3.3 and Eq. 10.

For each condition, let the finite benchmark target distribution have support $\{u_1, \ldots, u_M\}$ with probabilities

$$r_m := r(u_m \mid x), \qquad \sum_{m=1}^{M} r_m = 1.$$

For generic examples, $M = 1$ and $r_1 = 1$. For the non-identifiable families, the support and probabilities are those defined in Appendix B.

A generated family graph is assigned to target class u_m if it is isomorphic to that target as a directed attributed graph. Edge directions must be preserved. The observed individuals have distinct node labels, so an isomorphism must preserve their identities, whereas all unobserved ancestors share the same node label and may be permuted. Any generated graph that is not isomorphic to one of the valid target classes is mapped to a single additional invalid class u_{inv} with target probability zero.

Collapsing all off-family outputs into this shared class prevents a model from receiving a smaller discrepancy merely by spreading invalid probability mass across many different invalid graphs.

For PSGP, let $U_1, \ldots, U_K$ be $K = 16$ IID generated graphs for one condition. Each candidate is generated independently from the model's finite sampling procedure, including an independent graph size draw from the size distribution used for the Genotypes evaluation and independent graph generation randomness. No deterministic size or modal decoding are used for this distributional evaluation. Let N_m be the number of samples assigned to valid class u_m and define

$$N_{\text{inv}} = K - \sum_{m=1}^{M} N_m.$$

Because the generated graphs are IID and the target probabilities are known exactly, we evaluate the target-dependent terms of MMD analytically and use the unbiased U-statistic for the generated-sample term (Gretton et al., 2012). The resulting per-condition estimator is

$$\widehat{\text{MMD}}^2_{\text{GI}} = \frac{\sum_{m=1}^{M} N_m(N_m - 1) + N_{\text{inv}}(N_{\text{inv}} - 1)}{K(K-1)} - \frac{2}{K}\sum_{m=1}^{M} N_m r_m + \sum_{m=1}^{M} (r_m)^2. \tag{14}$$

Indeed, if $p_{\theta,\psi,m}$ denotes the probability that a generated graph falls in class u_m, then IID sampling gives

$$\mathbb{E}\left[\frac{N_m(N_m-1)}{K(K-1)}\right] = p^2_{\theta,\psi,m},$$

with the analogous identity for the invalid class. Therefore, Eq. 14 is an unbiased estimator of Eq. 10 for the distribution induced by the finite sampling procedure after invalid outputs are collapsed as described above. With the equality kernel, the corresponding population quantity is the squared Euclidean distance between the predicted and target class probability vectors on this collapsed class space.

Any2Graph (Krzakala et al., 2024) produces one deterministic prediction per condition. We interpret this prediction as a point-mass predictive distribution rather than applying the K-sample estimator. If its prediction belongs to valid class u_j, its exact distributional discrepancy is

$$\text{MMD}^2_{\text{GI}} = 1 - 2r_j + \sum_{m=1}^{M} r_m^2. \tag{15}$$

If the prediction is outside the valid target family, we set $r_j = 0$, corresponding to a point mass on u_{inv}.

We compute Eq. 14 independently for each test condition and report its mean over the full test set and separately over the generic and non-identifiable subsets. Lower MMD^2 indicates closer agreement between the predicted and target distributions, with a population value of zero exactly when the two collapsed class distributions agree.

E Flow matching and sampling details

This section gives the flow matching and finite-step sampling specification used by PSGP. We distinguish the exact rates and velocity fields that define the construction from the finite numerical updates used in the experiments.

E.1 Source distributions and conditional paths

For categorical graph variables, all experiments use the empirical marginal source distribution. For each node- or edge-feature channel, the source probabilities are the corresponding empirical category frequencies in the training data, shared across all node or node-pair positions of that channel. This is the marginal noise construction used in permutation-equivariant graph diffusion and flow models such as DiGress (Vignac et al., 2023) and DeFoG (Qin et al., 2025a). Source variables are sampled independently conditional on the graph size. For undirected edges, one source value is sampled for each unordered node pair and mirrored across the adjacency tensor.

Given a categorical clean endpoint $G_1^a = k$, the conditional path is the linear discrete flow matching path

$$q_t^a(j \mid k) = (1-t)q_0^a(j) + t\mathbf{1}\{j = k\}, \qquad 0 \le t \le 1, \tag{16}$$

following discrete flow matching (Campbell et al., 2024) and its graph adaptation in DeFoG (Qin et al., 2025a). In implementation, this path is sampled directly: independently for each categorical graph variable, the intermediate value is set to its clean value with probability t and to an independent source value with probability $1-t$. A single $t \sim \mathrm{Unif}(0,1)$ is shared across the graph variables of a training example.

To support continuous attributes, such as node coordinates in the road graph tasks, PSGP extends discrete graph flow matching with a continuous flow matching path. Specifically, for a continuous graph variable b, we define the source distribution and conditional path as

$$G_0^b \sim \mathcal{N}(0, I_{d_b}), \qquad G_t^b = (1-t)G_0^b + tG_1^b, \tag{17}$$

following continuous flow matching (Lipman et al., 2023) and the mixed discrete–continuous construction of Campbell et al. (2024). In the road graph experiments, we use this continuous path for node coordinates together with Eq. 16 for the binary edge variables.

E.2 Endpoint prediction and class weight correction

The denoiser receives the current graph G_t, time t, supervised condition x, and graph size n, and predicts the clean endpoint. Its categorical heads produce logits for the clean category of each graph variable, while its continuous heads directly predict the corresponding clean continuous value. Training uses the endpoint objective in Eq. 8.

The reported configurations assign category-dependent weights to the binary edge cross-entropy, with no-edge and edge weights 0.2 and 1.0, respectively. Because category-dependent weights tilt the Bayes-optimal softmax probabilities, we undo this tilt before the edge prediction is used as a clean endpoint posterior in the sampling rates. If $z_\theta^a(k)$ is the logit for category k and $\alpha_a(k) > 0$ is its training weight, the probability used for sampling is

$$\widehat{p}_{\theta,\mathrm{samp}}^a(k) = \mathrm{softmax}_k\left(z_\theta^a(k) - \log \alpha_a(k)\right). \tag{18}$$

For unweighted categorical heads, this reduces to the ordinary softmax. The correction is applied only when converting endpoint logits into probabilities for sampling, while training retains the specified weighted cross-entropy.

E.3 Categorical sampling dynamics

For a categorical graph variable a with category set $\mathcal{K}_a$, PSGP uses the endpoint-conditioned discrete flow matching rates of Campbell et al. (2024), applied to graph variables as in DeFoG (Qin et al., 2025a). For $0 < t < 1$, define

$$\mathcal{K}_{t,+}^a(k) = \{i \in \mathcal{K}_a : q_t^a(i \mid k) > 0\}, \qquad Z_t^a(k) = |\mathcal{K}_{t,+}^a(k)|.$$

Writing $[z]_+ = \max(z, 0)$, the linear path in Eq. 16 has derivative

$$\dot{q}_t^a(i \mid k) = \mathbf{1}\{i = k\} - q_0^a(i).$$

For $i \neq j$, the base conditional rate is

$$R_t^{a,*}(i, j \mid k) = \begin{cases} \dfrac{[\dot{q}_t^a(j \mid k) - \dot{q}_t^a(i \mid k)]_+}{Z_t^a(k) q_t^a(i \mid k)}, & i, j \in \mathcal{K}_{t,+}^a(k), \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

The support restriction avoids division by zero. At $t = 0$, the base rates are defined by their right-hand limits. When q_0^a has full support, these are obtained by direct substitution of $t = 0$.

We additionally allow the detailed balance term

$$R_t^{a,\mathrm{DB}}(i, j \mid k) = \eta q_t^a(j \mid k), \qquad i \neq j, \qquad \eta \ge 0, \tag{20}$$

and use

$$R_t^a = R_t^{a,*} + R_t^{a,\mathrm{DB}}.$$

The diagonal entries of each rate matrix are the negative sums of its off-diagonal row entries. At each fixed time,

$$q_t^a(i \mid k) R_t^{a,\mathrm{DB}}(i, j \mid k) = \eta q_t^a(i \mid k) q_t^a(j \mid k) = q_t^a(j \mid k) R_t^{a,\mathrm{DB}}(j, i \mid k).$$

Thus the detailed balance term contributes zero net probability flux under $q_t^a(\cdot \mid k)$ and does not change the prescribed endpoint-conditioned probability path.

At generation time the clean endpoint is unknown. The model therefore averages the endpoint-conditioned rates using the corrected endpoint probability estimates from Appendix E.2:

$$R_{t,\theta}^a(h, j \mid x, n) = \sum_{k \in \mathcal{K}_a} \widehat{p}_{\theta,\mathrm{samp}}^a(k \mid h, t, x, n) R_t^a(h^a, j \mid k), \qquad j \neq h^a. \tag{21}$$

The experimental sampler discretizes time using S steps. Let

$$t_m = f\left(\frac{m}{S}\right), \qquad s_m = f\left(\frac{m+1}{S}\right), \qquad m = 0, \ldots, S-1, \tag{22}$$

where $f(t) = t$ for the identity schedule and $f(t) = 2t - t^2$ for the polynomially decelerating schedule. Suppose graph variable a currently has category $i = h^a$ at time t_m, and write $\Delta_m = s_m - t_m$. For a nonterminal step, the implementation forms the nonnegative transition weights

$$W_m^a(j \mid h) = \begin{cases} \Delta_m R_{t_m,\theta}^a(h, j \mid x, n), & j \neq i, \\ \left[1 - \sum_{\ell \neq i} \Delta_m R_{t_m,\theta}^a(h, \ell \mid x, n)\right]_+, & j = i. \end{cases} \tag{23}$$

The next category is sampled from these weights after normalization. For undirected edges, only one value is retained for each unordered pair and the result is mirrored. Before the finite update, non-finite rates and rates above 10^5 are set to zero as a numerical stabilization.

On the final step, the implementation does not take another finite rate step. It instead uses the denoiser's clean endpoint probabilities directly. All configurations use deterministic argmax for this final categorical update.

E.4 Continuous and mixed sampling dynamics

For a continuous graph variable b and $t < 1$, differentiating the straight path in Eq. 17 gives the endpoint-conditioned velocity

$$v_t^b(h \mid g) = \frac{g^b - h^b}{1 - t}.$$

Using the denoiser's clean endpoint prediction gives the learned velocity

$$v_{t,\theta}^b(h \mid x, n) = \frac{\widehat{G}_{1,\theta}^b(h, t, x, n) - h^b}{1 - t}, \tag{24}$$

For a nonterminal finite step from t_m to s_m, the continuous flow matching sampler updates each continuous attribute according to

$$(G_{s_m}^b)_{\mathrm{num}} = G_{t_m}^b + \frac{s_m - t_m}{\max(1 - t_m, 10^{-4})} \left(\widehat{G}_{1,\theta}^b - G_{t_m}^b\right). \tag{25}$$

At the final step, the continuous state is replaced directly by $\widehat{G}_{1,\theta}^b$.

In the road graph experiments, generated coordinates are constrained to the coordinate range used by each benchmark: $[-1, 1]$ for Toulouse and $[0, 1]$ for USCities.

F PROOF OF REPRESENTATIVE INVARIANCE

We use the graph representation and permutation action from Section 2.1, and the graph variables and conditional paths from Section 2.2. The supervised input x is held fixed when the output graph is permuted. We write $q_0(\cdot \mid n)$ for the source distribution and $q_t(\cdot \mid g, n)$ for the conditional law of G_t given the clean target g and graph size n.

For $a \in \mathcal{A}_{\mathrm{cat}}$, let $\mathcal{K}_a$ be the finite category set of G^a. For $b \in \mathcal{A}_{\mathrm{cont}}$, write $G^b \in \mathbb{R}^{d_b}$. Each index specifies a feature channel of a node or node pair, and both index sets are finite for fixed n. A node permutation changes the associated node or node pair but not the feature channel. We denote the resulting indices by πa and πb. Consequently,

$$\mathcal{K}_{\pi a} = \mathcal{K}_a, \qquad d_{\pi b} = d_b, \tag{26}$$

and the feature values satisfy

$$(\pi \cdot g)^{\pi a} = g^a, \qquad (\pi \cdot g)^{\pi b} = g^b. \tag{27}$$

Proposition 2 (Representative invariance; restated). *Suppose the conditional probability path and denoiser are permutation equivariant, and the endpoint loss is invariant when the same node permutation is applied to both its prediction and target. Then, for every parameter setting θ, input $x \in \mathcal{X}$, graph $g \in \mathcal{G}_n$, and permutation $\pi \in \mathfrak{S}_n$,*

$$\mathcal{R}_\theta(x, \pi \cdot g) = \mathcal{R}_\theta(x, g). \tag{28}$$

The loss-invariance argument uses the same reindexing principle as DiGress (Vignac et al., 2023). DeFoG establishes analogous permutation-invariance properties for discrete graph flow matching (Qin et al., 2025a). The result here concerns the supervised setting: for a fixed external input x, it shows that the conditional training risk is unchanged by the indexed representative chosen for the target graph. We also allow continuous and mixed graph features.

Proof. We first give the argument when all graph features are categorical, so $\mathcal{G}_n$ is finite.

Let $D_\theta(h, t, x, n)$ denote the full clean endpoint prediction of the denoiser when the intermediate graph is h. Let $L(D_\theta(h, t, x, n), g)$ denote the endpoint loss against the clean target g, and define the corresponding per-sample training loss by

$$\ell_\theta(h, t, x, n; g) = L(D_\theta(h, t, x, n), g). \tag{29}$$

Permutation equivariance of the denoiser means that permuting the intermediate graph permutes its prediction in the same way:

$$D_\theta(\pi \cdot h, t, x, n) = \pi \cdot D_\theta(h, t, x, n). \tag{30}$$

The supervised input x is held fixed under this permutation.

By assumption, applying the same node permutation to the prediction and target does not change the endpoint loss:

$$L(\pi \cdot y, \pi \cdot g) = L(y, g) \tag{31}$$

for any prediction y. Combining Eqs. 30 and 31 gives

$$\ell_\theta(\pi \cdot h, t, x, n; \pi \cdot g) = \ell_\theta(h, t, x, n; g). \tag{32}$$

For a clean target g, its expected training loss is

$$\mathcal{R}_\theta(x, g) = \mathbb{E}_t \left[\sum_{h \in \mathcal{G}_n} q_t(h \mid g, n) \ell_\theta(h, t, x, n; g) \right]. \tag{33}$$

Permutation equivariance of the conditional probability path means

$$q_t(\pi \cdot h \mid \pi \cdot g, n) = q_t(h \mid g, n). \tag{34}$$

Because $h \mapsto \pi \cdot h$ is a bijection of $\mathcal{G}_n$, we may reindex the sum in Eq. 33 to obtain

$$\begin{aligned}\mathcal{R}_\theta(x, \pi \cdot g) &= \mathbb{E}_t \left[\sum_{h \in \mathcal{G}_n} q_t(\pi \cdot h \mid \pi \cdot g, n) \ell_\theta(\pi \cdot h, t, x, n; \pi \cdot g) \right] \\ &= \mathbb{E}_t \left[\sum_{h \in \mathcal{G}_n} q_t(h \mid g, n) \ell_\theta(h, t, x, n; g) \right] \\ &= \mathcal{R}_\theta(x, g), \end{aligned} \tag{35}$$

where the second equality uses Eqs. 34 and 32.

For continuous or mixed graph features, the same argument can be expressed directly in terms of random variables: draw t according to the training-time distribution and then draw $G_t \sim q_t(\cdot \mid g, n)$. Permutation equivariance of the path implies that, conditional on the same t,

$$\pi \cdot G_t \sim q_t(\cdot \mid \pi \cdot g, n). \tag{36}$$

Therefore,

$$\begin{aligned}\mathcal{R}_\theta(x, \pi \cdot g) &= \mathbb{E}\left[\ell_\theta(\pi \cdot G_t, t, x, n; \pi \cdot g)\right] \\ &= \mathbb{E}\left[\ell_\theta(G_t, t, x, n; g)\right] \\ &= \mathcal{R}_\theta(x, g), \end{aligned} \tag{37}$$

where the second equality follows from Eq. 32. Hence Eq. 28 holds for categorical, continuous, and mixed graph features. □

We next verify the assumptions of the proposition for the probability path, denoiser, and endpoint loss used by PSGP.

Equivariance of the probability path. For categorical node and edge features, PSGP uses a source distribution that is shared across graph variables belonging to the same feature channel. In the implementation used in our experiments, this is the empirical marginal distribution of that feature channel, following the marginal noise construction used by DiGress (Vignac et al., 2023) and the default marginal initial distribution of DeFoG (Qin et al., 2025a). Thus, if a and πa denote variables related by a node permutation,

$$q_0^{\pi a} = q_0^a. \tag{38}$$

Since the source variables are sampled independently using distributions shared across permuted node or node-pair positions, relabeling the nodes merely reindexes the sampled variables. Consequently, the categorical source graph distribution is permutation invariant.

Given a clean categorical value $G_1^a = k$, PSGP uses the linear conditional path

$$q_t^a(j \mid k) = (1 - t) q_0^a(j) + t \mathbf{1}\{j = k\}, \tag{39}$$

following discrete flow matching (Campbell et al., 2024) and its graph adaptation in DeFoG (Qin et al., 2025a). Conditional on the clean graph, the categorical path factorizes over graph variables:

$$q_t(h \mid g, n) = \prod_{a \in \mathcal{A}_{\text{cat}}} q_t^a(h^a \mid g^a). \tag{40}$$

Using Eq. 27, $q_0^{\pi a} = q_0^a$, and reindexing the product by $a \mapsto \pi a$ gives

$$q_t(\pi \cdot h \mid \pi \cdot g, n) = q_t(h \mid g, n), \tag{41}$$

which is Eq. 34.

For a continuous graph variable b, PSGP independently samples

$$G_0^b \sim \mathcal{N}(0, I_{d_b}), \tag{42}$$

and uses the straight conditional path

$$G_t^b = (1 - t) G_0^b + t G_1^b. \tag{43}$$

following standard continuous flow matching (Lipman et al., 2023). Variables related by a node permutation have the same dimension by Eq. 26, and therefore the same Gaussian source distribution. Since the source variables are sampled independently, a node permutation merely reindexes the factors of the joint source distribution, so the continuous source law is permutation invariant. Applying the same permutation to G_0 and G_1 then reindexes the linear interpolation in the same way. Hence the continuous path is permutation equivariant. Combining the categorical and continuous constructions gives Eq. 36 for mixed graphs, following the mixed discrete–continuous flow construction of Campbell et al. (2024).

Equivariance of the denoiser. For undirected outputs, PSGP uses the permutation-equivariant graph transformer of DeFoG (Qin et al., 2025a), and for directed outputs, it uses the directed graph architecture of DIRECTO (Carballo-Castro et al., 2026). In both cases, the graph state is conditioned on representations of the supervised input x. These conditioning representations remain fixed when the output nodes are permuted.

For categorical predictions, denoiser equivariance means

$$\widehat{p}_\theta^{\pi a}\left(k \mid \pi \cdot h, t, x, n\right) = \widehat{p}_\theta^{a}\left(k \mid h, t, x, n\right) \tag{44}$$

for every $a \in \mathcal{A}_{\text{cat}}$ and $k \in \mathcal{K}_a$. For continuous predictions, it means

$$\widehat{G}_{1,\theta}^{\pi b}(\pi \cdot h, t, x, n) = \widehat{G}_{1,\theta}^{b}(h, t, x, n) \tag{45}$$

for every $b \in \mathcal{A}_{\text{cont}}$. These are the coordinate-wise forms of Eq. 30.

Invariance of the endpoint loss. The endpoint objective used by PSGP consists of coordinate-wise cross-entropy for categorical features and squared error for continuous features. For an intermediate graph h and clean target g, the corresponding per-sample loss is

$$\ell_\theta(h, t, x, n; g) = -\sum_{a \in \mathcal{A}_{\text{cat}}} w_a \alpha_a(g^a) \log \widehat{p}_\theta^{a}\left(g^a \mid h, t, x, n\right) + \sum_{b \in \mathcal{A}_{\text{cont}}} \lambda_b \left\| \widehat{G}_{1,\theta}^{b}(h, t, x, n) - g^b \right\|_2^2. \tag{46}$$

If one feature type is absent, its corresponding sum is omitted. The category weights $\alpha_a(k) > 0$ are those in Eq. 8, with $\alpha_a \equiv 1$ for unweighted heads.

The loss weights are shared across graph variables related by a node permutation:

$$w_{\pi a} = w_a, \qquad \lambda_{\pi b} = \lambda_b, \qquad \alpha_{\pi a}(k) = \alpha_a(k). \tag{47}$$

The last equality holds for every $k \in \mathcal{K}_a$.

By Eq. 27, the target feature values are reindexed consistently under the same node permutation. Together with Eqs. 44, 45, and 47, a node permutation therefore only reindexes the terms in the two sums of Eq. 46. Hence applying the same permutation to the denoiser prediction and clean target leaves the endpoint loss unchanged, as required by Eq. 31. Equivalently,

$$\ell_\theta(\pi \cdot h, t, x, n; \pi \cdot g) = \ell_\theta(h, t, x, n; g), \tag{48}$$

which is Eq. 32.

Thus the conditional probability path, denoiser, and endpoint loss used by PSGP satisfy the assumptions of Proposition 2. The result holds for every parameter value θ and does not require successful optimization. Finally, the permutation may be chosen separately for each training target. Therefore, replacing any target by another indexed representative of the same attributed graph leaves the population training objective unchanged. Note that this is an equality of expected losses, and that independently sampled intermediate states for two equivalent indexed targets need not produce identical individual Monte Carlo losses.