Generative Modeling of Hypergraphs using Latent Hyperbolic Geometry

Anonymous authors
Paper under double-blind review

Abstract

Hypergraphs encode multi-way interactions among entities arising in a broad range of real-world relational systems. Generative modeling of hypergraphs aims to capture patterns in these interactions for understanding and simulating complex relational structures. Existing approaches often face a difficult trade-off: parsimonious probabilistic models may oversimplify rich multi-way interactions, whereas expressive deep neural architectures can sacrifice interpretability and scalability. To strike a better balance between parsimony and expressivity, we introduce a novel generative model, the *Hyperbolic Hypergraph Generator*, which harnesses hyperbolic geometry as a source of expressive power while retaining a parsimonious and interpretable probabilistic latent space formulation. Theoretically, we establish identifiability of the latent geometry under a geometric compatibility condition, and characterize the gain in expressivity afforded by the hyperbolic latent space. For training and sampling, we develop H2G, a scalable procedure that combines likelihood-based embedding with latent space generator construction, and establish theoretical guarantees for the consistency of the generated hyperlink distribution. Extensive empirical comparisons with baseline methods show that H2G captures meaningful latent geometry and delivers strong generative performance, achieving leading results on fidelity metrics that measure the reproduction of key structural characteristics of real-world hypergraphs. Together, these theoretical and empirical results demonstrate that our model achieves an effective balance between parsimony and expressivity, while highlighting the broader potential of hyperbolic geometry for modeling complex multi-way relational systems.

1 Introduction

Hypergraphs encode multi-way interactions among entities in a broad range of real-world relational systems. Each hyperlink records a set of entities participating in a common interaction, such as symptoms co-occurring in a patient record (Johnson et al., 2016), proteins belonging to a functional complex (Lawson et al., 2024), or individuals linked through collaboration, co-voting, or co-citation (Benson et al., 2018; Kumar et al., 2020; Wang et al., 2026). Given an observed hypergraph, generative modeling aims to capture the underlying interaction patterns and generate realistic, previously unseen hyperlinks on the same node set (Wu et al., 2025; Ma et al., 2026), thereby supporting data augmentation and simulation while offering insight into the structure of complex relational systems.

This task presents several challenges. First, hyperlinks are discrete and combinatorial structures, with the number of possible node combinations growing exponentially with the number of nodes, making distribution learning challenging at scale. Moreover, an effective generator should capture special characteristics of hypergraphs, including varying hyperlink sizes, heterogeneous node popularity, and complex higher-order dependence. Overly restrictive formulations can miss important interaction patterns, whereas excessive model complexity can obscure the underlying generative mechanism and impose substantial computational and storage costs. These challenges call for a generative model that balances expressivity with parsimony and interpretability and supports scalable procedures for training and hyperlink generation.

Hyperbolic geometry offers such a source of expressive power beyond increasing latent dimension or neural complexity, while still leaving room for parsimonious modeling. Its negative curvature gives rise to exponential volume growth, providing substantial geometric capacity even in low dimensions

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

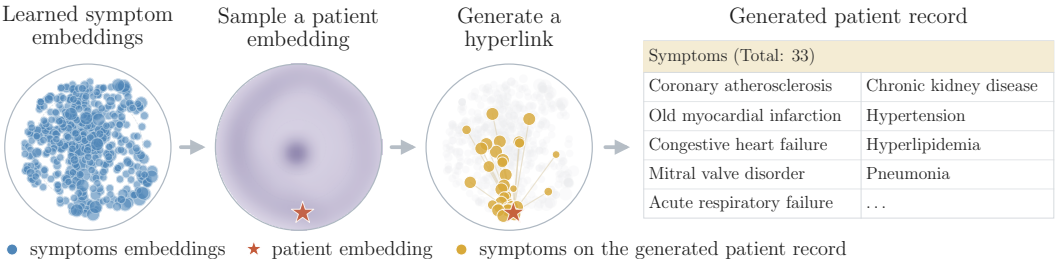

Figure 1: Illustration of the Hyperbolic Hypergraph Generator with a symptom–patient example based on MIMIC-III (Johnson et al., 2016). Given the learned symptom embeddings (blue dots), symptom popularity parameters (larger dots indicate higher popularity), and curvature of the hyperbolic space, we first sample a patient embedding from the estimated population distribution in that space and then generate a hyperlink corresponding to a synthetic patient record of co-occurring symptoms. Symptoms with embeddings closer to the patient embedding or with higher popularity are more likely to appear in the record. Symptom names are adapted from the official ICD-9 descriptions[1] for readability. Details of the MIMIC-III analysis are in Appendices D.2 and D.5.

and making it particularly well suited for capturing hierarchical, branching, and heterogeneous interaction patterns (Krioukov et al., 2010; Nickel & Kiela, 2017; 2018). This rich geometry allows complex latent structures to be encoded through simple distance-based relationships, motivating the use of a hyperbolic latent space to enhance the flexibility of a probabilistic generator without relying on an increasingly complex decoder.

In this work, we leverage hyperbolic geometry to address the challenges of hypergraph generative modeling and introduce the *Hyperbolic Hypergraph Generator*, a generative model built on a hyperbolic latent space with learnable curvature. Figure 1 illustrates the learned generative model through a symptom co-occurrence hypergraph example, where nodes correspond to symptom codes and each hyperlink represents a patient record containing a set of co-occurring symptoms. Given the learned symptom embeddings, symptom-specific popularity parameters, and hyperbolic curvature, the model first samples a patient embedding from the learned population distribution in the hyperbolic space and then generates a synthetic patient record, with symptoms closer to the patient embedding or with higher popularity being more likely to occur. For example, the generated record shown in Figure 1 represents a patient with heart conditions, including coronary disease and heart failure, accompanied by kidney disease and pneumonia. This distance-based mechanism naturally accommodates heterogeneous node frequencies and varying hyperlink sizes, while the shared latent hyperlink embedding induces higher-order dependence among co-occurring nodes. A formal definition and theoretical properties of the model are presented in Section 2, and the training and sampling procedures are described in Section 3.

1.1 Related work

Generative modeling of hypergraphs. Modeling hypergraphs has received growing attention in recent years. This includes planted-partition and degree-corrected block models for uniform hypergraphs (Ghoshdastidar & Dukkipati, 2015; Ke et al., 2019), as well as tensor-based latent models for higher-order link prediction (Yuan & Qu, 2023). However, these approaches rely on uniformity restrictions on hyperlink sizes and cannot accommodate multiplicity, i.e., repeated observations of identical hyperlinks. Deep neural architectures such as HGNN (Feng et al., 2019), Hyper-SAGNN (Zhang et al., 2019), and AllSet (Chien et al., 2021) primarily focus on representation learning and prediction rather than hyperlink generation. For generative modeling of hypergraphs, general-purpose generative approaches include variational autoencoders (Kingma & Welling, 2013), Wasserstein generative adversarial networks (Arjovsky et al., 2017), and Gaussian score-based diffusion (Song et al., 2020). Continuous-output generators require additional processing to produce discrete hyperlinks, whereas Bernoulli diffusion (Sohl-Dickstein et al., 2015) and TabDDPM (Kotelnikov et al., 2023) can accommodate discrete hyperlinks directly. Nevertheless, their generic formulations do not explicitly exploit hypergraph-specific structure and can be computationally demanding when applied

[1] https://www.cms.gov/medicare/coding-billing/icd-10-codes/icd-9-cm-diagnosis-procedure-codes-abbreviated-and-full-code-titles

directly to large-scale hypergraphs. Hypergraph-specific generators include HyperLap (Lee et al., 2021), which targets hyperlink overlap patterns, and HyperPLR (Wen & Yu, 2025), which generates weighted graph projections and reconstructs hyperlinks. Wu et al. (2026) propose hyperlink–node interaction models, with extensions for generative modeling (Wu et al., 2025; Ma et al., 2026), which primarily employ Euclidean inner-product models in their implementations.

Hyperbolic geometry. Hyperbolic geometry has been a powerful tool in mathematics and physics, from the geometry and topology of three-dimensional manifolds (Thurston, 1986) to relativistic velocity addition and Lorentz transformations (Rhodes & Semon, 2004). In machine learning, Poincaré and Lorentz embeddings exploit this geometry to capture hierarchy and similarity in low dimensions (Nickel & Kiela, 2017; 2018), while hyperbolic graph convolutional architectures incorporate it into neural message passing (Chami et al., 2019). Smith et al. (2019) examine how latent geometry shapes pairwise interactions, and Lubold et al. (2023) develop methods for identifying manifold type, dimension, and curvature. Li et al. (2026) develop a hyperbolic latent-space model for pairwise interactions with learnable curvature and consistency guarantees for likelihood-based estimation. For hypergraphs, Fritz et al. (2025) focus on scalable sample-to-population estimation of hyperbolic node positions to characterize core–periphery structure and proximity. To the best of our knowledge, our work is the first to exploit the rich geometry of hyperbolic space for hypergraph generative modeling with learnable latent geometry and efficient training and sampling processes.

1.2 Contribution and organization of the paper

The contributions of this work are threefold:

1. Methodologically, we introduce the Hyperbolic Hypergraph Generator, a latent-space generative model for hypergraphs that balances parsimony and expressivity through hyperbolic geometry, node-specific degree parameters, and a flexible hyperlink-embedding distribution. We further develop a two-stage procedure, H2G, for learning the model from an observed hypergraph and generating previously unseen hyperlinks.
2. Theoretically, we study model identifiability, including curvature and latent geometry, characterize the induced higher-order dependence of the model, and compare the expressivity of two-dimensional hyperbolic spaces with that of Euclidean spaces of increasing dimension. We further relate the discrepancy between the true hyperlink distribution and the distribution generated by H2G to errors in parameter estimation and latent distribution recovery.
3. Empirically, we evaluate two implementations of H2G, referred to as H2G-EB and H2G-RD, through simulations and real-data applications. Extensive comparisons with baseline methods show that H2G captures meaningful latent geometry and achieves strong generative performance, with leading results on fidelity metrics that measure the reproduction of key structural characteristics of real-world hypergraphs.

The remainder of the paper is organized as follows. Section 2 introduces the hyperbolic hypergraph generator and discusses its geometric and probabilistic properties. Section 3 presents the training and sampling procedures together with their theoretical guarantees. Section 4 reports empirical studies on real-world hypergraphs, and Section 5 concludes the paper with a discussion. Additional theoretical results and formal statements, proofs, implementation details, supplementary simulation studies, and additional real-data analyses are provided in the appendices.

2 Generating hypergraphs from latent hyperbolic spaces

We denote a hypergraph by $\mathcal{H}(\mathcal{V}_n, \mathcal{E}_m)$, where $\mathcal{V}_n$ is the node set and $\mathcal{E}_m = \{e_1, \ldots, e_m\}$ is the collection of m *observed* hyperlinks. For notational convenience, we let $\mathcal{V}_n = [n]$, indexing the n nodes. Each e_j, for $j \in [m]$, is a subset of $[n]$, indicating the nodes in that hyperlink. We aim to model the distribution of a random hyperlink E, valued in the power set of $[n]$, i.e., the collection of all subsets of $[n]$, with $\{e_j\}_{j\in[m]}$ as observed realizations. In this section, we introduce a generative model for hyperlinks based on latent interactions in a hyperbolic space and discuss its properties. We introduce the basic concepts of hyperbolic spaces in Section 2.1 and describe how hyperlinks are generated from latent hyperbolic spaces in Section 2.2. We then describe, in Section 3, how to learn

the generative distribution from the observed hypergraph and sample new hyperlinks that were not seen during training.

2.1 Hyperbolic space and the Poincaré disk

A hyperbolic space is a non-Euclidean space with constant negative sectional curvature. In contrast to Euclidean space, where the volume of a ball grows polynomially with its radius, the volume of a hyperbolic ball grows exponentially. This additional geometric capacity allows low-dimensional hyperbolic spaces to represent hierarchical and branching structures with less distortion, providing an expressive yet parsimonious alternative to Euclidean latent spaces.

We adopt the hyperboloid parametrization of a two-dimensional hyperbolic space with constant sectional curvature $-K < 0$. Throughout this work, we focus on the two-dimensional case, while extensions to general d-dimensional hyperbolic spaces are discussed in Appendix A. Let $\mathbf{\Lambda} = \mathrm{diag}(1, 1, -1) \in \mathbb{R}^{3\times 3}$ and define the Lorentz inner product on $\mathbb{R}^3$ by $\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathrm{L}} := \boldsymbol{x}^\top \mathbf{\Lambda} \boldsymbol{y}$. The hyperboloid model is the metric space

$$\mathbb{H}_2^{-K} = \left\{ \boldsymbol{x} = (x_1, x_2, x_3)^\top \in \mathbb{R}^3 : \langle \boldsymbol{x}, \boldsymbol{x}\rangle_{\mathrm{L}} = -1,\ x_3 > 0 \right\},$$

equipped with the hyperbolic distance

$$d_{\mathbb{H}_2^{-K}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{K}} \operatorname{arcosh}(-\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathrm{L}}),\ \boldsymbol{x}, \boldsymbol{y} \in \mathbb{H}_2^{-K}.$$

Although points in $\mathbb{H}_2^{-K}$ are represented by three coordinates, the hyperboloid constraint gives the space an intrinsic dimension of two. An equivalent representation of $\mathbb{H}_2^{-K}$ is the Poincaré disk $\mathbb{D}_2^{-K} = \{\boldsymbol{u} \in \mathbb{R}^2 : \|\boldsymbol{u}\|_2 < 1\}$. In particular, the map $\pi : \mathbb{H}_2^{-K} \to \mathbb{D}_2^{-K}, \pi(\boldsymbol{x}) = \frac{(x_1, x_2)^\top}{1+x_3}$, is an isometry, so that $d_{\mathbb{D}_2^{-K}}(\pi(\boldsymbol{x}), \pi(\boldsymbol{y})) = d_{\mathbb{H}_2^{-K}}(\boldsymbol{x}, \boldsymbol{y})$. The Poincaré disk is conformal, i.e., it preserves angles locally, and represents the entire hyperbolic space within an open 2-dimensional unit disk, making it particularly convenient for visualization. We therefore formulate our generative model and algorithmic implementation using the hyperboloid representation and map the learned latent embeddings to the Poincaré disk for visualization.

Beyond these representational conveniences, negative curvature offers a favorable balance between parsimony and expressivity. With only two intrinsic coordinates per embedding, $\mathbb{H}_2^{-K}$ can represent collections of well-separated latent positions, for which preserving distances within a fixed multiplicative error requires a Euclidean dimension that grows with the collection's size. Thus, even a relatively high but fixed Euclidean dimension may not match the geometric expressivity of two hyperbolic dimensions. One example is embedding hierarchical tree metrics. We formalize these comparisons with discussions in Appendix A.

2.2 Generating hyperlinks using latent hyperbolic geometry

We introduce the hyperbolic hypergraph generator in this section. In the hyperbolic hypergraph generator, each node $i \in [n]$ is associated with a latent embedding $\boldsymbol{z}_i \in \mathbb{H}_2^{-K}$ and a node-specific parameter $\alpha_i \in \mathbb{R}$, which capture its latent position and baseline popularity in generating hyperlinks, respectively. Let $\mathcal{Z}_n = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\} \subset \mathbb{H}_2^{-K}$ and $\boldsymbol{\alpha}_n = (\alpha_1, \ldots, \alpha_n)^\top \in \mathbb{R}^n$. For fixed $K > 0$, $\mathcal{Z}_n$, and $\boldsymbol{\alpha}_n$, we first draw a hyperlink embedding $\mathbf{F}$ from a distribution $\mathbb{P}_{\mathbf{F}}$ supported on $\mathbb{H}_2^{-K}$ and then include nodes conditionally independently:

$$\mathbf{F} \sim \mathbb{P}_{\mathbf{F}},\ \mathbf{1}_{\{i\in E\}} \mid \{\mathbf{F} = \boldsymbol{f}\} \overset{\text{ind}}{\sim} \text{Bernoulli}(p_i(\boldsymbol{f})),\ p_i(\boldsymbol{f}) = \sigma\big(\alpha_i - d_{\mathbb{H}_2^{-K}}(\boldsymbol{f}, \boldsymbol{z}_i)\big), \tag{1}$$

where $\sigma(x) = 1/\{1 + \exp(-x)\}$. Thus, nodes with latent embeddings closer to $\boldsymbol{f}$ are more likely to be included, while α_i controls node-level degree heterogeneity. Let E be a random hyperlink taking values in the collection of subsets of $[n]$, and let $e \subseteq [n]$. Consequently,

$$\mathbb{P}(E = e|\mathbf{F} = \boldsymbol{f}) = \prod_{i\in e} p_i(\boldsymbol{f}) \prod_{i\notin e} \{1 - p_i(\boldsymbol{f})\}$$

Although node inclusions are independent conditional on $\mathbf{F}$, Proposition 1 shows that marginalizing over $\mathbf{F}$ can induce dependence. More broadly, the joint inclusion probability in Proposition 1 need

not factor into the corresponding marginal probabilities, allowing the model to capture general higher-order dependence patterns.

Proposition 1 (Higher-order dependence induced by the hyperlink embedding)**.** *Let* $\Theta_n = (\mathcal{Z}_n, \boldsymbol{\alpha}_n, K, \mathbb{P}_{\mathbf{F}})$. *For any nonempty* $e \subseteq [n]$, $\mathbb{P}(e \subseteq E \mid \Theta_n) = \mathbb{E}_{\mathbf{F}}[\prod_{i\in e} p_i(\mathbf{F})]$. *In particular, for any distinct* $i, k \in [n]$, $\mathrm{Cov}(\mathbf{1}_{\{i\in E\}}, \mathbf{1}_{\{k\in E\}} \mid \Theta_n) = \mathrm{Cov}_{\mathbf{F}}\{p_i(\mathbf{F}), p_k(\mathbf{F})\}$.

We next study identifiability of the latent geometry through its hyperbolic curvature. Let $\boldsymbol{f}_j \in \mathbb{H}_2^{-K}$ denote the embedding of hyperlink $j \in [m]$, and define $\boldsymbol{P} = (p_i(\boldsymbol{f}_j))_{j\in[m],\, i\in[n]}$. Throughout this comparison, the node degree parameters $\boldsymbol{\alpha}_n$ are held fixed; primed quantities denote a second configuration with the same $\boldsymbol{\alpha}_n$ and conditional inclusion matrix $\boldsymbol{P}'$.

Theorem 1 (Identifiability of hyperbolic curvature)**.** *Suppose* $K, K' \in [c_K^{-1}, c_K]$ *for* $c_K \geq 1$, *and all hyperlink–node distances are at least* $c_l > 0$. *Let* u_j, v_i *and* u'_j, v'_i *denote the complex Poincaré coordinates of their hyperlink and node embeddings, respectively. When* $K < K'$, *assume a holomorphic self-map* ϕ *of the complex unit disk satisfies* $\phi(u_j) = u'_j$ *and* $\phi(v_i) = v'_i$ *for every* j, i; *when* $K' < K$, *assume the corresponding map exists in the reverse direction. Then* $|K - K'| \leq \frac{2c_K}{c_l\sqrt{mn}} \|\mathrm{logit}(\boldsymbol{P}) - \mathrm{logit}(\boldsymbol{P}')\|_F$, *where* $\mathrm{logit} = \log\{p/(1-p)\}, p \in (0,1)$ *acts entrywise. In particular,* $\boldsymbol{P} = \boldsymbol{P}'$ *implies* $K = K'$.

Under the compatibility condition in Theorem 1, a change in curvature cannot be absorbed by relocating the latent embeddings while preserving the conditional inclusion probabilities. The bound also links curvature recovery to the recovery of the conditional logits. This relationship between the latent curvature and the probability matrix $\mathbf{P}$ also motivates the likelihood-based method in Section 3 for identifying the latent curvature and embedding the observed hypergraph. Further discussion of the condition and the proof are provided in Appendix E.3.

3 Training and sampling

In this section, we describe the training and sampling procedures for the hyperbolic hypergraph generator, which we refer to as H2G. The overall pipeline of H2G is summarized in Figure 2. In brief, H2G is trained in two stages. We first embed the observed hypergraph into the hyperbolic space via maximum-likelihood estimation under the hyperbolic hyperlink model, yielding estimates of the node embeddings, node-specific degree parameters, curvature, and an embedding for each observed hyperlink. We then learn the distribution of the estimated hyperlink embeddings in the latent hyperbolic space. This learning procedure can be general, as long as the distribution can be learned accurately and is easy to sample from. To generate a new hyperlink, we sample a hyperlink embedding from the learned distribution and then include each node according to the hyperbolic hypergraph generation mechanism, using the learned node parameters and the newly sampled hyperlink embedding. Below, we specify the two stages.

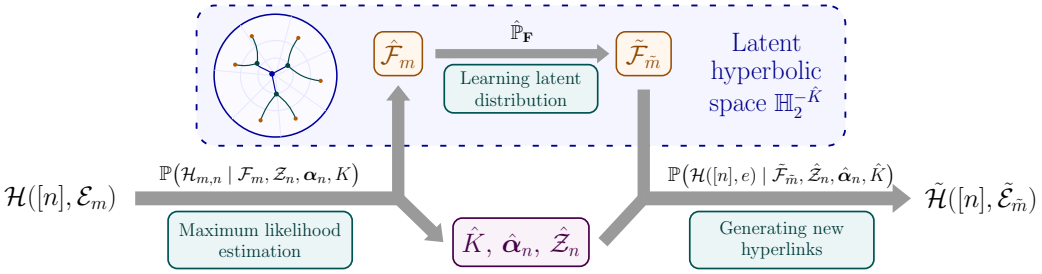

Figure 2: The general pipeline of H2G.

Hyperbolic embedding via maximum likelihood estimation. Write $\mathcal{F}_m = \{\boldsymbol{f}_j\}_{j=1}^m$ where $\boldsymbol{f}_j$ is the hyperlink embedding associated with the observed hyperlink e_j. Conditional on the hyperlink

Algorithm 1 Maximum-likelihood embedding and estimation

Require: Hypergraph $\mathcal{H}([n], \mathcal{E}_m)$; curvature grid $\mathcal{K}$; initial values $\boldsymbol{\alpha}^{(0)}, \boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)}$; step sizes $\eta_\alpha, \eta_F, \eta_Z$; tolerance $\varepsilon > 0$.

```
for K ∈ 𝒦 do
    (α, F, Z) ← (α^(0), F^(0), Z^(0)).
    repeat
        L_old ← ℒ(K, α, F, Z).
        (F, Z) ← MGD(K, α, F, Z; η_F, η_Z); α ← α − η_α ∇_α ℒ(K, α, F, Z).
    until |ℒ(K, α, F, Z) − L_old| ≤ ε
    (α̂(K), F̂(K), Ẑ(K)) ← (α, F, Z); ℓ*_K ← −ℒ(K, α, F, Z).
end for
K̂ ← arg max_{K∈𝒦} ℓ*_K;   return K̂, α̂(K̂), F̂(K̂), Ẑ(K̂).
```

embeddings, the log-likelihood of the observed hypergraph is

$$\ell(\mathcal{F}_m, \mathcal{Z}_n, \boldsymbol{\alpha}_n, K) = \sum_{j=1}^{m} \sum_{i=1}^{n} [\mathbf{1}_{\{i \in e_j\}} \log p_i(\boldsymbol{f}_j) + \mathbf{1}_{\{i \notin e_j\}} \log\{1 - p_i(\boldsymbol{f}_j)\}],$$

where $p_i(\boldsymbol{f})$ is defined in Equation 1. To embed the observed hypergraph into the hyperbolic space, we maximize the training log-likelihood for each candidate curvature K over the node-specific degree parameters and the node and hyperlink embeddings, with $\boldsymbol{f}_j, \boldsymbol{z}_i \in \mathbb{H}_2^{-K}$. Denote the resulting estimates by $(\widehat{K}, \widehat{\mathcal{F}}_m, \widehat{\mathcal{Z}}_n, \widehat{\boldsymbol{\alpha}}_n)$. At this stage, the hyperlink embeddings are treated as unknown parameters, without specifying the population distribution $\mathbb{P}_{\mathbf{F}}$.

Algorithm 1 implements this estimation through a grid search over curvature and alternating gradient updates. Let $\boldsymbol{F} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m)^\top$, $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)^\top$, and $\boldsymbol{\alpha} = \boldsymbol{\alpha}_n$, and write $\mathcal{L}(K, \boldsymbol{\alpha}, \boldsymbol{F}, \boldsymbol{Z}) = -\ell(\mathcal{F}_m, \mathcal{Z}_n, \boldsymbol{\alpha}_n, K)$ for the negative log-likelihood. For each candidate $K \in \mathcal{K}$, we alternate between Manifold Gradient Descent (MGD) (Nickel & Kiela, 2018) for the node and hyperlink embeddings and a Euclidean gradient step for the degree parameters. The manifold updates preserve the hyperbolic constraints on the embeddings, and the iterations terminate when the absolute change in $\mathcal{L}$ does not exceed ε. Algorithm 1 selects the curvature using the in-sample training log-likelihood. When a validation set is available, curvature can alternatively be selected based on the validation performance of the trained models. In the reported simulation setting, both criteria recover the ground-truth curvature; see Appendix C.2.5. Additional implementation details are provided in Appendices B.1 and B.2.

Learning the latent distribution and generating hyperlinks. The estimated hyperlink embeddings $\hat{\mathcal{F}}_m$ serve as an estimated sample for learning $\mathbb{P}_{\mathbf{F}}$ on $\mathbb{H}_2^{-\widehat{K}}$. We consider two implementations: empirical bootstrap (H2G-EB), which samples from $m^{-1} \sum_{j=1}^{m} \delta_{\widehat{\boldsymbol{f}}_j}$, where $\delta_{\boldsymbol{f}}$ denotes a point mass at $\boldsymbol{f}$, and a Riemannian diffusion model (H2G-RD) trained on $\widehat{\mathcal{F}}_m$. The former directly resamples the estimated embeddings, whereas the latter learns a latent-space generator that respects the hyperbolic geometry. The resulting embedding distribution is denoted by $\hat{\mathbb{P}}_{\mathbf{F}}$.

Whereas EB resamples the estimated hyperlink embeddings, RD learns their distribution on $\mathcal{M} = \mathbb{H}_2^{-\widehat{K}}$ through Riemannian score-based diffusion (De Bortoli et al., 2022). Initialized from these embeddings, the forward process $\mathrm{d}X_t = -\frac{1}{2}\beta(t)\nabla_{\mathcal{M}} U(X_t)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}B_t^{\mathcal{M}}$ gradually perturbs them toward a tractable reference density proportional to e^{-U}. Here, $\beta(t) > 0$ is a noise schedule, $\nabla_{\mathcal{M}}$ is the Riemannian gradient, and $B_t^{\mathcal{M}}$ is Brownian motion on $\mathcal{M}$. We train a tangent-valued neural network $s_\theta(x, t)$ to approximate $\nabla_{\mathcal{M}} \log p_t(x)$ through denoising score matching, where p_t denotes the perturbed density and all densities are defined relative to Riemannian volume. The learned score determines the reverse diffusion used for sampling. The training objective and implementation details are provided in Appendix B.4. To generate $\tilde{m}$ new hyperlinks, we draw $\widetilde{\boldsymbol{f}}_1, \ldots, \widetilde{\boldsymbol{f}}_{\tilde{m}} \overset{\text{i.i.d.}}{\sim} \widehat{P}_F$. For each $j \in [\tilde{m}]$, conditional on $\widetilde{\boldsymbol{f}}_j$, we independently include each node $i \in [n]$ in $\tilde{e}_j$ with probability $\widehat{p}_i(\widetilde{\boldsymbol{f}}_j)$, obtained by substituting the estimated node parameters and curvature into p_i. This produces a synthetic hypergraph $\widetilde{\mathcal{H}}([n], \widetilde{\mathcal{E}}_{\tilde{m}})$ on the same node set, where $\widetilde{\mathcal{E}}_{\tilde{m}} = \{\tilde{e}_1, \ldots, \tilde{e}_{\tilde{m}}\}$.

Theoretical guarantees of H2G. We next quantify how errors in the two learning stages affect the population distribution of generated hyperlinks. The following theorem bounds the generative error in terms of parameter estimation and latent distribution recovery.

Theorem 2 (Generative consistency)**.** *Let $\mathbb{P}_E$ denote the true hyperlink law and $\mathbb{P}_{\tilde{E}}$ the law of a hyperlink $\tilde{E}$ generated by the fitted H2G model, conditional on the training data. Under Condition 1, with estimation error $r_{m,n}$ and latent reconstruction error ε_m defined in Appendix E.4,*

$$\frac{1}{n} D_{\mathrm{KL}}(\mathbb{P}_E \,\|\, \mathbb{P}_{\tilde{E}}) \le C\big(r_{m,n}^2 + \varepsilon_m^2\big) + O_p(m^{-1/2}),$$

where C does not depend on m or n. Consequently, if $r_{m,n} + \varepsilon_m = o_p(1)$ as $m, n \to \infty$, then $\mathbb{P}_{\tilde{E}}$ is consistent for $\mathbb{P}_E$ in normalized KL divergence.

The bound separates the generative error into parameter estimation, latent distribution reconstruction, and finite-sample variation. For H2G-EB, $\varepsilon_m = 0$ because the fitted latent law is the empirical distribution of the estimated hyperlink embeddings, whereas for H2G-RD, ε_m additionally captures diffusion approximation error. Thus, consistency follows when both learning stages are sufficiently accurate. Full assumptions and proofs are given in Appendix E.4, with additional discussion of the diffusion error in Appendix E.4.2.

4 Empirical studies

In this section, we empirically evaluate H2G on real-world hypergraphs. Implementation details, additional simulation results, and dataset pre-processing with additional real-data results and analyses are provided in Appendices B, C, and D, respectively.

4.1 Experimental setup

Datasets. We report results on three real-world hypergraphs: (i) the symptom co-occurrence hypergraph from MIMIC-III (Johnson et al., 2016), where symptoms are nodes and symptoms co-occurring within a patient record form a hyperlink; (ii) MathOverflow-Answers (Benson et al., 2018), where questions are nodes and questions answered by a user form a hyperlink; and (iii) the protein-complex hypergraph from CORUM (Tsitsiridis et al., 2023; Steinkamp et al., 2025), where proteins are nodes and proteins within a complex form a hyperlink. After pre-processing, the sizes (n, m) are $(891, 1000)$, $(3000, 2272)$, and $(5227, 5338)$, respectively. For all datasets, we use a fixed 8:1:1 train/validation/test split shared across methods, with validation for hyperparameter selection and test for evaluation. For each method, we generate 32 times the number of hyperlinks in the training split. Construction and pre-processing details are provided in Appendix D.2.

Baselines. We compare the two H2G implementations (H2G-EB, H2G-RD) against eight baselines for synthetic hyperlink generation: DDE (Wu et al., 2025), Gau-Diff (Song et al., 2020), WGAN (Arjovsky et al., 2017; Gulrajani et al., 2017), TabDDPM (Kotelnikov et al., 2023), Ber-Diff (Sohl-Dickstein et al., 2015), VAE (Kingma & Welling, 2013), HyperLap (Lee et al., 2021), HyperPLR (Wen & Yu, 2025). HyperPLR is omitted from all real-world datasets and Ber-Diff from all but MIMIC-III due to GPU memory limits in the setup. Notably, HyperPLR was originally evaluated on substantially smaller hypergraphs. Gau-Diff, TabDDPM, and WGAN produce continuous outputs and therefore require post-hoc calibration using the observed hypergraph to obtain binary hyperlink vectors. We refer readers to Appendix B.5 for more details and discussion of these methods.

4.2 Real-world results and analysis

Evaluation metrics. We evaluate generation fidelity using metrics of the hyperlink distribution together with marginal and pairwise statistics. The Energy distance (Székely & Rizzo, 2013) and squared maximum mean discrepancy (MMD^2) (Gretton et al., 2012) compare the overall distributions of real and generated hyperlinks. We further report the root mean squared error (RMSE) of node frequencies (Δ_{H_d}) and node co-occurrence covariance (Δ_{H_v}), which assess marginal frequencies and the covariance structure of node occurrences, respectively. Smaller values indicate closer correspondence between real and generated hyperlinks. The Energy distance uses the Jaccard

Method	MIMIC-III (Fidelity ↓)				MathOverflow-Answers (Fidelity ↓)			
	Energy ($\times 10^{-3}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-2}$)	Δ_{H_v} ($\times 10^{-3}$)	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-3}$)	Δ_{H_v} ($\times 10^{-4}$)
H2G-EB	<u>1.30</u> (0.07)	<u>7.44</u> (0.08)	<u>2.16</u> (0.01)	<u>4.17</u> (0.00)	**3.71** (0.47)	**3.50** (0.03)	**4.08** (0.01)	**4.08** (0.00)
H2G-RD	**1.23** (0.02)	**7.41** (0.08)	**2.13** (0.00)	4.20 (0.00)	<u>5.74</u> (0.14)	<u>3.56</u> (0.04)	4.96 (0.03)	4.26 (0.01)
DDE	1.35 (0.11)	7.47 (0.11)	2.21 (0.01)	**4.16** (0.00)	30.2 (5.5)	5.34 (0.76)	<u>4.20</u> (0.02)	<u>4.13</u> (0.01)
Gau-Diff^c	1.49 (0.04)	7.53 (0.11)	2.16 (0.00)	4.32 (0.00)	6.20 (0.36)	3.60 (0.03)	4.00 (0.00)	**4.08** (0.00)
TabDDPM^c	1.59 (0.09)	7.59 (0.10)	2.16 (0.00)	4.33 (0.00)	58.0 (3.3)	6.42 (0.53)	4.00 (0.00)	14.0 (0.02)
WGAN^c	1.65 (0.05)	7.66 (0.02)	2.16 (0.00)	4.45 (0.00)	47.1 (4.2)	5.94 (0.24)	4.00 (0.00)	5.06 (0.05)
Ber-Diff	163 (0.37)	71.5 (1.1)	203 (0.43)	307 (1.4)	OOM	OOM	OOM	OOM
VAE	163 (0.76)	83.1 (1.6)	14.4 (0.13)	39.8 (0.51)	1220 (37)	52.9 (0.99)	250 (0.65)	598 (3.0)
HyperLap	144 (4.4)	55.4 (1.2)	11.3 (0.18)	5.49 (0.02)	20.5 (0.75)	4.05 (0.03)	5.89 (0.06)	4.27 (0.01)

Table 1: Hypergraph generation performance on MIMIC-III and MathOverflow-Answers. Superscript ^{c} marks methods using post-hoc calibration. OOM denotes out-of-memory failure.

distance (Kosub, 2019) between binary hyperlink vectors, capturing similarity through overlap of their active node sets, while MMD^2 uses a Jaccard-based RBF kernel. Detailed definitions and implementation settings are provided in Appendix B.6.

Generative performance. Tables 1 and 2 summarize mean (standard deviation) over five random sampling seeds with fixed training seed. Means use three significant digits, while standard deviations use the shorter of two significant digits and two decimal places. For Δ_{H_d}, we rank only uncalibrated methods, since Gau-Diff, TabDDPM, and WGAN require post-hoc calibration.

Method	CORUM (Fidelity ↓)			
	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-3}$)	Δ_{H_v} ($\times 10^{-5}$)
H2G-EB	<u>8.31</u> (0.83)	<u>2.29</u> (0.12)	**1.22** (0.00)	**4.55** (0.00)
H2G-RD	**6.23** (0.49)	**2.12** (0.06)	1.35 (0.01)	4.70 (0.01)
DDE	2530 (84)	163 (7.2)	<u>1.27</u> (0.00)	4.74 (0.02)
Gau-Diff^c	13.1 (2.5)	2.63 (0.23)	1.18 (0.00)	<u>4.58</u> (0.00)
TabDDPM^c	7420 (110)	471 (9.2)	1.18 (0.00)	5.90 (0.94)
WGAN^c	1440 (27)	96.2 (5.8)	1.18 (0.00)	17.7 (0.34)
VAE	2570 (43)	141 (4.3)	201 (0.90)	5590 (22)
HyperLap	17.7 (0.08)	2.46 (0.02)	1.84 (0.00)	5.35 (0.01)

Table 2: Hypergraph generation performance on CORUM. Superscript ^{c} marks methods using post-hoc calibration.

Overall, both H2G implementations achieve strong generation fidelity across all three real-world datasets. For Energy and MMD^2, H2G-RD leads on MIMIC-III and CORUM, while H2G-EB leads on MathOverflow-Answers. On CORUM and MathOverflow-Answers, several baselines have values up to three orders of magnitude larger, with gaps amplified by the high sensitivity of Jaccard-based discrepancies in sparse hypergraphs, highlighting H2G's stronger recovery of selective co-membership patterns. For Δ_{H_d}, H2G-RD is the best uncalibrated method on MIMIC-III, while H2G-EB leads on MathOverflow-Answers and CORUM, comparable to calibrated baselines without post-hoc marginal calibration. For Δ_{H_v}, H2G-EB leads on CORUM, ties for best on MathOverflow-Answers, and remains competitive on MIMIC-III. Together, these results show that H2G closely reproduces the hyperlink distribution, node marginals, and node co-occurrence covariance structure. Notably, H2G uses only a two-dimensional hyperbolic latent space, yet remains competitive with or outperforms DDE tuned over higher-dimensional Euclidean embeddings; see Appendix D.3.

Beyond generation fidelity, we also examine whether the estimated node effects $\hat{\alpha}_i$ capture node-level heterogeneity. We use Spearman's rank correlation (Spearman, 1987), which measures monotonic association without assuming linearity. The correlations with node degrees in hypergraphs generated by H2G-RD are 0.976 on MIMIC-III, 0.804 on MathOverflow-Answers, and 0.857 on CORUM, indicating that $\hat{\alpha}_i$ effectively captures heterogeneity in node popularity.

Hyperbolic embeddings and generated hyperlinks. We next examine the learned hyperbolic embeddings and the structure of generated hyperlinks. Figure 3 shows seven clusters obtained by

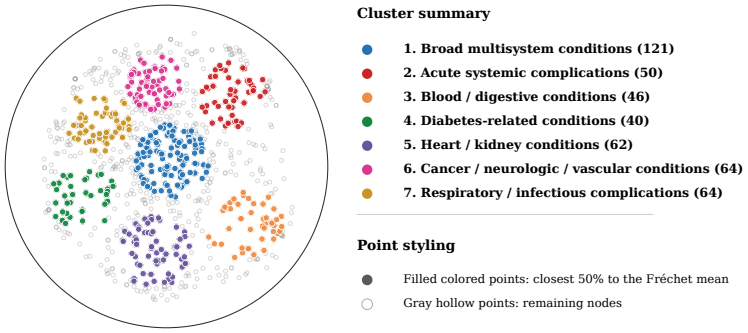

Figure 3: Riemannian k-means clustering of the learned symptom embeddings on MIMIC-III into seven clusters. Colored points denote the 50% of nodes with the smallest hyperbolic distances to the Fréchet mean of each cluster, while gray hollow points denote the remaining nodes. The clusters exhibit recognizable clinical comorbidity patterns, with the central cluster capturing broader multisystem conditions.

Riemannian k-means (Turaga et al., 2011) on the learned MIMIC-III symptom embeddings. Cluster assignments are based on hyperbolic geodesic distances, and cluster centers are estimated by Fréchet means (Pennec, 2006). The clusters exhibit recognizable clinical comorbidity patterns, while the central cluster contains broader multisystem conditions and several peripheral clusters capture more specific clinical patterns. This organization is consistent with the hierarchical structure naturally represented in hyperbolic space (Nickel & Kiela, 2017). Representative diagnoses for all seven clusters are provided in Appendix D.5.3.

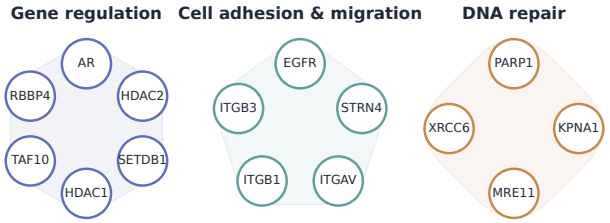

Figure 4: Representative protein complexes generated by H2G-RD on CORUM. Each shaded region denotes one generated hyperlink, and the title above each complex summarizes its dominant biological function inferred from the constituent proteins.

At the hyperlink level, the generated samples also exhibit coherent domain-specific structure. Figure 4 shows three representative generated hyperlinks whose constituent proteins are associated with distinct biological functions, including gene regulation, cell adhesion and migration, and DNA repair. These examples suggest that H2G can generate functionally coherent protein sets, indicating potential utility for proposing candidate protein associations and previously uncharacterized complexes for downstream biological investigation. The full protein names, functional interpretation of each generated complex, and supporting biological evidence are provided in Appendix D.7.3.

5 Conclusion

In this work, we introduce the Hyperbolic Hypergraph Generator, a probabilistic latent-space model that balances parsimony and expressivity through hyperbolic geometry, and develop an efficient training and sampling procedure, H2G. We study curvature identifiability and geometric expressivity and establish conditions for normalized KL consistency of the generated hyperlink distribution. Real-world experiments demonstrate strong generation fidelity and meaningful latent organization in two dimensions, highlighting geometry as a source of expressivity beyond increasing embedding dimension or decoder complexity.

We also acknowledge several limitations that suggest future research directions. The framework assumes constant curvature and operates on a fixed node set. Future work could assess robustness to geometric misspecification, develop adaptive procedures for selecting the latent dimension or geometry, and establish sharper theoretical guarantees for optimization and diffusion. Moreover, the framework could be extended to incorporate attributes and temporal evolution for conditional generation and to generate hyperlinks involving previously unseen nodes.

AI USE STATEMENT

In this work, we have not used generative AI tools to generate synthetic datasets; help develop theoretical models or conceptual frameworks; formulate mathematical claims; provide critical ingredients for proving mathematical claims; assist in the writing of proofs; propose or refine hypotheses; design or provide feedback on research methodology or experiments; implement methods; assist with translation; clean or reformat datasets; or interpret results. Support for qualitative and thematic data analysis was not applicable to this work.

Additionally, we used generative AI tools for sourcing and searching for information, identifying relevant literature, editing the paper to improve readability, and creating or editing software code.

We have reviewed all AI-assisted work. References identified with the assistance of generative AI were manually checked by the authors against the original sources. AI-assisted language edits were reviewed by the authors to ensure that they preserved the intended technical meaning. AI-assisted code debugging and checking were independently reviewed and tested by the authors for correctness.

We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only existing datasets and does not involve the collection of new human-subject data. MIMIC-III contains de-identified clinical records and is accessed under the credentialing and data-use requirements of PhysioNet, while the remaining datasets are publicly available from their respective sources; dataset construction and pre-processing are described in Appendix D.2. Although our method generates synthetic hyperlinks, such data should not be assumed to provide a formal privacy guarantee, particularly when models are trained on sensitive clinical records. We use these datasets solely for methodological evaluation and do not make clinical or individual-level decisions based on the generated data.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproduction of both the theoretical and empirical results. Additional theoretical statements and complete proofs are provided in Appendix A and Appendix E, respectively. Appendix B gives implementation and experimental details for H2G, including likelihood-based training and optimization (Appendix B.1), parameter initialization (Appendix B.2), empirical bootstrap, and the Riemannian diffusion procedure used in H2G-RD (Appendix B.4). Baseline implementations and post-hoc calibration are documented in Appendix B.5, with the DDE embedding-dimension selection and sensitivity analysis reported in Appendix D.3. Evaluation metrics are defined in Appendix B.6. Appendix C provides the simulation design together with additional results on parameter recovery, curvature selection, and generative performance. Appendix D provides the full real-data experimental protocol, dataset construction and pre-processing, DDE embedding-dimension selection and sensitivity analysis, memorization, novelty, and privacy-related diagnostics, and dataset-specific optimization, latent-space, and qualitative analyses. Anonymous source code and scripts for reproducing the experiments are available at `https://anonymous.4open.science/r/H2G-E231`.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223. PMLR, 2017.

Austin R Benson, Rediet Abebe, Michael T Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, 115 (48):E11221–E11230, 2018.

Sara Cabodi, Laura Moro, E Bergatto, E Boeri Erba, Paola Di Stefano, Emilia Turco, Guido Tarone, and Paola Defilippi. Integrin regulation of epidermal growth factor (egf) receptor and of egf-dependent responses. *Biochemical Society Transactions*, 32(3):438–442, 2004.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32, 2019.

Sourav Chatterjee. Matrix estimation by universal singular value thresholding. *Annals of Statistics*, 43:177–214, 2015.

Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. *arXiv preprint arXiv:2106.13264*, 2021.

Kern Rei Chng, Cheng Wei Chang, Si Kee Tan, Chong Yang, Shu Zhen Hong, Noel Yan Wei Sng, and Edwin Cheung. A transcriptional repressor co-regulatory network governing androgen response in prostate cancers. *The EMBO journal*, 31(12):2810–2823, 2012.

Sunwha Cho, Jung Sun Park, and Yong-Kook Kang. Ago2 and setdb1 cooperate in promoter-targeted transcriptional silencing of the androgen receptor gene. *Nucleic acids research*, 42(22): 13545–13556, 2014.

Damien D'amours and Stephen P Jackson. The mre11 complex: at the crossroads of dna repair and checkpoint signalling. *Nature reviews Molecular cell biology*, 3(5):317–327, 2002.

Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian score-based generative modelling. *Advances in neural information processing systems*, 35:2406–2422, 2022.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019.

Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability theory and related fields*, 162(3):707–738, 2015.

Cornelius Fritz, Yubai Yuan, and Michael Schweinberger. Scalable sample-to-population estimation of hyperbolic space models for hypergraphs, 2025.

Debarghya Ghoshdastidar and Ambedkar Dukkipati. A provable generalized tensor spectral method for uniform hypergraph partitioning. In *International Conference on Machine Learning*, pp. 400–409. PMLR, 2015.

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.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. volume 30, 2017.

Chin-Wei Huang, Milad Aghajohari, Joey Bose, Prakash Panangaden, and Aaron Courville. Riemannian diffusion models. volume 35, pp. 2750–2761, 2022.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985. doi: 10.1007/BF01908075.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):160035, 2016.

Zheng Tracy Ke, Feng Shi, and Dong Xia. Community detection for hypergraph networks via regularized tensor power iteration. *arXiv preprint arXiv:1909.06503*, 2019.

Richard DW Kelly and Shaun M Cowley. The physiological roles of histone deacetylase (hdac) 1 and 2: complex co-stars with multiple leading parts. *Biochemical Society Transactions*, 41(3): 741–749, 2013.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. 2013.

Sven Kosub. A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters*, 120:36–38, 2019.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In *International conference on machine learning*, pp. 17564–17579. PMLR, 2023.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 82(3):036106, 2010.

Tarun Kumar, Sankaran Vaidyanathan, Harini Ananthapadmanabhan, Srinivasan Parthasarathy, and Balaraman Ravindran. Hypergraph clustering by iteratively reweighted modularity maximization. *Applied Network Science*, 5(1):52, 2020.

Sarah Lawson, Diane Donovan, and James Lefevre. An application of node and edge nonlinear hypergraph centrality to a protein complex hypernetwork. *PloS one*, 19(10):e0311433, 2024.

Geon Lee, Minyoung Choe, and Kijung Shin. How do hyperedges overlap in real-world hypergraphs?-patterns, measures, and generators. In *Proceedings of the web conference 2021*, pp. 3396–3407, 2021.

Jinming Li, Gongjun Xu, and Ji Zhu. Hyperbolic network latent space model with learnable curvature. *Journal of the American Statistical Association*, (just-accepted):1–20, 2026.

Shane Lubold, Arun G Chandrasekhar, and Tyler H McCormick. Identifying the latent space geometry of network models through analysis of curvature. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 85(2):240–292, 2023.

Feiyan Ma, Shihao Wu, Gongjun Xu, and Ji Zhu. ReLaSH: Reconstructing joint latent spaces for efficient generation of synthetic hypergraphs with hyperlink attributes. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zhuang Ma, Zongming Ma, and Hongsong Yuan. Universal latent space model fitting for large networks with edge covariates. *Journal of Machine Learning Research*, 21(4):1–67, 2020.

Christopher J Millard, Niranjan Varma, Almutasem Saleh, Kyle Morris, Peter J Watson, Andrew R Bottrill, Louise Fairall, Corinne J Smith, and John WR Schwabe. The structure of the core nurd repression complex provides insights into its interaction with chromatin. *Elife*, 5:e13941, 2016.

Nina Miolane, Nicolas Guigui, Alice Le Brigant, Johan Mathe, Benjamin Hou, Yann Thanwerdas, Stefan Heyder, Olivier Peltre, Niklas Koep, Hadi Zaatiti, et al. Geomstats: A python package for riemannian geometry in machine learning. *Journal of Machine Learning Research*, 21(223):1–9, 2020.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International conference on machine learning*, pp. 3779–3788. PMLR, 2018.

Xavier Pennec. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1):127–154, 2006.

Arnab Ray Chaudhuri and André Nussenzweig. The multifaceted roles of parp1 in dna repair and chromatin remodelling. *Nature reviews Molecular cell biology*, 18(10):610–621, 2017.

Giordano Reginato and Petr Cejka. The mre11 complex: A versatile toolkit for the repair of broken dna. *DNA repair*, 91:102869, 2020.

John A Rhodes and Mark D Semon. Relativistic velocity space, wigner rotation, and thomas precession. *American journal of physics*, 72(7):943–960, 2004.

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International symposium on graph drawing*, pp. 355–366. Springer, 2011.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy*, pp. 3–18. IEEE, 2017.

Anna L Smith, Dena M Asta, and Catherine A Calder. The geometry of continuous latent space models for network data. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 34(3):428, 2019.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. 2020.

Charles Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 100(3/4):441–471, 1987.

Ralph Steinkamp, George Tsitsiridis, Barbara Brauner, Corinna Montrone, Gisela Fobo, Goar Frishman, Sorin Avram, Tudor I Oprea, and Andreas Ruepp. Corum in 2024: protein complexes as drug targets. *Nucleic acids research*, 53(D1):D651–D657, 2025.

Gábor J Székely and Maria L Rizzo. Energy statistics: A class of statistics based on distances. *Journal of statistical planning and inference*, 143(8):1249–1272, 2013.

Joanna R Thomas, Nikki R Paul, and Mark R Morgan. Adhesion and growth factor receptor crosstalk mechanisms controlling cell migration. *Essays in Biochemistry*, 63(5):553–567, 2019.

William P Thurston. Hyperbolic structures on 3-manifolds i: Deformation of acylindrical manifolds. *Annals of Mathematics*, 124(2):203–246, 1986.

Morgan P Torchy, Ali Hamiche, and Bruno P Klaholz. Structure and function insights into the nurd chromatin remodeling complex. *Cellular and Molecular Life Sciences*, 72(13):2491–2507, 2015.

George Tsitsiridis, Ralph Steinkamp, Madalina Giurgiu, Barbara Brauner, Gisela Fobo, Goar Frishman, Corinna Montrone, and Andreas Ruepp. Corum: the comprehensive resource of mammalian protein complexes–2022. *Nucleic acids research*, 51(D1):D539–D545, 2023.

Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2273–2286, 2011.

Yunping Wang, Zhiheng Zhou, Dengyi Zhao, and Xingqin Qi. Link predicting for signed hypergraphs under deep learning framework. *Physica A: Statistical Mechanics and its Applications*, pp. 131623, 2026.

Weihuang Wen and Tianshu Yu. Hyperplr: Hypergraph generation through projection, learning, and reconstruction. In *International Conference on Learning Representations*, volume 2025, pp. 21180–21196, 2025.

Shihao Wu, Junyi Yang, Gongjun Xu, and Ji Zhu. Denoising diffused embeddings: A generative approach for hypergraphs. *arXiv preprint arXiv:2501.01541*, 2025.

Shihao Wu, Gongjun Xu, and Ji Zhu. A general latent embedding approach for modeling non-uniform high-dimensional sparse hypergraphs with multiplicity. *Journal of the American Statistical Association*, (just-accepted):1–108, 2026.

Xingyu Xu, Ziyi Zhang, Yorie Nakahira, Guannan Qu, and Yuejie Chi. Polynomial convergence of riemannian diffusion models. 2026.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P. Bennett. Generation and evaluation of privacy preserving synthetic health data. *Neurocomputing*, 416: 244–255, 2020. ISSN 0925-2312.

Yubai Yuan and Annie Qu. High-order joint embedding for multi-level link prediction. *Journal of the American Statistical Association*, 118(543):1692–1706, 2023.

Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-sagnn: a self-attention based graph neural network for hypergraphs. 2019.

Appendix Contents

A Additional theoretical results **15**

B Implementation and experimental details **17**

B.1 Model training and optimization 17

B.2 Initialization of model parameters 19

B.3 Empirical bootstrap 20

B.4 Riemannian diffusion 21

B.5 Baseline implementations 22

B.6 Fidelity metrics 25

B.7 Computational cost and resources 26

C Simulation study and results **26**

C.1 Simulation setup 27

C.2 Recovery of parameters 27

C.3 Generation performance on the simulated dataset 32

D Real-data experiments **32**

D.1 Experimental settings 32

D.2 Datasets and pre-processing 33

D.3 DDE embedding-dimension sensitivity 35

D.4 Memorization, novelty, and privacy analysis 36

D.5 Additional MIMIC-III results and analysis 38

D.6 Additional MathOverflow-Answers results and analysis 42

D.7 Additional CORUM results and analysis 44

E Proofs of theoretical results **47**

E.1 Proof of Proposition A.1 47

E.2 Proof of Proposition 1 48

E.3 Proof of Theorem 1 48

E.4 Consistency of the hyperbolic hypergraph generator 50

A Additional theoretical results

This section collects additional theoretical results that complement the two-dimensional formulation in the main text. We first extend the geometric definitions to general intrinsic dimension, then review

the volume-growth and metric-distortion properties that motivate the use of hyperbolic latent spaces. Finally, we formalize the representational advantage of low-dimensional hyperbolic geometry for branching structures through Proposition A.1.

General-dimensional formulation. For an intrinsic dimension $d \geq 2$ and curvature parameter $K > 0$, let

$$\mathbf{\Lambda}_d = \mathrm{diag}(1, \ldots, 1, -1) \in \mathbb{R}^{(d+1)\times(d+1)}$$

and define the Lorentz inner product by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathrm{L}} = \boldsymbol{x}^\top \mathbf{\Lambda}_d \boldsymbol{y}$. Under the normalized hyperboloid parametrization used in the main text,

$$\mathbb{H}_d^{-K} = \left\{ \boldsymbol{x} \in \mathbb{R}^{d+1} : \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\mathrm{L}} = -1,\ x_{d+1} > 0 \right\},$$

with geodesic distance

$$d_K(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{K}} \operatorname{arcosh}(-\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathrm{L}}).$$

Thus, under this convention, the underlying hyperboloid does not depend on K, while all geodesic distances are rescaled by $K^{-1/2}$.

The corresponding Poincaré ball is

$$\mathbb{D}_d^{-K} = \left\{ \boldsymbol{u} \in \mathbb{R}^d : \|\boldsymbol{u}\|_2 < 1 \right\},$$

and the map

$$\pi_d(\boldsymbol{x}) = \frac{(x_1, \ldots, x_d)^\top}{1 + x_{d+1}}$$

is an isometry between the hyperboloid and Poincaré representations. In particular, for $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{D}_d^{-K}$,

$$d_{\mathbb{D}_d^{-K}}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{\sqrt{K}} \operatorname{arcosh}\left(1 + \frac{2\|\boldsymbol{u} - \boldsymbol{v}\|_2^2}{(1 - \|\boldsymbol{u}\|_2^2)(1 - \|\boldsymbol{v}\|_2^2)}\right).$$

The two-dimensional formulation in Section 2.1 is obtained by setting $d = 2$.

Volume growth. Let ω_{d-1} denote the surface area of the Euclidean unit sphere in $\mathbb{R}^d$. The Riemannian volume of a radius-r ball in $\mathbb{H}_d^{-K}$ is

$$\mathrm{Vol}_{d,K}(r) = \omega_{d-1} \int_0^r \left\{ \frac{\sinh(\sqrt{K}\, t)}{\sqrt{K}} \right\}^{d-1} \mathrm{d}t.$$

Consequently, for fixed d and K,

$$\mathrm{Vol}_{d,K}(r) = \Theta_{d,K}\left(e^{(d-1)\sqrt{K}\, r}\right) \qquad \text{as } r \to \infty,$$

whereas the volume of a radius-r Euclidean ball grows as r^d. For $d = 2$, the formula reduces to

$$\mathrm{Vol}_{2,K}(r) = \frac{2\pi}{K}\left\{\cosh(\sqrt{K}\, r) - 1\right\}.$$

This exponential growth provides room for representing branching configurations in a fixed intrinsic dimension.

Metric distortion. For an injective map g from a finite metric space $(\mathcal{X}, d_{\mathcal{X}})$ into $(\mathcal{Y}, d_{\mathcal{Y}})$, define

$$\mathrm{Lip}(g) = \sup_{x \neq y} \frac{d_{\mathcal{Y}}\{g(x), g(y)\}}{d_{\mathcal{X}}(x, y)}, \qquad \mathrm{coLip}(g) = \inf_{x \neq y} \frac{d_{\mathcal{Y}}\{g(x), g(y)\}}{d_{\mathcal{X}}(x, y)}.$$

Its bi-Lipschitz distortion is

$$D(g) = \frac{\mathrm{Lip}(g)}{\mathrm{coLip}(g)} \geq 1.$$

This quantity is invariant to a common rescaling of all target-space distances.

Let $T_{b,L}$ denote the rooted tree of depth L in which every vertex at depths $0, \dots, L-1$ has exactly b children. We equip its vertex set with the unit-edge shortest-path metric d_T. The tree has b^L leaves and

$$N_{b,L} = \frac{b^{L+1} - 1}{b - 1}$$

vertices.

Proposition A.1 (Hyperbolic versus Euclidean representations of branching trees)**.** *Let $b \geq 2$ and $L \geq 1$.*

For every $K > 0$ and $\varepsilon > 0$, there exist a scale $s = s(b, L, K, \varepsilon) > 0$ and an injective map $\psi : T_{b,L} \to \mathbb{H}_2^{-K}$ such that

$$\frac{s}{1+\varepsilon} d_T(u, v) \leq d_K\{\psi(u), \psi(v)\} \leq s\, d_T(u, v), \qquad u, v \in T_{b,L}. \tag{2}$$

In particular, $D(\psi) \leq 1 + \varepsilon$, independently of the depth L.

In contrast, for every $q \geq 1$ and every injective map $\phi : T_{b,L} \to \mathbb{R}^q$,

$$D(\phi) \geq \max\left\{1,\ \frac{b^{L/q} - 1}{L}\right\}. \tag{3}$$

Consequently, a Euclidean representation with distortion at most $D_0 \geq 1$ requires

$$q \geq \frac{L \log b}{\log(1 + D_0 L)}. \tag{4}$$

For fixed b and D_0, this implies $q = \Omega\{\log(N_{b,L}) / \log\log(N_{b,L})\}$ as $L \to \infty$.

Proposition A.1 formalizes the geometric expressivity discussed in Section 2.1. A two-dimensional hyperbolic space represents branching tree metrics with distortion arbitrarily close to one, regardless of tree depth. By contrast, any fixed-dimensional Euclidean representation incurs distortion diverging with depth; maintaining a fixed distortion requires the Euclidean dimension to increase with the tree size. The proposition concerns multiplicative distortion rather than exact isometry, and the global scale s is allowed to depend on the tree and the desired distortion level.

B Implementation and experimental details

This section provides implementation and experimental details for H2G and the competing baseline methods. We describe the likelihood-based optimization and initialization procedures, the two latent-distribution estimators used in H2G-EB and H2G-RD, baseline implementations and calibration, the evaluation metrics, and the computational resources used in the experiments.

B.1 Model training and optimization

We provide additional implementation details for the likelihood-based estimation procedure in Algorithm 1. Since the resulting optimization problem is nonconvex and is solved by alternating gradient-based updates, we do not assume that the algorithm attains the global maximum of the likelihood. Accordingly, throughout the implementation and experimental discussion, we refer to the fitted solution as an approximate maximum-likelihood estimate (approximate MLE). For each candidate curvature K, we initialize $(\boldsymbol{\alpha}, \boldsymbol{F}, \boldsymbol{Z})$ as described in Appendix B.2 and optimize the negative log-likelihood $\mathcal{L}(K, \boldsymbol{\alpha}, \boldsymbol{F}, \boldsymbol{Z})$ by alternating between manifold gradient descent updates of the latent embeddings $(\boldsymbol{F}, \boldsymbol{Z})$ and Euclidean gradient descent updates of the node effects $\boldsymbol{\alpha}$. We first describe this alternating optimization procedure and the MGD update, and then provide details on curvature selection.

Alternating optimization. For each fixed candidate curvature K, we optimize the node effects $\boldsymbol{\alpha}$ and latent embeddings $(\boldsymbol{F}, \boldsymbol{Z})$ by alternating Euclidean gradient descent and manifold gradient descent, as summarized in Algorithm 1. At outer iteration t, the MGD subroutine updates $(\boldsymbol{F}^{(t)}, \boldsymbol{Z}^{(t)})$ to $(\boldsymbol{F}^{(t+1)}, \boldsymbol{Z}^{(t+1)})$ while holding the current node effects fixed, after which $\boldsymbol{\alpha}$ is updated by

Algorithm 2 Synchronous MGD update for latent embeddings

Require: Current embeddings $\boldsymbol{F}^{(t)} = (\boldsymbol{f}_1^{(t)}, \ldots, \boldsymbol{f}_m^{(t)})^\top$ and $\boldsymbol{Z}^{(t)} = (\boldsymbol{z}_1^{(t)}, \ldots, \boldsymbol{z}_n^{(t)})^\top$; node effects $\boldsymbol{\alpha}$; fixed curvature K; step sizes η_F, η_Z.
Ensure: Updated embeddings $\boldsymbol{F}^{(t+1)}, \boldsymbol{Z}^{(t+1)}$.
1: Compute the ambient gradients $\boldsymbol{G}_F^{(t)} = \nabla_{\boldsymbol{F}} \mathcal{L}(K, \boldsymbol{\alpha}, \boldsymbol{F}^{(t)}, \boldsymbol{Z}^{(t)})$ and $\boldsymbol{G}_Z^{(t)} = \nabla_{\boldsymbol{Z}} \mathcal{L}(K, \boldsymbol{\alpha}, \boldsymbol{F}^{(t)}, \boldsymbol{Z}^{(t)})$.
2: **for** $j = 1, \ldots, m$ **do**
3: $\quad \boldsymbol{g}_{F,j}^{(t)} = \mathrm{proj}_{\boldsymbol{f}_j^{(t)}} \Big(\boldsymbol{\Lambda} (\boldsymbol{G}_F^{(t)})_{j,:}^\top \Big)$.
4: **end for**
5: **for** $i = 1, \ldots, n$ **do**
6: $\quad \boldsymbol{g}_{Z,i}^{(t)} = \mathrm{proj}_{\boldsymbol{z}_i^{(t)}} \Big(\boldsymbol{\Lambda} (\boldsymbol{G}_Z^{(t)})_{i,:}^\top \Big)$.
7: **end for**
8: $\boldsymbol{f}_j^{(t+1)} = \exp_{\boldsymbol{f}_j^{(t)}} \Big(-\eta_F \boldsymbol{g}_{F,j}^{(t)} \Big)$ for all $j = 1, \ldots, m$, and $\boldsymbol{z}_i^{(t+1)} = \exp_{\boldsymbol{z}_i^{(t)}} \Big(-\eta_Z \boldsymbol{g}_{Z,i}^{(t)} \Big)$ for all $i = 1, \ldots, n$.
9: **return** $\boldsymbol{F}^{(t+1)}, \boldsymbol{Z}^{(t+1)}$.

Euclidean gradient descent. Since the negative log-likelihood is summed over all hyperlink–node pairs, we account for the number of observations contributing to each parameter vector through the learning rates. In particular, each hyperlink embedding $\boldsymbol{f}_j$ aggregates contributions from n nodes, whereas each node embedding $\boldsymbol{z}_i$ and node effect α_i aggregates contributions from m hyperlinks. We therefore absorb the corresponding factors $1/n$ and $1/m$ into the learning rates rather than explicitly normalizing the gradients. The hyperlink and node embeddings are updated synchronously by MGD, while the node effects $\boldsymbol{\alpha}$ are updated by standard Euclidean gradient descent. For each fixed K, the iterative updates stop when the loss stabilizes according to the early-stopping criterion, or when a maximum number of 10^5 iterations is reached. For all real-world datasets, we use $\eta_F = 1/n$ and $\eta_Z = \eta_\alpha = 1/m$.

Manifold Gradient Descent. Following the MGD update of Li et al. (2026), which builds on Riemannian optimization in the Lorentz model (Nickel & Kiela, 2018), we adapt the procedure from the graph setting with a single set of node embeddings to our hypergraph setting with separate hyperlink and node embeddings, both optimized directly on the hyperboloid manifold. Algorithm 2 applies the ambient gradients to the two embedding blocks synchronously. For each current embedding $\boldsymbol{x} \in \mathbb{H}_2^{-K}$, we first convert the ambient Euclidean gradient to the Lorentz steepest-descent direction and project it onto the tangent space

$$T_{\boldsymbol{x}} \mathbb{H}_2^{-K} := \{ \boldsymbol{g} \in \mathbb{R}^3 : \langle \boldsymbol{g}, \boldsymbol{x} \rangle_{\mathrm{L}} = 0 \},$$

using the Lorentz-orthogonal projection

$$\mathrm{proj}_{\boldsymbol{x}}(\boldsymbol{g}) := \boldsymbol{g} + \langle \boldsymbol{g}, \boldsymbol{x} \rangle_{\mathrm{L}} \, \boldsymbol{x}, \qquad \boldsymbol{g} \in \mathbb{R}^3.$$

The resulting tangent direction is then mapped back to the hyperboloid along the corresponding geodesic through the exponential map. For $\boldsymbol{g} \in T_{\boldsymbol{x}} \mathbb{H}_2^{-K}$, define

$$\|\boldsymbol{g}\|_{\mathcal{L}} := \sqrt{\langle \boldsymbol{g}, \boldsymbol{g} \rangle_{\mathrm{L}}},$$

and

$$\exp_{\boldsymbol{x}}(\boldsymbol{g}) := \cosh(\|\boldsymbol{g}\|_{\mathcal{L}}) \, \boldsymbol{x} + \frac{\sinh(\|\boldsymbol{g}\|_{\mathcal{L}})}{\|\boldsymbol{g}\|_{\mathcal{L}}} \boldsymbol{g}.$$

Curvature selection. Unless otherwise specified, we use a coarse-to-fine grid search over the curvature, where we first evaluate $\mathcal{K}_{\mathrm{coarse}} = \{0.1, 0.3, 1.0\}$ and then refine the search using a logarithmically spaced grid around the most promising coarse-scale region. Within the embedding stage, Algorithm 1 selects curvature using the attained in-sample training log-likelihood. When a validation set is available, curvature can alternatively be selected by validation performance. In our simulations, both criteria recover the ground-truth curvature, as discussed in Appendix C.2.5. For

real-data experiments, we select curvature by the mean validation Energy distance across random seeds. We use validation Energy distance because it compares the full hyperlink distributions rather than only marginal or pairwise summaries, requires no kernel or bandwidth choice, and unlike Δ_{H_d}, is not directly controlled by calibration. The same criterion is used to select DDE's embedding dimension. Detailed definitions of the evaluation metrics are provided in Appendix B.6.

B.2 Initialization of model parameters

Since the optimization problem is nonconvex, initialization plays an important role in obtaining a stable fitted solution. We initialize the node effects from empirical node degrees and construct the latent hyperlink and node embeddings through a USVT-based probability estimate, rectangular multidimensional scaling, and a Euclidean likelihood refinement before mapping the resulting configuration to the target hyperbolic space.

Node-effect initialization. We initialize each node effect from its empirical degree as

$$\alpha_i^{(0)} = \log(\max\{s_\alpha d_i, \epsilon\}), \qquad d_i = \sum_{j=1}^{m} H_{ji},$$

where d_i is the empirical degree of node i, $s_\alpha > 0$ is a scaling constant, and ϵ is a small numerical safeguard. This initialization assigns larger node effects to nodes that occur more frequently in the observed hyperlinks, consistent with the role of α_i in controlling node-specific inclusion propensity. We use $s_\alpha = 5 \times 10^{-3}$ in all experiments. The resulting node-wise effects are subsequently kept fixed during latent-embedding initialization and enter both the inversion from estimated incidence probabilities to hyperlink–node distances and the Euclidean refinement.

Latent embedding initialization. We initialize the latent embeddings $\mathcal{F}_m, \mathcal{Z}_n$ using a Universal Singular Value Thresholding (USVT)-based procedure (Chatterjee, 2015), followed by a 20-dimensional Euclidean refinement inspired by Ma et al. (2020) and Li et al. (2026). In contrast to the graph setting considered in these works, our incidence matrix is rectangular and our model contains separate hyperlink and node embeddings together with node-specific effects. We therefore adapt the initialization to recover hyperlink–node distances using the initialized $\boldsymbol{\alpha}^{(0)}$.

Given the rank-r approximation $\widehat{\boldsymbol{P}}$ of $\boldsymbol{H}$, we first clip its entries to $[10^{-12}, 1-10^{-12}]$ before applying the logit transformation. Since

$$\mathrm{logit}(p_{ji}) = \alpha_i - d_K(\boldsymbol{f}_j, \boldsymbol{z}_i),$$

we estimate the hyperlink–node distances by

$$\widehat{D}_{ji} = \alpha_i^{(0)} - \mathrm{logit}(\widehat{P}_{ji}),$$

and clip the resulting values to $[0, 6]$ to ensure valid and numerically stable distances. We then apply rectangular multidimensional scaling (MDS) to $\widehat{\boldsymbol{D}}$ to obtain q-dimensional Euclidean hyperlink and node embeddings. Here, r controls only the rank of the USVT probability estimate, whereas q denotes the Euclidean embedding dimension.

For EU20, we set $q = 20$ and refine the Euclidean embeddings while holding $\boldsymbol{\alpha}^{(0)}$ fixed by minimizing

$$\mathcal{L}_{\mathrm{Euc}}(\boldsymbol{F}, \boldsymbol{Z}) = -\sum_{j=1}^{m}\sum_{i=1}^{n} \left[H_{ji} \log p_{ji} + (1 - H_{ji}) \log(1 - p_{ji})\right],$$

where

$$p_{ji} = \sigma\Big(\alpha_i^{(0)} - \|\boldsymbol{f}_j - \boldsymbol{z}_i\|_2\Big).$$

After refinement, the Euclidean configuration is converted to the target hyperbolic space through the curvature-dependent Gram construction of Li et al. (2026). We use $r = 10$ and $q = 20$ throughout. The two initialization procedures are summarized in Algorithms 3 and 4.

For the reported experiments, we use $r = 10$, $q = 20$, $T = 1000$, and $\eta = 5/\max\{m, n\}$, with early stopping when the change in the mean negative log-likelihood is at most 10^{-6}.

Algorithm 3 USVT initialization of latent embeddings

Require: Incidence matrix $\boldsymbol{H} \in \{0,1\}^{m\times n}$; initialized node effects $\boldsymbol{\alpha}^{(0)}$; curvature K; target dimension d; USVT rank r; Euclidean dimension q.
Ensure: Initial hyperbolic embeddings $\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)}$.
1: Compute $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ and set $\widehat{\boldsymbol{P}} = \boldsymbol{U}_r\boldsymbol{\Sigma}_r\boldsymbol{V}_r^\top$.
2: Clip $\widehat{P}_{ji}$ to $[10^{-12}, 1-10^{-12}]$ and set $\widehat{D}_{ji} = \alpha_i^{(0)} - \mathrm{logit}(\widehat{P}_{ji})$.
3: Clip $\widehat{D}_{ji}$ to $[0,6]$.
4: Form $\boldsymbol{B} = -\frac{1}{2}\boldsymbol{J}_m\widehat{\boldsymbol{D}}^{\circ 2}\boldsymbol{J}_n$, where $\boldsymbol{J}_m = \boldsymbol{I}_m - m^{-1}\mathbf{1}_m\mathbf{1}_m^\top$ and $\boldsymbol{J}_n = \boldsymbol{I}_n - n^{-1}\mathbf{1}_n\mathbf{1}_n^\top$.
5: Compute $\boldsymbol{B} = \boldsymbol{U}_B\boldsymbol{\Sigma}_B\boldsymbol{V}_B^\top$ and set $\boldsymbol{F} = \boldsymbol{U}_{B,q}\boldsymbol{\Sigma}_{B,q}^{1/2}$, $\boldsymbol{Z} = \boldsymbol{V}_{B,q}\boldsymbol{\Sigma}_{B,q}^{1/2}$.
6: Jointly rescale and center $(\boldsymbol{F}, \boldsymbol{Z})$.
7: Convert $(\boldsymbol{F}, \boldsymbol{Z})$ to $\mathbb{H}_d^{-K}$ using the curvature-dependent Gram construction to obtain $(\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)})$.
8: **return** $\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)}$.

Algorithm 4 USVT+EU20 initialization

Require: Incidence matrix $\boldsymbol{H}$; initialized node effects $\boldsymbol{\alpha}^{(0)}$; curvature K; target dimension d; USVT rank r; Euclidean dimension q; step size η; maximum iterations T.
Ensure: Initial hyperbolic embeddings $\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)}$.
1: Apply Steps 1–6 of Algorithm 3 with USVT rank r and Euclidean dimension q to obtain Euclidean embeddings $(\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)})$.
2: **for** $t = 0, \ldots, T-1$ **do**
3: $\quad p_{ji}^{(t)} = \sigma\Big(\alpha_i^{(0)} - \|\boldsymbol{f}_j^{(t)} - \boldsymbol{z}_i^{(t)}\|_2\Big)$.
4: $\quad \boldsymbol{F}^{(t+1)} = \boldsymbol{F}^{(t)} - \eta\nabla_{\boldsymbol{F}}\mathcal{L}_{\mathrm{Euc}}, \qquad \boldsymbol{Z}^{(t+1)} = \boldsymbol{Z}^{(t)} - \eta\nabla_{\boldsymbol{Z}}\mathcal{L}_{\mathrm{Euc}}$.
5: $\quad$ Jointly center $(\boldsymbol{F}^{(t+1)}, \boldsymbol{Z}^{(t+1)})$.
6: **end for**
7: Convert $(\boldsymbol{F}^{(T)}, \boldsymbol{Z}^{(T)})$ to $\mathbb{H}_d^{-K}$ using the curvature-dependent Gram construction to obtain $(\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)})$.
8: **return** $\boldsymbol{F}^{(0)}, \boldsymbol{Z}^{(0)}$.

B.3 Empirical bootstrap

For H2G-EB, we model the latent hyperlink distribution by the empirical distribution of the fitted hyperlink embeddings,

$$\widehat{\mathbb{P}}_{\mathbf{F}}^{\mathrm{EB}} = \frac{1}{m}\sum_{j=1}^{m}\delta_{\widehat{\mathbf{f}}_j},$$

where $\widehat{\mathbf{F}} = (\widehat{\mathbf{f}}_1, \ldots, \widehat{\mathbf{f}}_m)^\top$ is obtained from the embedding stage. To generate a latent synthetic hyperlink, we sample $\widetilde{\mathbf{f}} \sim \widehat{\mathbb{P}}_{\mathbf{F}}^{\mathrm{EB}}$, implemented by uniformly sampling an index from $\{1, \ldots, m\}$ with replacement and setting $\widetilde{\mathbf{f}} = \widehat{\mathbf{f}}_j$. Hence, the same fitted hyperlink embedding may be selected more than once across generated samples.

Given $\widetilde{\mathbf{f}}$, we keep the fitted node embeddings $\widehat{\mathbf{Z}}$, node effects $\widehat{\boldsymbol{\alpha}}$, and curvature $\widehat{K}$ fixed and decode a new binary hyperlink from the fitted probabilistic model. Specifically, for each node i, we compute

$$\widehat{p}_i(\widetilde{\mathbf{f}}) = \sigma\left(\widehat{\alpha}_i - \frac{1}{\sqrt{\widehat{K}}}\,\mathrm{arcosh}\Big(-\widetilde{\mathbf{f}}^\top\Lambda\widehat{\mathbf{z}}_i\Big)\right),$$

and independently sample

$$\widetilde{H}_i \sim \mathrm{Bernoulli}\Big(\widehat{p}_i(\widetilde{\mathbf{f}})\Big).$$

Thus, bootstrap resampling is performed in the learned latent space rather than directly on the observed binary hyperlinks. Two generated samples that reuse the same latent embedding need not produce identical hyperlinks because the Bernoulli decoder is sampled anew for each generation.

In our implementation, we generate $32m$ bootstrap samples for each fitted model using a fixed random seed for reproducibility and perform Bernoulli decoding in batches for memory efficiency. No additional latent perturbation, smoothing, or post-hoc calibration is applied to H2G-EB.

B.4 Riemannian diffusion

We use Riemannian diffusion in H2G-RD to model the distribution of learned hyperlink embeddings and generate new latent hyperlinks directly in the hyperbolic space. We implement the diffusion model using the Riemannian score-SDE framework of Song et al. (2020), together with the geometric operations provided by `Geomstats` (Miolane et al., 2020). Since our hyperbolic embeddings use a different coordinate and curvature convention from the underlying implementation, we introduce a lightweight geometric adapter to align the manifold representation and metric with those used in H2G. This enables score estimation and reverse-time sampling directly in the learned hyperbolic latent space.

Forward Riemannian diffusion. Conditioning on the estimated hyperlink embeddings $\widehat{\mathcal{F}}_m = \{\widehat{\boldsymbol{f}}_j\}_{j=1}^m$ and fitted curvature $\widehat{K}$, we define the empirical latent distribution $\widehat{\mu}_m = m^{-1}\sum_{j=1}^m \delta_{\widehat{\boldsymbol{f}}_j}$ on $\mathcal{M}_{\widehat{K}} = \mathbb{H}_2^{-\widehat{K}}$. Starting from $X_0 \sim \widehat{\mu}_m$, H2G-RD evolves the latent embeddings according to the Riemannian Langevin diffusion

$$\mathrm{d}X_t = -\frac{1}{2}\beta(t)\nabla_{\mathcal{M}} U(X_t)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}B_t^{\mathcal{M}},$$

where $B_t^{\mathcal{M}}$ denotes Brownian motion on $\mathcal{M}_{\widehat{K}}$ and U specifies the limiting reference distribution $\pi_U(\mathrm{d}x) \propto e^{-U(x)}\mathrm{vol}_{\widehat{K}}(\mathrm{d}x)$. We denote the law of X_t by μ_t and its density with respect to the Riemannian volume measure by p_t. The forward process progressively perturbs the empirical embedding distribution toward the reference distribution, from which the reverse-time process is initialized.

We implement this process using the Langevin SDE in the Riemannian score-SDE framework together with the hyperbolic operations in `Geomstats`. Since `Geomstats` represents the Lorentz model using unit-hyperboloid coordinates, whereas H2G represents curvature through $\widehat{K}$, our geometric adapter retains the unit coordinates and rescales the metric as $g_{\widehat{K}} = g_1/\widehat{K}$. Consequently, geodesic distances and tangent-vector norms are scaled by $1/\sqrt{\widehat{K}}$ relative to their unit-curvature counterparts. The adapter also converts between the $[\mathrm{spatial}, \mathrm{time}]$ Lorentz convention used by H2G and the $[\mathrm{time}, \mathrm{spatial}]$ convention used internally by `Geomstats`. For numerical conditioning, the fitted hyperlink embeddings are first mapped to the tangent space at the origin, centered and rescaled using a common global tangent-space scale, and mapped back to the hyperboloid; the inverse transformation is applied after sampling. We use a linear diffusion schedule

$$\beta(t) = \beta_0 + \frac{t - t_0}{t_f - t_0}(\beta_f - \beta_0), \qquad t \in [t_0, t_f].$$

The values of the diffusion horizon, noise levels, reference scale, and discretization parameters used in the experiments are reported with the remaining implementation details below.

Denoising score matching. We train the score network using denoising score matching (DSM), which replaces the intractable marginal score of the perturbed distribution with the conditional score of the forward transition kernel. Let $p_{t|0}(x_t \mid x_0)$ denote the transition density with respect to Riemannian volume. For $x_0 \sim \widehat{\mu}_m$ and $x_t \sim p_{t|0}(\cdot \mid x_0)$, we minimize the likelihood-weighted DSM objective

$$\mathcal{L}_{\mathrm{DSM}}(\theta) = \mathbb{E}_{t,x_0,x_t}\left[\beta(t)\left\|s_\theta(x_t,t) - \nabla_{x_t}\log p_{t|0}(x_t \mid x_0)\right\|_{g_{\widehat{K}}}^2\right].$$

Here, $\|\cdot\|_{g_{\widehat{K}}}$ is the norm induced by the Riemannian metric at x_t. The weighting $\beta(t)$ is the squared diffusion coefficient of the forward Langevin SDE. The conditional transition score provides a tractable denoising target, while its conditional expectation given $X_t = x_t$ equals the marginal score,

$$\mathbb{E}\big[\nabla_{x_t}\log p_{t|0}(x_t \mid X_0) \mid X_t = x_t\big] = \nabla_{\mathcal{M}}\log p_t(x_t),$$

Table 3: Dataset-specific hyperparameters for H2G-RD. Hyperparameters shared across datasets are reported in the text.

	MIMIC-III	MathOverflow-Answers	CORUM
$\widehat{K}$	1.0	0.3	1.0
Hidden dimension	128	128	64
Hidden layers	6	8	4
Diffusion batch size	256	128	128
Learning rate ($\times 10^{-5}$)	4	5	5
Diffusion steps	10	20	20
Sampling steps	30	5	5
Reference scale	Adaptive	1.0	0.5

under the usual regularity conditions.

For the hyperbolic Langevin process, the exact transition score is not evaluated explicitly. Instead, following the Riemannian score-SDE implementation, we set `n_max=-1` in the DSM loss and use its Varadhan approximation to the conditional transition score. Thus, $\nabla_{x_t} \log p_{t|0}(x_t \mid x_0)$ in the objective above denotes the corresponding approximate target during optimization, whereas the population quantity required by the reverse process remains the marginal score $\nabla_{\mathcal{M}} \log p_t(x_t)$. The score-estimation error considered in Appendix E.4.2 therefore also covers the error introduced by this approximate transition-score target. In implementation, diffusion times are sampled away from the singular endpoint using a cutoff $\varepsilon = 10^{-3}$.

Score network. We parameterize the score field $s_\theta(x, t)$ using a multilayer perceptron with SiLU activations. The network takes the extrinsic hyperboloid coordinate x_t together with the diffusion time t as input and outputs an ambient vector of the same dimension. This output is projected onto the tangent space $T_{x_t}\mathbb{H}_2^{-\widehat{K}}$, so that $s_\theta(x_t, t)$ defines a valid tangent vector field on the manifold. We use the scalar diffusion time directly as the default temporal input; additional sinusoidal time features are disabled in the reported experiments. The hidden dimension and number of layers are dataset dependent and are summarized in Table 3.

Optimization and sampling. We optimize the score network with Adam and maintain an exponential moving average of the model parameters with decay rate 0.999. The learning rate is linearly warmed up for the first 500 optimization steps and subsequently decayed using a cosine schedule. At generation time, we use the averaged parameters to construct the reverse-time diffusion and initialize the process from the limiting distribution of the forward Langevin SDE. We numerically simulate the reverse process with the geodesic random-walk (GRW) predictor using the dataset-specific number of sampling steps reported in Table 3. The generated samples are then converted back to the Lorentz coordinate convention used by H2G, projected onto the hyperboloid, and transformed back to the original latent scale by inverting the tangent-space normalization before hyperlink decoding.

Hyperparameter settings. For the real-world experiments, we train all Riemannian diffusion models for 2500 optimization steps using a linear noise schedule with $\beta_0 = 0.001$ and $\beta_f = 3.0$. We use Adam with a 500-step linear warm-up followed by cosine learning-rate decay, enable tangent-space normalization, and do not use additional sinusoidal time features. Dataset-specific architecture and discretization parameters are summarized in Table 3. Hyperparameters for the simulated experiments are reported together with the simulation setup in Appendix C.1. Hyperparameters are selected using the validation split only, while the held-out test split is reserved for final evaluation.

B.5 Baseline implementations

We provide implementation details for the baseline methods introduced in Section 4.1. Unless otherwise specified, all baselines use the same data splits, number of generated synthetic hyperlinks, and evaluation protocol as H2G. For baselines with continuous outputs, we apply the post-hoc calibration procedure described at the end of this section to obtain binary hyperlink vectors.

DDE Denoising Diffused Embeddings (DDE) (Wu et al., 2025) generates hypergraphs by first learning a low-dimensional Euclidean representation of the observed hyperlinks and then modeling the distribution of the learned hyperlink embeddings with a score-based diffusion model. New embeddings generated by the diffusion model are subsequently decoded into hyperlinks through the fitted conditional Bernoulli model. We follow the linear Euclidean embedding formulation described in the original paper. Following the dimension choices reported in the original DDE real-data experiments, we consider $d \in \{2, 4, 8\}$ and select the embedding dimension using the validation set. The hyperlink embeddings, node embeddings, and node-specific degree parameters are jointly estimated by maximizing the conditional Bernoulli likelihood of the training incidence matrix using projected gradient ascent. A score-based diffusion model is then trained on the estimated hyperlink embeddings to generate new embeddings in the latent space, from which binary hyperlinks are sampled using the fitted conditional Bernoulli probabilities.

Despite the apparent similarity in the overall embedding–diffusion–decoding pipeline, our implementation differs fundamentally from DDE throughout the entire procedure. DDE operates in a Euclidean latent space with a linear embedding model, whereas H2G represents both node and hyperlink embeddings in a hyperbolic space whose geometry is coupled with the learned curvature K. Consequently, embedding estimation requires Riemannian optimization rather than Euclidean projected gradient updates, and the generative stage is formulated as a Riemannian score-based diffusion process that evolves directly on the learned hyperbolic manifold. Thus, although both methods build on score-based diffusion, their embedding optimization, diffusion dynamics, and sampling procedures are algorithmically distinct.

Gau-Diff. Gaussian Diffusion (Gau-Diff) applies score-based diffusion (Song et al., 2020) directly to the ambient hyperlink space by treating binary incidence vectors as continuous observations in $\mathbb{R}^n$. Gau-Diff applies this framework directly to the ambient hyperlink space rather than first learning a low-dimensional representation. Specifically, we encode each hyperlink as an n-dimensional binary incidence vector and treat these vectors as continuous observations in $\mathbb{R}^n$. A Gaussian score-based diffusion model is trained directly on these vectors, and the reverse diffusion process generates continuous n-dimensional outputs. We convert these outputs into binary hyperlinks using the post-hoc calibration procedure described at the end of this subsection.

TabDDPM. Tabular Denoising Diffusion Probabilistic Models (TabDDPM) (Kotelnikov et al., 2023) is a diffusion-based generative model for tabular data that represents each observation as a collection of heterogeneous features and models their joint distribution through a denoising diffusion process. We adapt this framework to hypergraph generation by viewing each hyperlink as one tabular observation and its n-dimensional incidence vector as the corresponding feature vector, where each coordinate indicates whether a node is contained in the hyperlink. In principle, the binary node indicators can be modeled using TabDDPM's multinomial diffusion by treating each coordinate as a binary categorical feature. However, in our high-dimensional incidence setting, this formulation was numerically unstable: the multinomial training objective became `NaN`, preventing reliable optimization and resulting in degenerate samples. A similar numerical instability for datasets with large numbers of binary categorical variables has also been reported in the official TabDDPM repository[2]. This behavior may be related to numerical instability in the log-space categorical posterior and variational objective when evaluated jointly over hundreds to thousands of binary variables, a substantially higher-dimensional regime than the categorical settings considered in the original TabDDPM experiments.

We therefore use a continuous Gaussian diffusion over the incidence coordinates, which remained numerically stable in our experiments. Unlike Gau-Diff, which uses a continuous-time score-based diffusion formulation, this adaptation follows the discrete-time DDPM parameterization and directly predicts the Gaussian noise added at each diffusion step. Specifically, the binary incidence vectors are linearly rescaled from $0, 1^n$ to $-1, 1^n$ before training. We use a standard Gaussian DDPM with $T = 500$ diffusion steps and a linear variance schedule from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. The full reverse DDPM process generates continuous n-dimensional incidence scores, which are converted into binary hyperlinks using the post-hoc calibration procedure described at the end of this subsection.

[2] `https://github.com/yandex-research/tab-ddpm/issues/40`

Ber-Diff. Bernoulli Diffusion (Ber-Diff) performs diffusion directly in the discrete binary state space, avoiding the continuous relaxation used by Gaussian diffusion models. Following the Bernoulli diffusion construction used in Wu et al. (2025), the forward process progressively randomizes each coordinate of an n-dimensional incidence vector through bit-flipping transitions, while a neural network parameterizes the corresponding reverse transitions. We use $T = 2000$ discrete flipping steps following their experimental setup. The reverse process directly produces binary incidence vectors and therefore requires no post-hoc calibration. Due to the high memory cost of discrete diffusion directly in the full n-dimensional binary incidence space, which requires repeatedly processing high-dimensional intermediate states and reverse-transition predictions over many diffusion steps, Ber-Diff is only feasible on MIMIC-III among our real-world datasets.

WGAN. Generative adversarial networks (GANs) learn a data distribution through an adversarial game between a generator and a discriminator, while Wasserstein GANs (WGANs) replace the original adversarial objective with the Wasserstein distance to improve training stability (Arjovsky et al., 2017). We use the gradient-penalized variant WGAN-GP (Gulrajani et al., 2017), which enforces the Lipschitz constraint through a penalty on the critic gradients. Each hyperlink is represented by its n-dimensional binary incidence vector, and the generator maps a standard Gaussian latent vector to a continuous n-dimensional output. We train the critic five times per generator update and use a gradient-penalty weight of $\lambda_{\mathrm{GP}} = 10$. Both networks are optimized with Adam for 2,000 epochs with batch size 128. Since the generator outputs continuous node-wise scores, we convert them into binary hyperlinks using the post-hoc calibration procedure described at the end of this subsection.

VAE. Variational autoencoders (VAEs) (Kingma & Welling, 2013) combine an autoencoding architecture with probabilistic latent-variable modeling. An encoder approximates the posterior distribution of a latent variable for each observation, while a decoder reconstructs the observation from the latent code; training maximizes the evidence lower bound, balancing reconstruction accuracy against a Kullback–Leibler regularization toward a standard Gaussian prior. We represent each hyperlink as its n-dimensional binary incidence vector and train the VAE using binary cross-entropy reconstruction loss and the standard KL regularization. We treat the VAE bottleneck dimension as a method-specific hyperparameter and select it according to validation performance; a bottleneck dimension of 2 gives the best validation results and is used for all reported VAE experiments. This bottleneck dimension controls the capacity of the autoencoding latent code and is not directly comparable to the geometric embedding dimensions used by H2G or DDE. At generation time, latent codes are sampled from a standard Gaussian prior, decoded into node-wise probabilities, and thresholded at 0.5 to obtain binary hyperlinks.

HyperLap. HyperLap (Lee et al., 2021) is a hypergraph-specific generative model designed to reproduce higher-order structural patterns through a multilevel partition of the node set. Hyperlinks are generated within groups at different hierarchy levels, allowing the model to control the extent to which nodes repeatedly co-occur across hyperlinks. We use the official implementation with the automatic weight-selection procedure of HyperLap+. For each hyperlink, HyperLap selects a hierarchy level and a group within that level and samples nodes according to their observed degrees until the target hyperlink size is reached. The hierarchy-level weights are automatically selected by minimizing the discrepancy between the hyperlink-homogeneity distributions of the training and generated hypergraphs. We otherwise use the default settings of the official implementation.

HyperPLR. Hypergraph Generation through Projection, Learning, and Reconstruction (HyperPLR) (Wen & Yu, 2025) generates hypergraphs through a projection–learning–reconstruction strategy. Rather than modeling hyperlinks directly, it projects the observed hypergraph onto pairwise graphs, learns a generative model for the projected structure, and subsequently reconstructs higher-order hyperlinks from the generated projection. We use the official implementation with its default settings. HyperPLR first constructs weighted and unweighted graph projections of the training hypergraph, then generates a new graph structure and predicts its edge weights, and finally reconstructs the hypergraph using its Greedy Weighted Cover procedure. The resulting selected cliques are treated as generated hyperlinks. The official implementation of HyperPLR runs out of GPU memory on all real-world datasets at our data scale, as its pairwise projection can contain a large number of node pairs and the subsequent graph-generation, edge-weight prediction, and clique-reconstruction

stages operate on this representation. Nevertheless, it remains feasible on the simulated dataset and is included in the corresponding generative-performance comparison in Appendix C.3.

Calibration. Gau-Diff, TabDDPM, and WGAN generate continuous vectors and therefore require post-hoc calibration to obtain binary hyperlinks. Following the calibration procedure of Ma et al. (2026), let $\tilde{\mathbf{y}}_j = (\tilde{y}_{j1}, \ldots, \tilde{y}_{jn})^\top \in \mathbb{R}^n$, $j = 1, \ldots, \tilde{m}$, denote the generated continuous vectors, and let $\mathbf{H}^{\mathrm{tr}} \in \{0,1\}^{m_{\mathrm{tr}} \times n}$ denote the incidence matrix of the training hypergraph. For each node i, we choose a threshold τ_i and define

$$\hat{y}_{ji} = \mathbf{1}\{\tilde{y}_{ji} \geq \tau_i\},$$

such that

$$\frac{1}{\tilde{m}} \sum_{j=1}^{\tilde{m}} \hat{y}_{ji} = \frac{1}{m_{\mathrm{tr}}} \sum_{j=1}^{m_{\mathrm{tr}}} H^{\mathrm{tr}}_{ji}.$$

Thus, calibration preserves the marginal frequency of each node in the training hypergraph, while converting the continuous outputs into binary hyperlink vectors. Since we generate $\tilde{m} = 32 m_{\mathrm{tr}}$ hyperlinks, this is equivalent to matching the generated node degrees to 32 times their corresponding training degrees. Because calibration explicitly matches the empirical node marginals of the generated hyperlinks to those of the training hypergraph, Δ_{H_d} is largely controlled by construction for calibrated methods. We therefore distinguish calibrated from uncalibrated methods when interpreting this metric.

B.6 Fidelity metrics

This subsection provides the definitions and implementation details of the fidelity metrics used throughout the experiments. We first define the Jaccard distance underlying our distributional comparisons, then describe the Energy distance and MMD^2, and finally define the first- and second-order node-statistic errors Δ_{H_d} and Δ_{H_v}.

Jaccard distance of hyperlinks. Each hyperlink is represented as a binary vector $x \in \{0,1\}^p$, where $x_i = 1$ indicates that node i is contained in the hyperlink. For two binary hyperlink vectors x and y, we use the Jaccard distance (Kosub, 2019),

$$d_J(x,y) = 1 - \frac{\sum_{i=1}^p x_i y_i}{\sum_{i=1}^p \mathbf{1}\{x_i + y_i > 0\}},$$

which is equivalently

$$d_J(x,y) = 1 - \frac{|S_x \cap S_y|}{|S_x \cup S_y|},$$

where $S_x = \{i : x_i = 1\}$ and $S_y = \{i : y_i = 1\}$ denote the corresponding active-node sets. We set $d_J(x,y) = 0$ when both sets are empty. The Jaccard distance is particularly suitable for sparse binary hyperlinks because it normalizes disagreement by the union of active nodes and ignores coordinates that are absent from both hyperlinks. Consequently, missing or spurious nodes can induce substantial relative changes in the distance when hyperlinks contain only a small number of active nodes, making the metric sensitive to failures in recovering selective co-membership patterns.

Energy distance. We use the Energy distance (Székely & Rizzo, 2013) to compare the distributions of real and generated hyperlinks, with the Jaccard distance as the ground metric. For $X_1, \ldots, X_n \sim P$ and $Y_1, \ldots, Y_m \sim Q$, its empirical estimator is

$$\widehat{\mathcal{E}}(P,Q) = \frac{2}{nm} \sum_{i=1}^n \sum_{j=1}^m d_J(X_i, Y_j) - \frac{1}{n(n-1)} \sum_{i \neq i'} d_J(X_i, X_{i'}) - \frac{1}{m(m-1)} \sum_{j \neq j'} d_J(Y_j, Y_{j'}).$$

The first term measures the average discrepancy between real and generated hyperlinks, while the latter two terms account for the within-distribution variability of the real and generated samples, respectively. Smaller values indicate greater similarity between the two hyperlink distributions.

Squared maximum mean discrepancy (MMD^2). We further measure distributional discrepancy using the squared maximum mean discrepancy (Gretton et al., 2012),

$$\mathrm{MMD}^2(P,Q) = \mathbb{E}[k(X,X')] + \mathbb{E}[k(Y,Y')] - 2\mathbb{E}[k(X,Y)],$$

where $X, X' \sim P$ and $Y, Y' \sim Q$ are independent. We use a Jaccard-based RBF kernel

$$k_\gamma(x,y) = \exp\{-\gamma d_J(x,y)\}.$$

The base inverse bandwidth γ_0 is determined by the median heuristic on the training hyperlinks,

$$\gamma_0 = \frac{1}{\mathrm{median}_{i<i'}\, d_J\big(X_i^{\mathrm{train}}, X_{i'}^{\mathrm{train}}\big)},$$

where zero pairwise distances are excluded from the median. To reduce sensitivity to a single bandwidth choice, we use three scales,

$$\gamma_\ell \in \{0.5\gamma_0, \gamma_0, 2\gamma_0\},$$

and report the average of the corresponding empirical MMD^2 values,

$$\widehat{\mathrm{MMD}}^2 = \frac{1}{3}\sum_{\ell=1}^{3}\left[\frac{1}{n^2}\sum_{i,i'=1}^{n} k_{\gamma_\ell}(X_i,X_{i'}) + \frac{1}{m^2}\sum_{j,j'=1}^{m} k_{\gamma_\ell}(Y_j,Y_{j'}) - \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} k_{\gamma_\ell}(X_i,Y_j)\right].$$

Smaller MMD^2 values indicate closer agreement between the real and generated hyperlink distributions.

Means and covariances. Following DDE (Wu et al., 2025), we compare the first- and second-order node-level statistics of the real and generated hyperlinks. Let $Y = (Y_1, \ldots, Y_n)^\top$ denote a binary hyperlink vector and $\widetilde{Y}$ its generated counterpart. We define the RMSE of node frequencies as

$$\Delta_{H_d} = \sqrt{\frac{1}{n}\sum_{i\in[n]}\Big(\mathbb{E}(Y_i) - \widehat{\mathbb{E}}(\widetilde{Y}_i)\Big)^2}.$$

Let Σ and $\widetilde{\Sigma}$ denote the covariance matrices of the real and generated hyperlink vectors, respectively. We define Δ_{H_v} as the RMSE between the corresponding covariance entries,

$$\Delta_{H_v} = \sqrt{\frac{1}{n(n+1)/2}\sum_{i\le j}\Big(\Sigma_{ij} - \widetilde{\Sigma}_{ij}\Big)^2}.$$

Thus, Δ_{H_d} measures discrepancies in marginal node frequencies, while Δ_{H_v} measures discrepancies in the second-order node covariance structure. Smaller values indicate closer agreement between the real and generated hypergraphs.

B.7 Computational cost and resources

Real-data experiments are conducted on a server equipped with an NVIDIA GeForce RTX 4090 GPU with 24 GB of memory, while simulation experiments are run on a local machine equipped with an NVIDIA GeForce RTX 2050 GPU with 4 GB of memory. On the real-world dataset of MIMIC-III, for a single fixed curvature K, H2G-RD requires 3 hours of wall-clock time and reaches a peak GPU memory usage of approximately 888 MB.

C Simulation study and results

We conduct simulation studies to evaluate H2G from two complementary perspectives. First, we examine whether the proposed maximum-likelihood estimation procedure can recover the ground-truth latent parameters under controlled settings. Second, we assess generative performance by comparing synthetic hyperlinks produced by H2G and competing methods against held-out simulated data. Unless otherwise specified, all simulation settings follow the data-generation procedure described below.

C.1 Simulation setup

Simulated data generation. We generate simulated hypergraphs directly from the proposed latent hyperbolic model. Specifically, we independently sample the ground-truth hyperlink embeddings $\{\boldsymbol{f}_j\}_{j=1}^m$ and node embeddings $\{\boldsymbol{z}_i\}_{i=1}^n$ uniformly from a two-dimensional hyperbolic disk of radius $R = 3$, following the uniform hyperbolic-disk construction of Li et al. (2026). For each node, we independently draw $\alpha_i \sim \mathrm{Unif}(1.5, 2.0)$ and then sample the incidence entries conditionally independently as

$$H_{ji} \sim \mathrm{Bernoulli}(\sigma(\alpha_i - d_K(\boldsymbol{f}_j, \boldsymbol{z}_i))) .$$

Experimental settings. Unless otherwise specified, we set $n = 500$, $m = 1000$, $d = 2$, and $K = 0.5$. For generative-performance evaluation, the simulated hyperlinks are split into training, validation, and test sets with proportions $0.8/0.1/0.1$, whereas experiments evaluating parameter recovery use the full simulated hypergraph without data splitting.

Diffusion hyperparameters. For the simulated generative experiments involving H2G-RD, we use the Riemannian diffusion implementation described in Appendix B.4. We set the hidden dimension to 128, use 4 hidden layers, and train with batch size 128 for 2500 optimization steps. The forward diffusion uses 20 discretization steps and a linear noise schedule with $\beta_0 = 0.001$ and $\beta_f = 3.0$, while reverse-time sampling uses 5 steps. We set the reference scale to 0.5, enable tangent-space normalization, and do not use additional sinusoidal time features. The score network is optimized with learning rate 5×10^{-5} using a 250-step linear warm-up followed by cosine decay.

C.2 Recovery of parameters

We evaluate whether the likelihood-based embedding procedure recovers the latent parameters and geometry used to generate the simulated hypergraphs. We first specify the alignment needed to compare latent configurations and define the corresponding recovery metrics. We then examine finite-sample recovery, its behavior as the number of observed hyperlinks increases, and recovery and selection of the curvature parameter.

C.2.1 Identifiability of latent positions

Under our model, the likelihood depends on the latent positions through their hyperbolic distances. We therefore impose the standard identifiability convention that latent configurations differing only by a common hyperbolic isometry are regarded as equivalent, as is customary in latent-space models (Li et al., 2026; Wu et al., 2025). Accordingly, recovery of the latent coordinates should be assessed modulo this common isometry rather than by direct coordinate-wise comparison.

For the parameter-recovery experiments, we map both the estimated and ground-truth latent positions to the two-dimensional Poincaré disk and remove the remaining common orientation by a shared orthogonal alignment. Specifically, we use a single orthogonal transformation, corresponding to a common rotation or reflection, to best match the estimated configuration to the ground truth. The hyperlink and node embeddings are transformed jointly using the same matrix, so their relative configuration is preserved. The corresponding transformation matrix $\boldsymbol{Q}^\star$ and the aligned RMSEs are defined in Appendix C.2.2. This alignment is introduced solely to make coordinate-level recovery against the known ground truth well defined. In all experiments concerned only with generative performance, including both simulated and real-world experiments, no rotation, alignment, or other transformation is applied to the estimated latent positions before hyperlink generation.

C.2.2 Recovery metrics

We measure parameter recovery using root mean squared error (RMSE). For the node-specific effects, we report $\Delta\alpha$, the RMSE between the estimated and ground-truth parameters. For the latent positions, we first map the estimated and ground-truth hyperlink and node embeddings to the two-dimensional Poincaré disk and apply the shared orthogonal alignment introduced in Appendix C.2.1. We then report ΔF and ΔZ, the coordinate-wise RMSEs of the aligned hyperlink and node embeddings, respectively. These two quantities are computed using ordinary Euclidean distances between the two-dimensional disk coordinates, rather than hyperbolic distances.

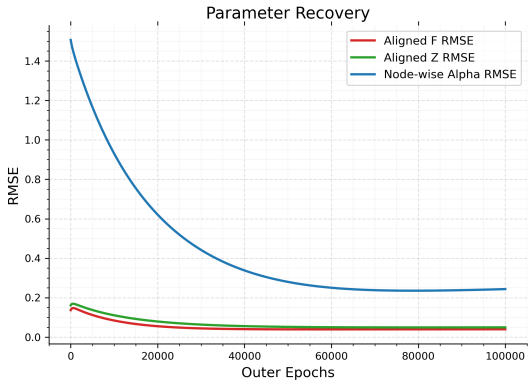

Figure 5: Finite-sample parameter recovery during optimization. We report the aligned hyperlink-position RMSE ΔF, aligned node-position RMSE ΔZ, and node-effect RMSE $\Delta\alpha$ as functions of the outer optimization epochs. The latent-position errors are computed from Euclidean coordinates in the two-dimensional Poincaré disk after the shared orthogonal alignment defined in Appendix C.2.2. All three errors decrease during optimization and stabilize, with final values $\Delta F = 0.0399$, $\Delta Z = 0.0372$, and $\Delta\alpha = 0.184$.

Specifically,

$$\Delta\alpha = \left(\frac{1}{n}\sum_{i=1}^{n}(\hat{\alpha}_i - \alpha_i^\star)^2\right)^{1/2}.$$

Let

$$\widehat{\boldsymbol{X}} = \begin{bmatrix}\widehat{\boldsymbol{F}}\\ \widehat{\boldsymbol{Z}}\end{bmatrix}, \qquad \boldsymbol{X}^\star = \begin{bmatrix}\boldsymbol{F}^\star\\ \boldsymbol{Z}^\star\end{bmatrix},$$

where $\widehat{\boldsymbol{F}}, \widehat{\boldsymbol{Z}}, \boldsymbol{F}^\star, \boldsymbol{Z}^\star$ denote the corresponding two-dimensional Poincaré-disk coordinates. We choose the shared orthogonal transformation

$$\boldsymbol{Q}^\star = \arg\min_{\boldsymbol{Q}^\top\boldsymbol{Q}=\boldsymbol{I}} \|\widehat{\boldsymbol{X}}\boldsymbol{Q} - \boldsymbol{X}^\star\|_F^2.$$

If $\widehat{\boldsymbol{X}}^\top\boldsymbol{X}^\star = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, then $\boldsymbol{Q}^\star = \boldsymbol{U}\boldsymbol{V}^\top$. We then define

$$\Delta F = \left(\frac{1}{2m}\|\widehat{\boldsymbol{F}}\boldsymbol{Q}^\star - \boldsymbol{F}^\star\|_F^2\right)^{1/2}, \qquad \Delta Z = \left(\frac{1}{2n}\|\widehat{\boldsymbol{Z}}\boldsymbol{Q}^\star - \boldsymbol{Z}^\star\|_F^2\right)^{1/2}.$$

C.2.3 Finite-sample parameter recovery

We first examine parameter recovery under the main simulation setting with $n = 500$ nodes and $m = 1000$ hyperlinks. Figure 5 reports the optimization behavior under the main simulation setting with $n = 500$ nodes and $m = 1000$ hyperlinks for training seed 0. The log-likelihood increases rapidly during the early optimization stage and then approaches a stable plateau. Consistently, the three recovery errors defined in Appendix C.2.2 decrease substantially during training and stabilize at small values. At the end of optimization, we obtain $\Delta F = 0.0399$, $\Delta Z = 0.0372$, and $\Delta\alpha = 0.184$. These results indicate that the proposed estimation procedure achieves stable likelihood optimization while accurately recovering the latent hyperlink and node positions in the aligned two-dimensional Poincaré-disk coordinates and substantially reducing the estimation error of the node-specific effects.

Figure 6 further visualizes the recovered latent structure under the same setting. After the shared orthogonal alignment used only for recovery comparison, the estimated hyperlink and node embeddings closely reproduce the overall geometry of their ground-truth counterparts. The corresponding estimated incidence-probability matrix also recovers the heterogeneous structure of the simulated hypergraph. As described in Appendix C.2.1, this alignment is used solely for comparison with the known ground truth and is not applied during model fitting or hyperlink generation.

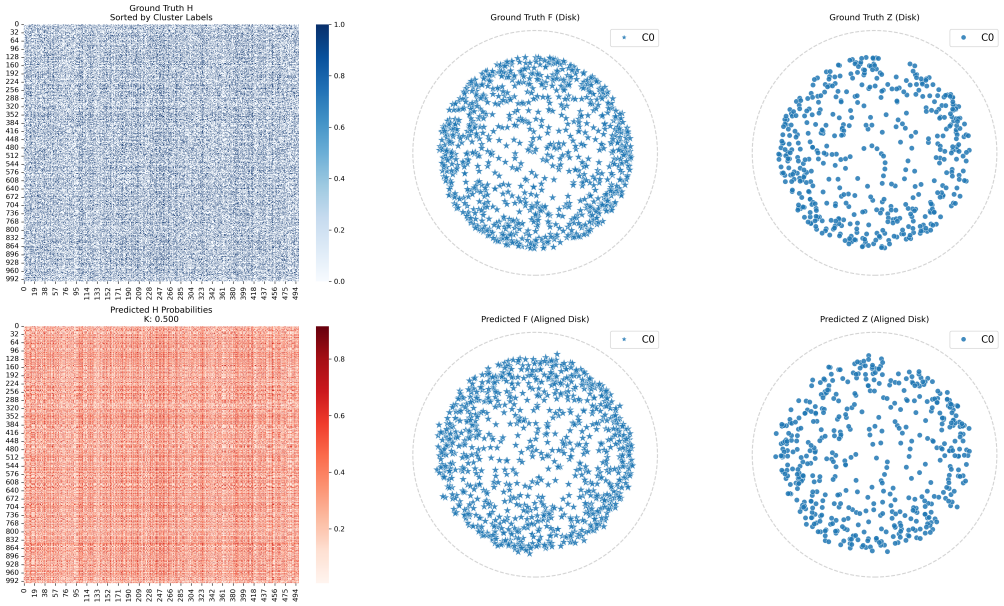

Figure 6: Ground-truth and recovered quantities in the finite-sample parameter-recovery experiment. Top: the simulated incidence matrix and the ground-truth hyperlink and node embeddings in the two-dimensional Poincaré disk. Bottom: the estimated incidence probabilities and the corresponding recovered hyperlink and node embeddings. The estimated embeddings are shown after the shared orthogonal alignment used solely for comparison with the ground truth; this transformation does not affect model fitting or hyperlink generation.

C.2.4 Parameter recovery under increasing sample sizes

We further examine parameter recovery under increasing sample sizes. Proposition E.2 establishes consistency of the conditional MLE in an averaged, aligned parameter error as $m, n \to \infty$. In the dense regime considered here and ignoring optimization error, the bound reduces to

$$r_{m,n}^2 = O_p\left(\frac{(m+n)\log(mn)}{mn}\right).$$

To examine the individual parameter blocks, we additionally use root-effective-sample-size scaling as a reference. Each hyperlink embedding $\boldsymbol{f}_j$ is informed by the n incidence entries in one row of the incidence matrix, whereas each node embedding $\boldsymbol{z}_i$ and node effect α_i is informed by the m entries in one column. This gives the reference scalings

$$\Delta_F \sim n^{-1/2}, \qquad \Delta_Z \sim m^{-1/2}, \qquad \Delta_\alpha \sim m^{-1/2}.$$

The reference lines in the following figures are drawn after placing both axes on a log–log scale. Hence, an error scaling as $x^{-1/2}$ appears as a straight line with slope $-1/2$, whereas an error of constant order with respect to x has slope 0.

We first consider the regime in which n is fixed and $m \to \infty$. In this regime, increasing m provides additional observations for each node-specific parameter but not for each individual hyperlink embedding. Therefore, with respect to m,

$$\Delta_F = O_p(1), \qquad \Delta_Z = O_p(m^{-1/2}), \qquad \Delta_\alpha = O_p(m^{-1/2}).$$

We instantiate this regime by fixing $n = 500$ and $K = 0.5$, varying $m \in \{500, 1000, 2000\}$, and repeating each setting over 10 random seeds. As shown in Figure 7, Δ_Z and Δ_α decrease approximately along the reference slope $-1/2$, while Δ_F remains nearly constant as m increases.

We next consider the joint-growth regime $m = n = N \to \infty$. In this case, the effective number of observations increases for all three parameter blocks, yielding a root-sample-size reference slope of

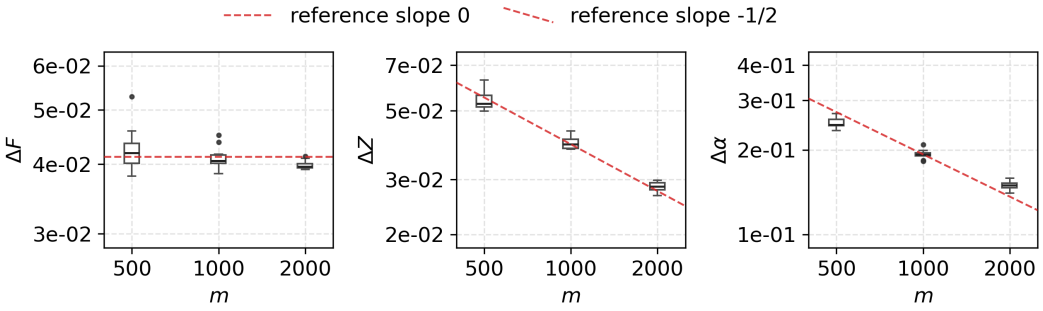

Figure 7: Parameter recovery as the number of hyperlinks increases. We fix $n = 500$ and $K = 0.5$, vary $m \in \{500, 1000, 2000\}$, and report 10 random seeds per setting. Boxes show the interquartile range with the median marked inside; whiskers extend to 1.5 times the interquartile range, with points beyond shown as outliers. Both axes are shown on a log–log scale. The dashed red lines indicate the reference slopes, 0 for Δ_F and $-1/2$ for Δ_Z and Δ_α.

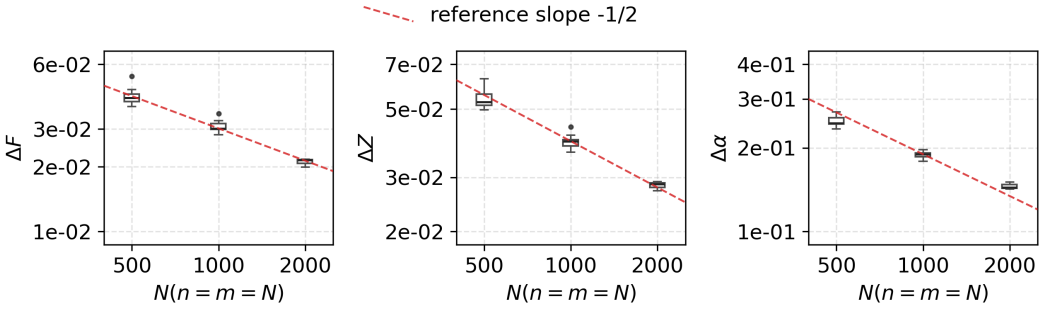

Figure 8: Parameter recovery as both dimensions increase. We set $n = m = N$, vary $N \in \{500, 1000, 2000\}$, fix $K = 0.5$, and report 10 random seeds per setting. Boxes show the interquartile range with the median marked inside; whiskers extend to 1.5 times the interquartile range, with points beyond shown as outliers. Both axes are shown on a log–log scale. The dashed red lines indicate the reference slope $-1/2$ for all three parameter recovery errors.

$-1/2$ for Δ_F, Δ_Z, and Δ_α. Moreover, substituting $m = n = N$ into Proposition E.2 gives

$$r_{N,N} = O_p\left(\sqrt{\frac{\log N}{N}}\right),$$

up to optimization error, which vanishes as $N \to \infty$. We instantiate this regime with $N \in \{500, 1000, 2000\}$, again fixing $K = 0.5$ and using 10 random seeds per setting. As shown in Figure 8, all three recovery errors decrease approximately along the reference slope $-1/2$ as N increases, consistent with the overall consistency result in Proposition E.2.

C.2.5 Curvature recovery and selection

We further examine whether the ground-truth curvature $K^\star = 0.5$ can be recovered under the two curvature-selection procedures described in the main text. We first select K by the attained in-sample training log-likelihood over the candidate grid. When using held-out data, we instead select K by the smallest mean validation Energy distance. In the reported simulation setting, both procedures recover the ground-truth curvature.

Unlike the parameter-recovery experiments above, the following experiments adopt the same data-split and random-seed protocol as the real-world experiments in Appendix D.1, use the evaluation metrics defined in Appendix B.6, and follow the baseline configurations described in Appendix B.5. The simulated hyperlinks are split with proportions $0.8/0.1/0.1$, and results are reported as mean

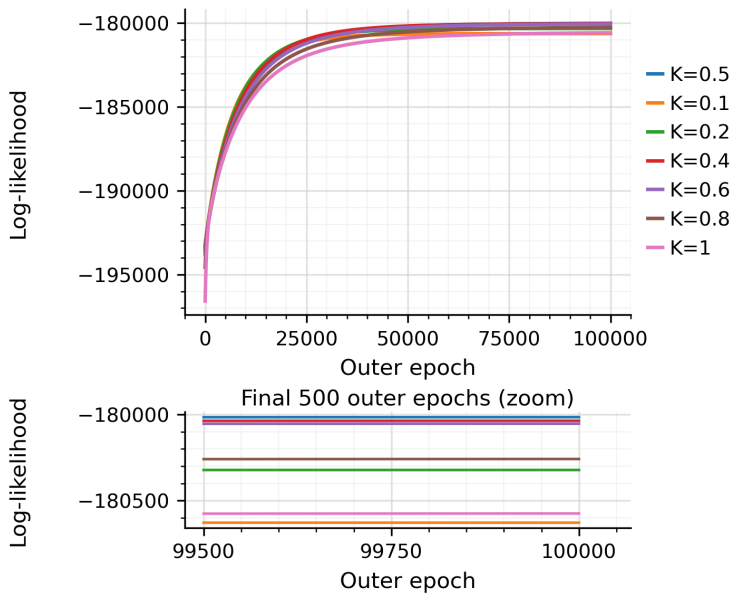

Figure 9: Likelihood-based recovery of the curvature parameter. We fit the latent hyperbolic model over the candidate curvature values and compare their optimized likelihoods. The maximum is attained at the ground-truth curvature $K^\star = 0.5$, indicating correct curvature recovery in the finite-sample simulation.

	H2G-EB (Fidelity ↓)				H2G-RD (Fidelity ↓)			
K	Energy ($\times 10^{-3}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-2}$)	Δ_{H_v} ($\times 10^{-2}$)	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-2}$)	Δ_{H_v} ($\times 10^{-2}$)
0.1	1.0503 (0.25)	7.32 (0.14)	4.72 (0.04)	2.00 (0.00)	6.3375 (2.1)	7.09 (0.10)	4.52 (0.02)	1.99 (0.00)
0.2	1.0532 (0.25)	7.32 (0.15)	4.72 (0.04)	1.99 (0.00)	5.8839 (1.9)	7.08 (0.08)	4.51 (0.02)	1.98 (0.00)
0.4	1.0424 (0.24)	7.31 (0.15)	4.72 (0.04)	1.99 (0.00)	5.8457 (1.9)	7.07 (0.09)	4.50 (0.02)	1.99 (0.00)
0.5†	**1.0416** (0.24)	7.30 (0.15)	4.72 (0.04)	1.99 (0.00)	**5.6274** (1.7)	7.08 (0.06)	4.49 (0.02)	1.99 (0.00)
0.6	1.0612 (0.23)	7.32 (0.15)	4.72 (0.04)	1.99 (0.00)	5.7745 (1.6)	7.10 (0.06)	4.49 (0.02)	2.00 (0.00)
0.8	1.1091 (0.22)	7.33 (0.14)	4.72 (0.03)	2.01 (0.00)	6.2915 (1.8)	7.14 (0.05)	4.48 (0.03)	2.01 (0.00)
1.0	1.1772 (0.20)	7.37 (0.14)	4.73 (0.03)	2.02 (0.00)	7.2576 (1.5)	7.20 (0.08)	4.53 (0.03)	2.04 (0.00)

Table 4: Validation performance of H2G-EB and H2G-RD across candidate curvature values on the simulated dataset. Energy means are reported to five significant digits to distinguish closely performing curvature candidates. The symbol † marks the curvature selected by the smallest mean validation Energy distance, which is bolded. Both implementations select the ground-truth value $K^\star = 0.5$ based on the unrounded results.

(standard deviation) across sampling seeds 0–4. We also follow the same rounding and highlighting conventions as in the main text.

Likelihood-based curvature recovery. Figure 9 reports the optimized likelihood across candidate curvature values. The likelihood is maximized at the ground-truth value $K^\star = 0.5$, showing that the likelihood-based estimator correctly recovers the curvature in this finite-sample setting.

Validation-based curvature selection. For the generative experiments, we select the curvature using the smallest mean Energy distance on the validation split of the dataset, following the same model-selection protocol as in the real-world experiments. Table 4 reports the validation performance of H2G-EB and H2G-RD across the candidate curvature values. For both methods, the minimum mean validation Energy distance is attained at $K = 0.5$, exactly matching the ground-truth curvature $K^\star = 0.5$. This shows that the validation-based selection criterion successfully recovers the true curvature for both H2G implementations.

Method	Fidelity ↓			
	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-2}$)	Δ_{H_v} ($\times 10^{-2}$)
H2G-EB	<u>3.52</u> (0.12)	<u>7.06</u> (0.10)	<u>4.45</u> (0.03)	<u>2.02</u> (0.00)
H2G-RD	**2.89** (1.4)	**7.02** (0.12)	4.57 (0.02)	**1.99** (0.00)
DDE	8.72 (1.9)	7.27 (0.15)	4.64 (0.05)	2.11 (0.00)
Gau-Diff^c	27.5 (0.92)	8.58 (0.06)	4.45 (0.00)	2.80 (0.00)
Tab-DDPM^c	30.5 (1.0)	8.73 (0.05)	4.45 (0.00)	2.87 (0.00)
WGAN^c	15.7 (1.5)	7.99 (0.13)	4.45 (0.00)	2.53 (0.00)
Ber-Diff	1340 (12)	78.7 (1.6)	24.5 (0.04)	6.59 (0.01)
VAE	798 (24)	44.4 (2.2)	15.9 (0.05)	2.95 (0.01)
HyperLap	217 (13)	15.8 (0.59)	9.95 (0.27)	2.82 (0.00)
HyperPLR	9.00 (0.86)	7.57 (0.07)	**4.44** (0.00)	2.09 (0.00)

Table 5: Generation performance on the simulated dataset. Values are mean (standard deviation) across sampling seeds 0–4. Best and second-best results are shown in bold and underlined, respectively. Superscript c denotes calibrated baselines.

Together with the likelihood-based result above, this provides two complementary empirical checks that the ground-truth curvature can be correctly identified in the simulated setting.

C.3 Generation performance on the simulated dataset

Based on the results above, both H2G-EB and H2G-RD select the ground-truth curvature, and we next compare their generative performance with the baselines following the same protocol as in the real-world experiments.

Table 5 shows that both H2G implementations achieve strong generative fidelity on the simulated dataset. H2G-RD obtains the best Energy distance, MMD^2, and Δ_{H_v}, while H2G-EB ranks second on all four fidelity metrics. Both implementations consistently outperform the uncalibrated baselines across the distributional metrics, with particularly large improvements over VAE and HyperLap. Although HyperPLR attains the best uncalibrated Δ_{H_d} by a small margin, its Energy distance, MMD^2, and Δ_{H_v} are all noticeably worse than those of H2G. The calibrated baselines achieve comparable Δ_{H_d} values by construction, whereas H2G-EB reaches the same level without post-hoc calibration.

D Real-data experiments

This section provides supplementary details and analyses for the real-world experiments in Section 4. We first describe the common experimental protocol and dataset construction and pre-processing, followed by a sensitivity analysis of the DDE embedding dimension and additional analyses of memorization, novelty, and privacy-related behavior. We then report dataset-specific optimization diagnostics, latent-space analyses, and qualitative interpretations for MIMIC-III, MathOverflow-Answers, and CORUM.

D.1 Experimental settings

For all real-world datasets, we use the fixed $8{:}1{:}1$ train/validation/test split described in Section 4.1, shared across all methods. Hyperparameters are selected on the validation set, and the selected configuration is subsequently evaluated on the held-out test set. For methods involving stochastic training, model fitting uses training seed 0. For all methods, we generate five synthetic hypergraphs using sampling seeds $0, \ldots, 4$ and report the mean and standard deviation of the resulting evaluation metrics in the main text.

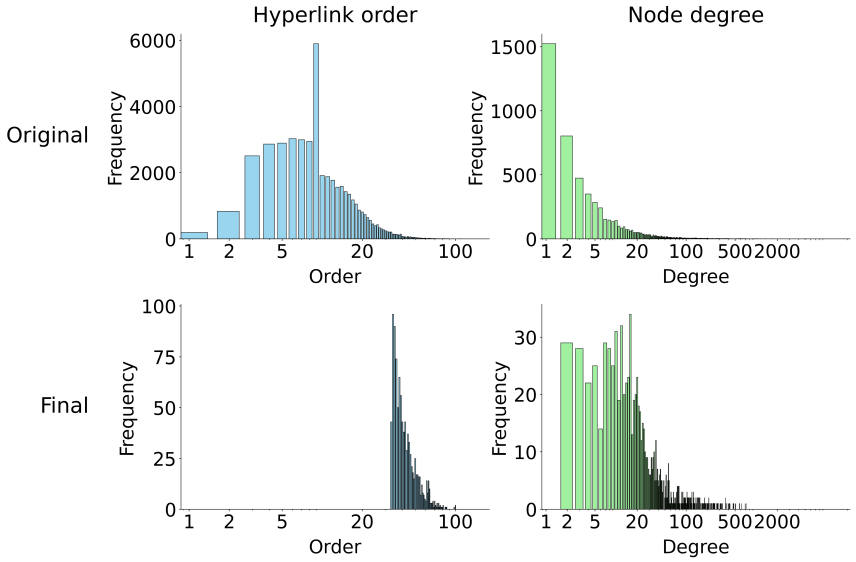

Figure 10: Hyperlink-order and node-degree distributions before and after pre-processing for MIMIC-III. "Original" and "Final" denote the distributions before pre-processing and those used in our experiments, respectively.

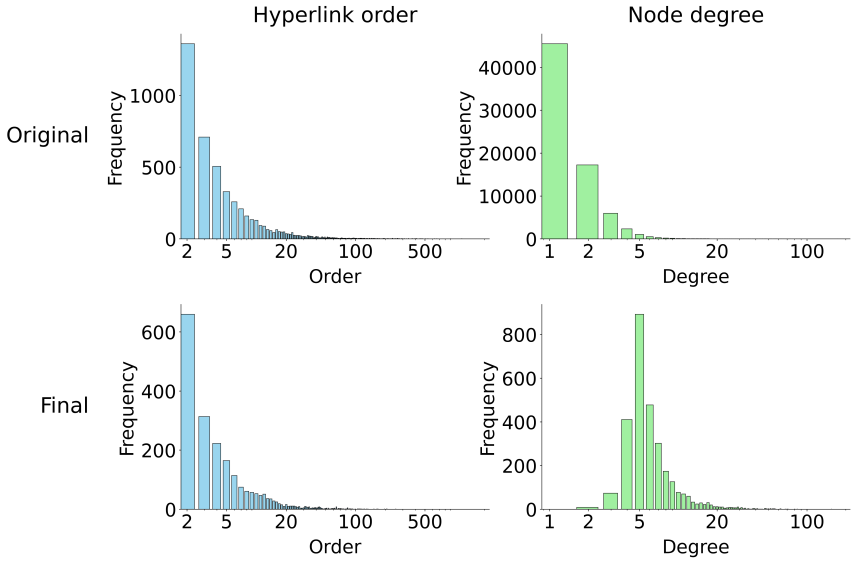

Figure 11: Hyperlink-order and node-degree distributions before and after pre-processing for MathOverflow-Answers. "Original" and "Final" denote the distributions before pre-processing and those used in our experiments, respectively.

D.2 Datasets and pre-processing

Our real-world datasets span diverse domains, including healthcare, online communities, and biology, and include both established hypergraph benchmarks and hypergraphs constructed from domain-specific data. Together, these datasets provide diverse real-world settings for hypergraph generation and span a broad range of graph sizes, allowing us to evaluate both the generation fidelity and scalability of H2G. All datasets are obtainable from their respective sources; MIMIC-III requires credentialed access through PhysioNet, while the others are publicly available. The hyperlink-order and node-degree distributions before and after pre-processing are shown in Figures 10, 11, and 12.

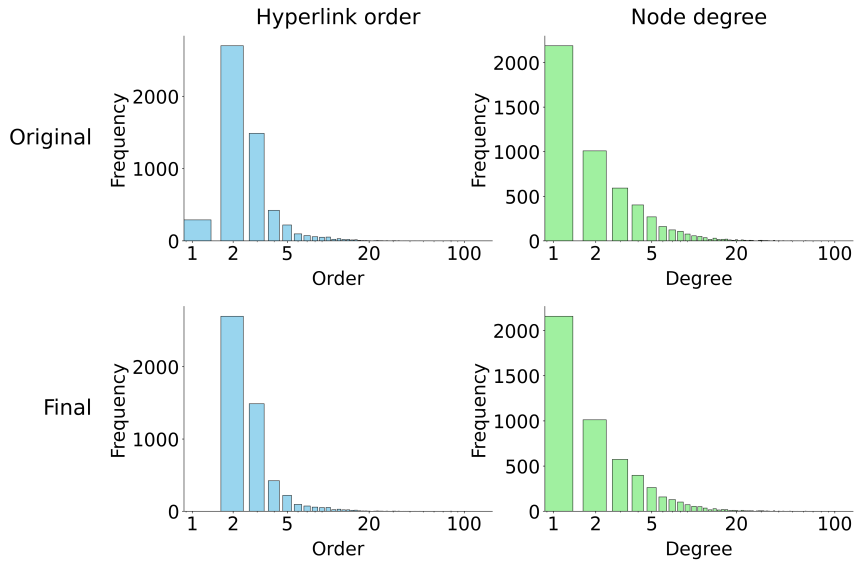

Figure 12: Hyperlink-order and node-degree distributions before and after pre-processing for CORUM. "Original" and "Final" denote the distributions before pre-processing and those used in our experiments, respectively.

MIMIC-III. Following the construction used in Wu et al. (2025), we construct a symptom co-occurrence hypergraph from MIMIC-III v1.4 (Johnson et al., 2016). Symptoms are represented by ICD codes[3], which inherit the hierarchy of the ICD-9-CM classification system, while each patient record induces a hyperlink among the recorded symptoms. The dataset therefore combines heterogeneous co-occurrence patterns with an underlying hierarchical node structure, making it particularly relevant for evaluating hyperbolic representations of clinical dependencies. Generating realistic symptom records can support privacy-conscious data sharing and methodological research without directly exposing individual patient records. We construct the patient–symptom incidence matrix from the MIMIC-III diagnosis table `DIAGNOSES_ICD.csv`, aggregating diagnosis records by `SUBJECT_ID`. Each patient therefore corresponds to one hyperlink, and an ICD-9-CM code is included if it appears at least once in that patient's diagnosis records. We remove diagnosis codes observed in fewer than 20 patients and exclude `V`- and `E`-prefixed codes. We then retain the 1,000 highest-degree symptom nodes and select the 1,000 largest-order patient hyperlinks. Finally, we iteratively remove hyperlinks of order below 2 and nodes appearing in fewer than two retained hyperlinks until the hypergraph stabilizes. The resulting hypergraph contains 891 nodes and 1,000 hyperlinks.

MathOverflow-Answers. MathOverflow-Answers (Benson et al., 2018) is constructed from user activity on MathOverflow, with questions as nodes and each user's set of answered questions forming a hyperlink. It contains heterogeneous hyperlink sizes and substantial overlap among hyperlinks, reflecting diverse patterns of user participation and shared interests across questions. Generating realistic user–question participation patterns can facilitate the study of collective expertise and interaction structure in online knowledge communities. For pre-processing, we retain the 3,000 highest-degree question nodes and restrict all hyperlinks to this node set. We then iteratively prune hyperlinks of order below 2 and isolated nodes until the hypergraph stabilizes. This results in 3,000 nodes and 2,272 hyperlinks.

CORUM. We construct a protein-complex hypergraph from the human protein complexes in CORUM Release 5.3 (Tsitsiridis et al., 2023; Steinkamp et al., 2025). For each complex, we extract the listed UniProt accessions from the CORUM complex table, treat each unique accession as a protein node, and use the set of proteins in that complex as one hyperlink. Duplicate UniProt

[3] `https://www.cms.gov/medicare/coding-billing/icd-10-codes/icd-9-cm-diagnosis-procedure-codes-abbreviated-and-full-code-titles`

Method	MIMIC-III (Fidelity ↓)				MathOverflow-Answers (Fidelity ↓)			
	Energy ($\times 10^{-3}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-2}$)	Δ_{H_v} ($\times 10^{-3}$)	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-3}$)	Δ_{H_v} ($\times 10^{-4}$)
H2G-EB	<u>1.30</u> (0.07)	<u>7.44</u> (0.08)	<u>2.16</u> (0.01)	4.17 (0.00)	**3.71** (0.47)	**3.50** (0.03)	<u>4.08</u> (0.01)	**4.08** (0.00)
H2G-RD	**1.23** (0.02)	**7.41** (0.08)	**2.13** (0.00)	4.20 (0.00)	<u>5.74</u> (0.14)	<u>3.56</u> (0.04)	4.96 (0.03)	4.26 (0.01)
DDE ($d = 2$)	1.42 (0.09)	<u>7.44</u> (0.06)	2.19 (0.01)	4.19 (0.00)	152 (8.2)	12.5 (2.0)	**4.06** (0.01)	<u>4.10</u> (0.00)
DDE ($d = 4$)†	1.35 (0.11)	7.47 (0.11)	2.21 (0.01)	<u>4.16</u> (0.00)	56.5 (10)	7.10 (1.0)	4.11 (0.01)	4.14 (0.00)
DDE ($d = 8$)‡	1.56 (0.09)	7.50 (0.08)	2.24 (0.01)	**4.15** (0.00)	30.2 (5.5)	5.34 (0.76)	4.20 (0.02)	4.13 (0.01)

Table 6: Generation performance under different DDE embedding dimensions on MIMIC-III and MathOverflow-Answers. All values are evaluated on the held-out test split. † marks the DDE dimension selected using validation Energy distance for MIMIC-III, while ‡ marks the selected dimension for MathOverflow-Answers. Bold and underlined values denote the best and second-best test performance, respectively, among the displayed methods.

Method	CORUM (Fidelity ↓)			
	Energy ($\times 10^{-4}$)	MMD^2 ($\times 10^{-3}$)	Δ_{H_d} ($\times 10^{-3}$)	Δ_{H_v} ($\times 10^{-5}$)
H2G-EB	<u>8.31</u> (0.83)	<u>2.29</u> (0.12)	**1.22** (0.00)	<u>4.55</u> (0.00)
H2G-RD	**6.23** (0.49)	**2.12** (0.06)	1.35 (0.01)	4.70 (0.01)
DDE ($d = 2$)†	2530 (84)	163 (7.2)	<u>1.27</u> (0.00)	4.74 (0.02)
DDE ($d = 4$)	2580 (62)	168 (3.6)	<u>1.27</u> (0.002)	4.56 (0.015)
DDE ($d = 8$)	2690 (55)	166 (9.9)	<u>1.27</u> (0.00)	**4.49** (0.01)

Table 7: Generation performance of DDE under different Euclidean embedding dimensions on CORUM. All values are evaluated on the held-out test split. The symbol † marks the embedding dimension selected using validation Energy distance. Bold and underlined values denote the best and second-best test performance, respectively, among the displayed methods.

accessions within a complex are removed. CORUM is highly sparse: each protein participates in only a small fraction of all complexes, and each complex contains only a small subset of the protein universe. Modeling such selective co-membership patterns is therefore challenging. Generating plausible protein complexes can provide candidate protein associations for hypothesis generation and may facilitate the identification of previously uncharacterized complexes. We retain all complexes without subsampling and iteratively prune complexes of order below 2 and isolated proteins until the hypergraph stabilizes. The resulting hypergraph contains 5,227 nodes and 5,338 hyperlinks.

D.3 DDE EMBEDDING-DIMENSION SENSITIVITY

The latent embedding dimension is an important modeling choice in DDE (Wu et al., 2025). Following the dimensions considered in the original DDE real-data experiments, we treat $d \in \{2, 4, 8\}$ as candidate embedding dimensions and select d using the mean Energy distance on the validation split. The selected dimensions are $d = 4$ for MIMIC-III, $d = 8$ for MathOverflow-Answers, and $d = 2$ for CORUM. After selection, the dimension is fixed and performance is evaluated on the held-out test split. Tables 6 and 7 additionally report the test performance of all candidate dimensions to assess the sensitivity of DDE to this hyperparameter. As in the main text, we report mean (standard deviation) over five sampling seeds with fixed training seed, with means to three significant digits and standard deviations using the shorter of two significant digits and two decimal places.

DDE exhibits noticeable and non-monotonic sensitivity to the embedding dimension: increasing d does not consistently improve generative fidelity, and the best-performing dimension varies across datasets and metrics. This motivates selecting d using held-out validation data rather than post-hoc test performance.

More importantly, H2G achieves strong fidelity using a fixed two-dimensional hyperbolic latent space. On MIMIC-III, both H2G implementations achieve lower Energy and MMD^2 than the validation-

selected DDE configuration. On MathOverflow-Answers, both implementations substantially reduce Energy and MMD^2 relative to all tested DDE dimensions, while H2G-EB also improves Δ_{H_d} and Δ_{H_v} over the selected DDE model. The difference is most pronounced on CORUM, where both implementations reduce Energy and MMD^2 by orders of magnitude relative to all tested DDE dimensions. These results indicate that the gains of H2G are not attributable to comparison against a single fixed Euclidean embedding dimension and support a two-dimensional hyperbolic space as an expressive and parsimonious latent representation.

D.4 Memorization, novelty, and privacy analysis

We complement the fidelity evaluation with an analysis of memorization, novelty, and privacy-related behavior for the two latent-distribution modeling strategies in H2G. Our goal is to characterize how empirical resampling in H2G-EB and Riemannian diffusion in H2G-RD affect the relationship between generated hyperlinks and the training data under otherwise identical experimental settings. In particular, while both variants use the same fitted hyperbolic generative model and stochastic Bernoulli decoder, they differ in how new latent hyperlink embeddings are generated. We therefore examine whether this difference is reflected in direct training-set reproduction, generation novelty, and empirical membership-related diagnostics.

Memorization, novelty, and privacy diagnostics. We examine whether the generated hyperlinks reproduce the training data and whether the two latent-generation strategies differ in observed-space novelty. Let $\mathcal{H}_{\text{train}}$, $\mathcal{H}_{\text{val}}$, $\mathcal{H}_{\text{test}}$, and $\widetilde{\mathcal{H}}$ denote the training, validation, test, and generated hyperlink sets, respectively. We report the exact training-match rate, defined as the fraction of generated hyperlinks that exactly coincide with at least one hyperlink in $\mathcal{H}_{\text{train}}$, and the fraction of unique generated hyperlinks. To quantify novelty beyond exact matching, for each generated hyperlink $\tilde{h} \in \widetilde{\mathcal{H}}$ we define its nearest-training Jaccard distance as

$$d_{\text{train}}(\tilde{h}) = \min_{h \in \mathcal{H}_{\text{train}}} d_J(\tilde{h}, h),$$

and report the average of $d_{\text{train}}(\tilde{h})$ over $\widetilde{\mathcal{H}}$.

As privacy-related diagnostics, we consider membership-inference attack (MIA) AUROC (Shokri et al., 2017) and nearest-neighbor adversarial accuracy (NNAA) (Yale et al., 2020). For MIA, hyperlinks in $\mathcal{H}_{\text{train}}$ are treated as members and those in $\mathcal{H}_{\text{test}}$ as non-members. For each real hyperlink h, we use its proximity to the generated set,

$$s(h) = -\min_{\tilde{h} \in \widetilde{\mathcal{H}}} d_J(h, \tilde{h}),$$

as the membership score and compute the corresponding AUROC A_{MIA} for distinguishing training from test hyperlinks. Because an AUROC below 0.5 corresponds to the same distinguishability after reversing the attack score, we report the symmetrized value

$$\text{MIA} = \frac{1}{2} + \left| A_{\text{MIA}} - \frac{1}{2} \right|,$$

so that values closer to 0.5 indicate weaker membership information.

For NNAA, let $\text{AA}(\mathcal{A}, \widetilde{\mathcal{H}})$ denote the nearest-neighbor adversarial accuracy between a real hyperlink set $\mathcal{A}$ and the generated set. For $a \in \mathcal{A}$, let

$$d(a, \widetilde{\mathcal{H}}) = \min_{\tilde{h} \in \widetilde{\mathcal{H}}} d_J(a, \tilde{h}), \qquad d(a, \mathcal{A} \setminus \{a\}) = \min_{a' \in \mathcal{A} \setminus \{a\}} d_J(a, a'),$$

with analogous definitions for generated hyperlinks. We compute

$$\text{AA}(\mathcal{A}, \widetilde{\mathcal{H}}) = \frac{1}{2} \left[\frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \mathbf{1}\Big\{ d(a, \widetilde{\mathcal{H}}) > d(a, \mathcal{A} \setminus \{a\}) \Big\} + \frac{1}{|\widetilde{\mathcal{H}}|} \sum_{\tilde{h} \in \widetilde{\mathcal{H}}} \mathbf{1}\Big\{ d(\tilde{h}, \mathcal{A}) > d(\tilde{h}, \widetilde{\mathcal{H}} \setminus \{\tilde{h}\}) \Big\} \right].$$

Writing A_{train}, A_{val}, and A_{test} for $\text{AA}(\mathcal{H}_{\text{train}}, \widetilde{\mathcal{H}})$, $\text{AA}(\mathcal{H}_{\text{val}}, \widetilde{\mathcal{H}})$, and $\text{AA}(\mathcal{H}_{\text{test}}, \widetilde{\mathcal{H}})$, respectively, we report

$$\text{NNAA Gap} = \frac{1}{2} \left(|A_{\text{train}} - A_{\text{val}}| + |A_{\text{train}} - A_{\text{test}}| \right),$$

Table 8: Memorization, novelty, and privacy-related diagnostics for H2G-EB and H2G-RD on the test set. Results are mean (standard deviation) over five sampling seeds. Bold values indicate the better result within each dataset according to the direction shown in the column header; ties are both bolded. For reference, the mean nearest-training Jaccard distances of the held-out real test hyperlinks are 0.744, 0.796, and 0.557 for MIMIC-III, MathOverflow-Answers, and CORUM, respectively.

Dataset	Method	Exact Train ($\times 10^{-3}$) $\downarrow$	Unique Gen. $\uparrow$	Gen–Train Dist. $\uparrow$	MIA ($\rightarrow 0.5$)	NNAA Gap $\downarrow$
MIMIC-III	H2G-EB	**0.00 (0.00)**	**1.00 (0.00)**	0.763 (0.00)	**0.528 (0.03)**	0.0415 (0.01)
	H2G-RD	**0.00 (0.00)**	**1.00 (0.00)**	**0.767 (0.00)**	0.529 (0.01)	**0.0358 (0.01)**
MathOverflow-Answers	H2G-EB	**0.0998 (0.03)**	0.989 (0.00)	0.838 (0.00)	**0.534 (0.00)**	**0.0474 (0.01)**
	H2G-RD	0.114 (0.02)	**0.992 (0.00)**	**0.845 (0.00)**	0.540 (0.01)	0.0505 (0.01)
CORUM	H2G-EB	1.08 (0.11)	0.920 (0.00)	0.762 (0.00)	0.571 (0.01)	0.0466 (0.00)
	H2G-RD	**0.670 (0.07)**	**0.956 (0.00)**	**0.787 (0.00)**	**0.558 (0.01)**	**0.0412 (0.00)**

where smaller values indicate less training-set-specific behavior. All distance-based diagnostics use the Jaccard distance defined in Appendix B.6. We follow the same data pre-processing, train/validation/test splits, H2G-EB/H2G-RD implementations, and five sampling seeds as in the main experiments, and compute these additional diagnostics on the corresponding generated samples.

Results. Table 8 shows that both H2G-EB and H2G-RD exhibit limited direct memorization of the training hypergraphs. Neither method produces an exact training-set match on MIMIC-III, and the corresponding rates remain extremely small on MathOverflow-Answers and CORUM. The fraction of unique generated hyperlinks is also high across all datasets, ranging from 0.920 to 1.00. In particular, these results clarify that empirical latent resampling in H2G-EB does not amount to directly reproducing observed training hyperlinks: although fitted hyperlink embeddings are resampled from their empirical support, the subsequent stochastic Bernoulli decoding can still produce new binary hyperlinks.

The clearest and most consistent difference between the two variants concerns generation novelty. H2G-RD achieves a larger mean nearest-training distance on all three datasets, increasing from 0.763 to 0.767 on MIMIC-III, from 0.838 to 0.845 on MathOverflow-Answers, and from 0.762 to 0.787 on CORUM. It also generates at least as large a fraction of unique hyperlinks on every dataset, with the most pronounced improvement on CORUM. While H2G-EB resamples from the empirical support of the fitted hyperlink embeddings, H2G-RD introduces an additional Riemannian diffusion stage to model their latent distribution. This provides a mechanism for exploring latent locations beyond those observed in the fitted sample. The consistently larger nearest-training distances observed for H2G-RD suggest that this additional flexibility may contribute to greater novelty after stochastic decoding, providing further empirical motivation for including the diffusion stage.

Importantly, the larger nearest-training distances of H2G-RD do not simply coincide with uniformly lower generation fidelity. As shown in Tables 1 and 2, H2G-RD achieves lower Energy and MMD^2 than H2G-EB on both MIMIC-III and CORUM, while simultaneously generating hyperlinks farther from the training set. On CORUM, H2G-EB remains better on Δ_{H_d} and Δ_{H_v}, indicating that the relationship between novelty and fidelity can depend on the aspect of the generated distribution being evaluated. On MathOverflow-Answers, H2G-EB achieves better fidelity across the reported metrics, whereas H2G-RD produces a larger fraction of unique hyperlinks and a larger nearest-training distance, suggesting a clearer fidelity–novelty trade-off on this dataset. Overall, these results indicate that the additional flexibility of Riemannian diffusion can increase novelty without necessarily degrading distributional fidelity, although the balance is dataset dependent.

The privacy-related diagnostics provide a more nuanced comparison. MIA AUROC remains between 0.528 and 0.571 across all settings, while NNAA gaps remain below 0.051, providing limited empirical evidence of strong training-set-specific behavior for either variant. H2G-RD yields smaller NNAA gaps on MIMIC-III and CORUM and an MIA AUROC closer to 0.5 on CORUM, whereas

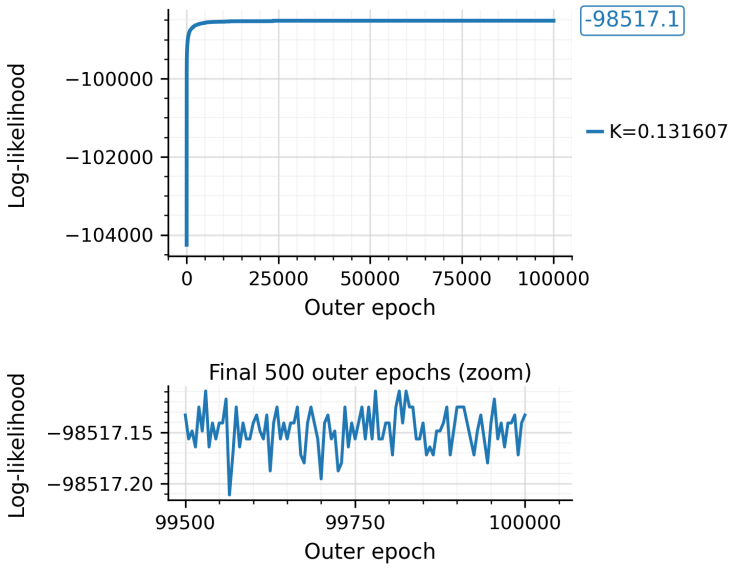

Figure 13: Embedding-stage optimization for H2G-EB on MIMIC-III with the validation-selected curvature $K \approx 0.13$. The curves show the optimization trajectory of the fitted latent model, together with a magnified view of the final training stage to illustrate the behavior near the attained solution.

H2G-EB is slightly better on both diagnostics for MathOverflow-Answers. We therefore do not observe a uniform privacy advantage for either variant across all diagnostics. Rather, the most consistent empirical benefit of H2G-RD is increased generation novelty, while both variants exhibit low direct memorization and relatively weak membership-related signals.

D.5 Additional MIMIC-III results and analysis

We provide additional analyses of the fitted H2G models on MIMIC-III. We first examine optimization diagnostics for the embedding and diffusion stages, then visualize the learned and generated latent hyperlink geometry, and finally analyze the organization of the learned node embeddings through Riemannian clustering.

D.5.1 Training and optimization diagnostics

Figures 13–15 show the optimization behavior of the embedding and diffusion stages on MIMIC-III. For H2G-EB, the log-likelihood increases rapidly during the early optimization stage and subsequently reaches a stable plateau, with only small fluctuations over the final 500 outer epochs (Figure 13). For H2G-RD, the embedding-stage log-likelihood exhibits the same rapid initial improvement followed by substantially slower changes, although a gradual increase remains visible near the end of optimization (Figure 14). The Riemannian diffusion training loss exhibits substantial stochastic variation at the individual-step level, as expected under random mini-batch sampling, diffusion-time sampling, and perturbation of latent embeddings, while its exponential moving average (EMA) decreases substantially during training and stabilizes in the later optimization steps (Figure 15). Overall, these diagnostics indicate stable optimization of both the latent embedding and diffusion stages on MIMIC-III.

D.5.2 Latent geometry of generated hyperlinks

Figures 16 and 17 visualize the latent geometry learned on MIMIC-III. In both fitted models, the hyperlink embeddings $\widehat{\mathbf{F}}$ exhibit an approximately annular structure in the Poincaré disk. This radial concentration can arise because the Bernoulli decoder depends on the relative hyperbolic distances between hyperlink and node embeddings: hyperlinks with comparable overall connectivity may occupy similar hyperbolic radii while differing primarily in angular position according to their

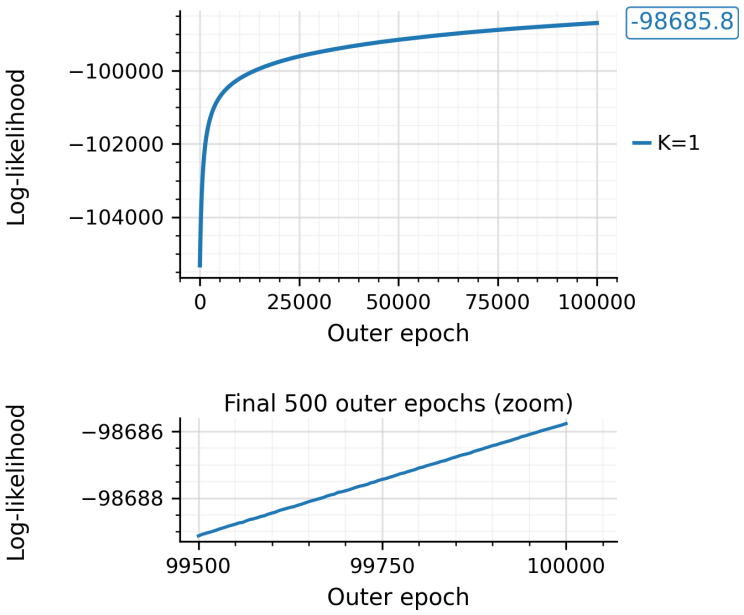

Figure 14: Embedding-stage optimization for H2G-RD on MIMIC-III with the validation-selected curvature $K = 1$. The curves show the optimization trajectory of the fitted latent model, together with a magnified view of the final training stage to illustrate the behavior near the attained solution.

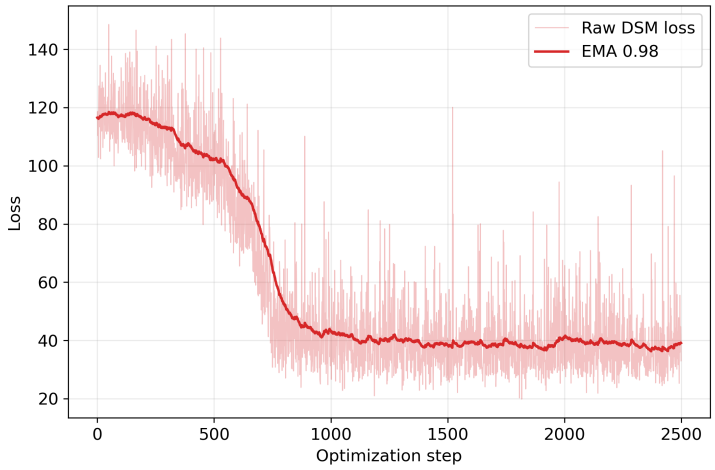

Figure 15: Riemannian diffusion training loss for H2G-RD on MIMIC-III with the validation-selected curvature $K = 1$. We report both the per-step loss and its exponential moving average (EMA), which provides a smoother view of the training trajectory.

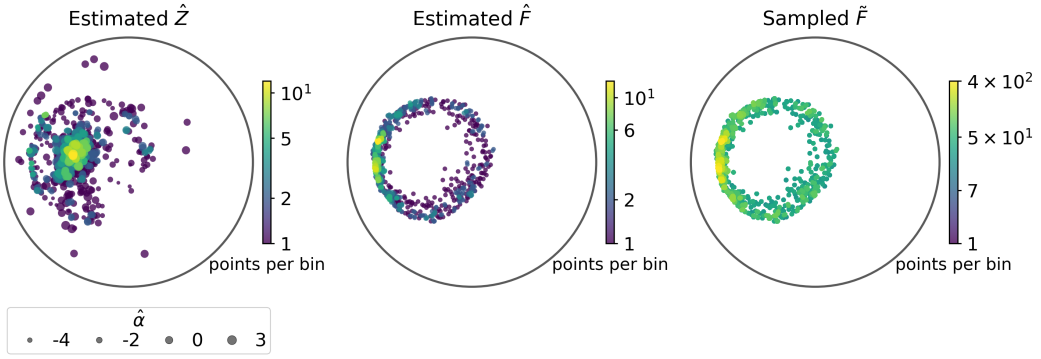

Figure 16: Latent representations learned on MIMIC-III and generated by H2G-EB. Left: estimated node embeddings $\widehat{\mathbf{Z}}$ in the Poincaré disk, where marker size increases with the estimated node effect $\widehat{\alpha}$. Middle: estimated hyperlink embeddings $\widehat{\mathbf{F}}$ obtained from the observed training hypergraph. Right: sampled hyperlink embeddings $\widetilde{\mathbf{F}}$ obtained by empirical resampling from the fitted hyperlink embeddings, shown for sampling seed 0. Colors indicate local point density on a logarithmic scale. As expected from empirical resampling, the generated embeddings preserve the fitted latent support.

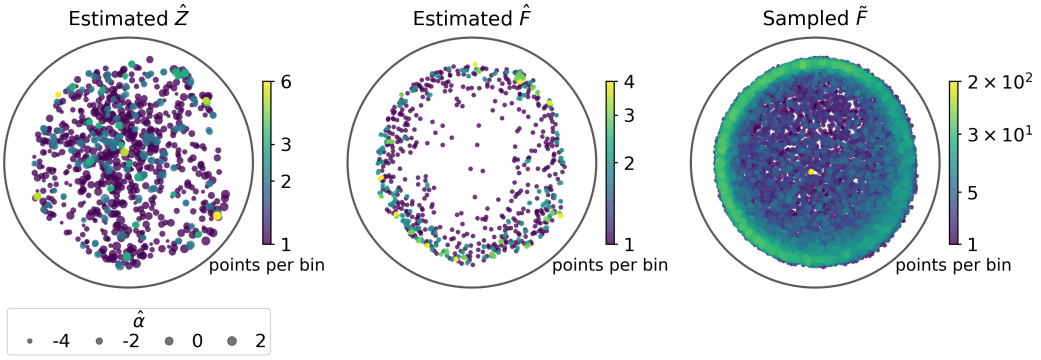

Figure 17: Latent representations learned on MIMIC-III and generated by H2G-RD. Left: estimated node embeddings $\widehat{\mathbf{Z}}$ in the Poincaré disk, where marker size increases with the estimated node effect $\widehat{\alpha}$. Middle: estimated hyperlink embeddings $\widehat{\mathbf{F}}$ obtained from the observed training hypergraph. Right: sampled hyperlink embeddings $\widetilde{\mathbf{F}}$ generated by the fitted Riemannian diffusion model. In contrast to empirical resampling, H2G-RD generates latent hyperlinks at previously unseen locations while retaining the broad geometric organization of the fitted hyperlink embeddings. Colors indicate local point density on a logarithmic scale.

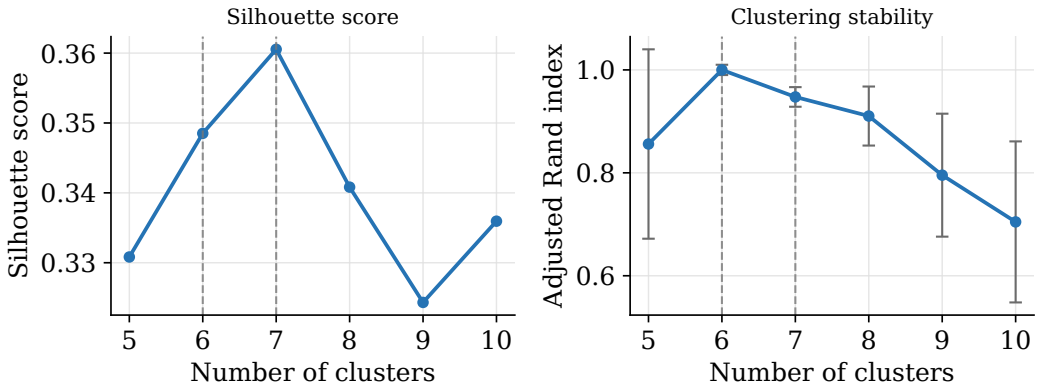

Figure 18: Selection of the number of clusters for the MIMIC-III symptom embeddings. The left panel reports the silhouette score based on hyperbolic geodesic distances, and the right panel reports the mean adjusted Rand index (ARI) across random initializations with ± 1 standard deviation error bars. Higher values indicate better cluster separation and greater clustering stability, respectively. The dashed vertical lines mark the six- and seven-cluster solutions.

node-composition patterns. The nonlinear Poincaré representation can further make concentration at similar hyperbolic radii appear as a pronounced ring in the disk.

The latent configurations learned by H2G-EB and H2G-RD differ substantially in their apparent scale and shape because the two implementations select different curvatures on MIMIC-III. Changing K changes the hyperbolic distance scale and consequently the latent configuration favored by likelihood-based fitting, so embeddings fitted under different curvatures are not directly comparable through their Euclidean coordinates in the Poincaré disk. Despite these geometric differences, both implementations achieve strong generation fidelity. The observed hyperlink distribution is determined jointly by K, the latent embeddings, the node effects, and the Bernoulli decoder, allowing distinct latent configurations to induce similar inclusion probabilities and comparable distributions in the observed hypergraph space.

The two implementations also differ in how they generate latent hyperlinks. H2G-EB resamples directly from the empirical distribution of $\widehat{\mathbf{F}}$ and therefore remains on the fitted latent support by construction. H2G-RD, in contrast, models the learned hyperlink embeddings with Riemannian diffusion and generates new latent hyperlinks at previously unseen locations in the hyperbolic space. Figure 17 shows that these samples retain the broad radial organization of the fitted embeddings while extending beyond their empirical support.

A localized high-density region is also visible in the H2G-RD samples. Similar concentration appears on multiple real-world datasets, suggesting that it is associated with the diffusion dynamics rather than a MIMIC-specific semantic cluster. A possible explanation is that the learned score field, reference distribution, and finite-step reverse diffusion jointly concentrate a subset of sampling trajectories within a small latent region; the nonlinear Poincaré projection may further affect its visual prominence. Nevertheless, this localized concentration coexists with strong fidelity after decoding, showing that close agreement in the observed hyperlink distribution does not require pointwise agreement between the fitted and generated latent densities.

D.5.3 Clustering of node embeddings

Cluster number selection. We apply Riemannian k-means to the learned MIMIC-III symptom embeddings in the Poincaré disk for qualitative interpretation. We consider five to ten clusters, selecting among them based on hyperbolic-distance silhouette scores (Rousseeuw, 1987) and clustering stability across random initializations measured by the adjusted Rand index (ARI) (Hubert & Arabie, 1985).

As shown in Figure 18, the seven-cluster solution achieves the highest silhouette score, while the six-cluster solution achieves the highest mean ARI. The seven-cluster solution nevertheless remains

Table 9: Representative diagnoses for the seven Riemannian k-means clusters in Figure 3. Representative diagnoses are selected from the 50% of nodes with the smallest hyperbolic distances to the Fréchet mean of each cluster.

#	Cluster interpretation	Representative diagnoses
1	Broad multisystem conditions	Osteoarthritis; depression; asthma; epilepsy; hiatal hernia
2	Acute systemic complications	Traumatic hemothorax; pleural effusion; heparin-induced thrombocytopenia; chronic lymphoid leukemia; abnormal weight loss
3	Blood / digestive conditions	Acquired coagulation factor deficiency; aplastic anemia; duodenal ulcer; celiac disease; hemorrhoids
4	Diabetes-related conditions	Type 1 diabetes; diabetic ketoacidosis; stage V chronic kidney disease; ankle ulcer; orthostatic hypotension
5	Heart / kidney conditions	Stage III chronic kidney disease; acute systolic heart failure; chronic systolic heart failure; ventricular fibrillation; hypertensive kidney disease
6	Cancer / neurologic / vascular conditions	Lung cancer; prostate cancer; hemiplegia; aortic dissection; cervical spondylosis
7	Respiratory / infectious complications	MRSA infection; pneumonia; bronchiectasis; laryngeal edema; postoperative pulmonary insufficiency

highly stable across random initializations. We therefore use seven clusters for visualization, as it provides the strongest cluster separation, retains high stability, and yields a finer partition for qualitative interpretation. This clustering is used only for interpretation and does not affect model training or evaluation.

Cluster interpretation. In Figure 3, we highlight the 50% of nodes with the smallest hyperbolic distances to the Fréchet mean of each cluster, while the remaining nodes are shown as gray hollow points. The resulting clusters exhibit several recognizable clinical patterns. The central cluster contains broad multisystem conditions, with representative diagnoses including depression, asthma, epilepsy, osteoarthritis, and cardiac arrhythmia, whose co-occurrence contexts span multiple disease systems. The remaining clusters are associated with more specific comorbidity patterns, including acute systemic complications, blood and digestive conditions, diabetes-related conditions, heart and kidney conditions, cancer, neurologic, and vascular conditions, and respiratory and infectious complications. Representative diagnoses for all seven clusters are reported in Table 9.

The cluster names are descriptive summaries and are not intended to define mutually exclusive clinical categories. Because the embeddings are learned from symptom co-occurrence in patient records rather than the ICD-9-CM taxonomy, diagnoses from different ICD families may appear in the same cluster when they share similar clinical contexts.

D.6 Additional MathOverflow-Answers results and analysis

We supplement the MathOverflow-Answers results with optimization diagnostics and an examination of the learned and generated latent geometry. These analyses assess both the stability of model fitting and the extent to which H2G-RD reproduces the broad geometric organization of the fitted hyperlink embeddings while generating new latent samples.

D.6.1 Training and optimization diagnostics

Figures 19 and 20 show the optimization behavior of the embedding and diffusion stages on MathOverflow-Answers. Both H2G-EB and H2G-RD select $K = 0.3$. The embedding-stage objective improves during optimization and becomes approximately stable toward the end of training. For H2G-RD, the raw diffusion loss exhibits the expected stochastic fluctuations, while its exponential moving average decreases and gradually stabilizes, indicating stable optimization of the diffusion model.

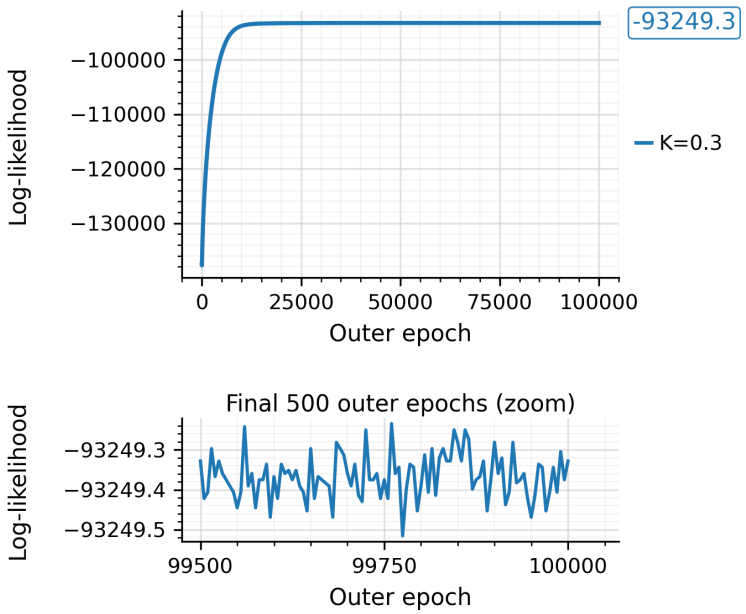

Figure 19: Embedding-stage optimization on MathOverflow-Answers for H2G-EB and H2G-RD. Both implementations select curvature $K = 0.3$. The curves show the optimization trajectories of the fitted latent model, together with an enlarged view of the final training stage to illustrate the behavior near the attained solution.

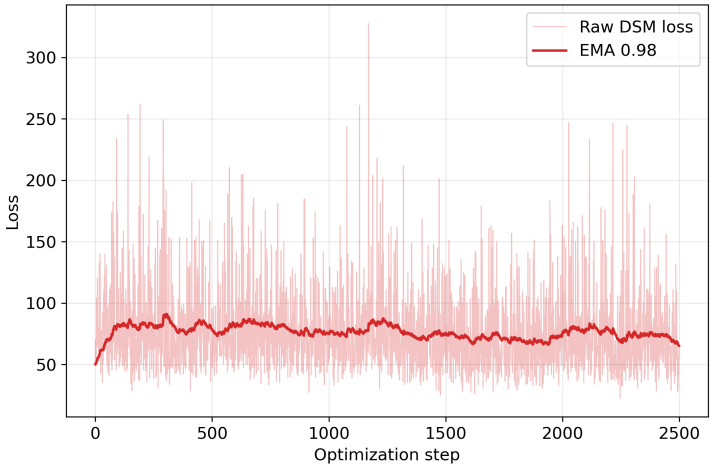

Figure 20: Riemannian diffusion training loss for H2G-RD on MathOverflow-Answers with the selected curvature $K = 0.3$. We report both the per-step loss and its exponential moving average (EMA), with the latter providing a smoother view of the training trajectory.

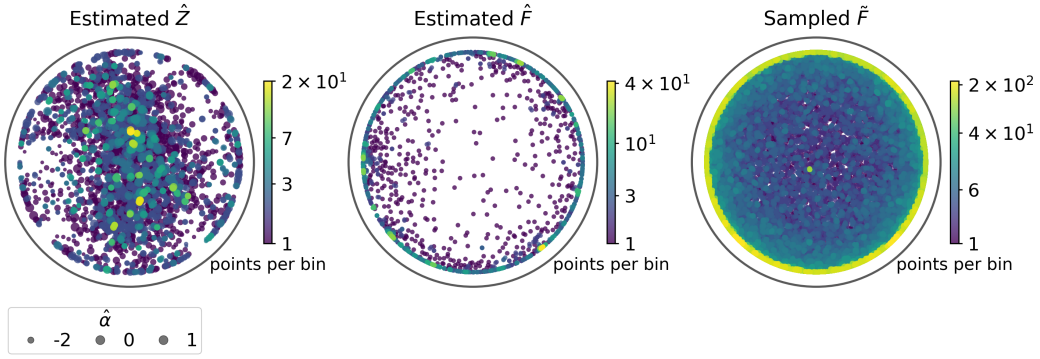

Figure 21: Latent representations learned on MathOverflow-Answers and generated by H2G-RD with the validation-selected curvature $K = 0.3$. Left: estimated node embeddings $\widehat{\mathbf{Z}}$ in the Poincaré disk, where marker size increases with the estimated node effect $\widehat{\alpha}$. Middle: estimated hyperlink embeddings $\widehat{\mathbf{F}}$ obtained from the observed training hypergraph. Right: sampled hyperlink embeddings $\widetilde{\mathbf{F}}$ generated by the fitted Riemannian diffusion model, shown for sampling seed 0. Colors indicate local point density on a logarithmic scale. The generated embeddings extend to previously unseen latent locations while retaining the broad radial organization of the fitted hyperlink embeddings.

D.6.2 Latent geometry of generated hyperlinks

For MathOverflow-Answers and CORUM, H2G-EB and H2G-RD select the same curvature, so we visualize only the H2G-RD generation results; the empirical-bootstrap samples reuse the fitted latent support by construction and therefore provide limited additional geometric information.

Figure 21 shows the learned and generated latent geometry on MathOverflow-Answers. As in MIMIC-III, the estimated hyperlink embeddings $\widehat{\mathbf{F}}$ exhibit a pronounced annular structure, reflecting concentration at similar hyperbolic radii.

A localized high-density region is also visible near the center of the generated embeddings. Similar concentration occurs on other real-world datasets, suggesting that it is associated with the diffusion sampling dynamics rather than dataset-specific semantic structure. Despite this localized mode, the generated embeddings remain broadly distributed across the latent space and yield strong fidelity after decoding.

D.7 Additional CORUM results and analysis

We provide additional analyses of H2G on CORUM from both geometric and biological perspectives. We first report optimization diagnostics and compare the fitted and generated latent hyperlink distributions, and then examine representative generated protein complexes to assess whether their constituent proteins exhibit coherent biological functions or known associations.

D.7.1 Training and optimization diagnostics

Both H2G-EB and H2G-RD select $K = 1$ on CORUM. Since the two implementations share the same likelihood-based embedding stage for a fixed curvature, we report the embedding optimization trajectory once in Figure 22. The log-likelihood increases rapidly during the early outer epochs and subsequently stabilizes, with only small fluctuations near the attained solution.

For H2G-RD, Figure 23 shows the denoising score-matching (DSM) loss during Riemannian diffusion training. The per-step loss exhibits substantial stochastic variation and occasional large spikes because each optimization step is evaluated on randomly sampled mini-batches, diffusion times, and perturbed latent embeddings. Consequently, individual DSM-loss values can vary considerably even when the overall optimization is stable. The exponential moving average (EMA) provides a clearer view of the training trend: after an initial increase, it gradually decreases and becomes relatively stable during the later stages of training.

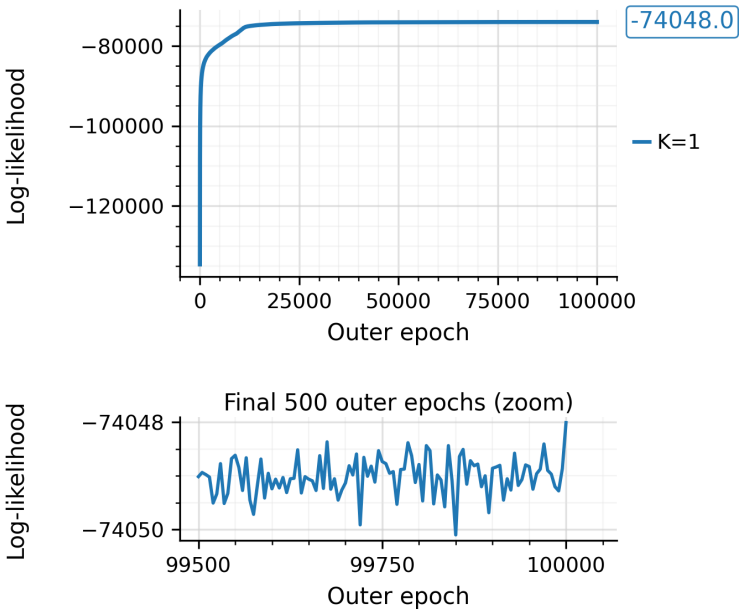

Figure 22: Embedding-stage optimization on CORUM with the validation-selected curvature $K = 1$, shared by H2G-EB and H2G-RD. The top panel shows the log-likelihood over the full optimization trajectory, while the bottom panel magnifies the final 500 outer epochs. The objective improves rapidly at the beginning of training and remains approximately stable near the attained solution.

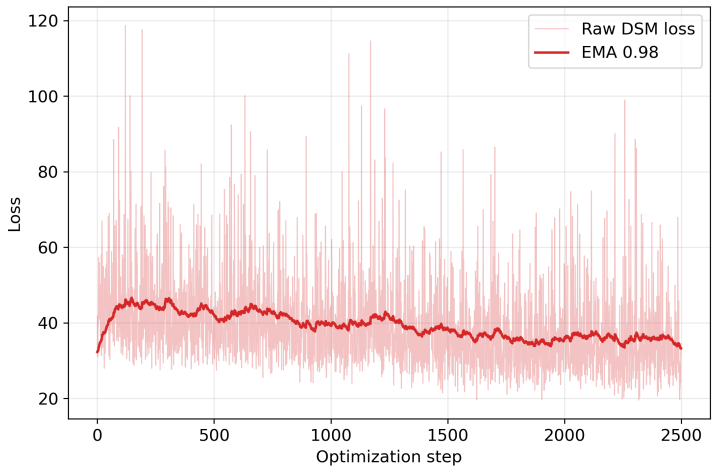

Figure 23: Riemannian diffusion training loss for H2G-RD on CORUM with the validation-selected curvature $K = 1$. The light curve shows the per-step denoising score-matching (DSM) loss, and the solid curve shows its exponential moving average (EMA; smoothing coefficient 0.98). The raw loss is highly variable because the DSM objective is evaluated using stochastic mini-batches, diffusion times, and perturbations, whereas the EMA reveals a gradual decrease and stabilization of the overall training trajectory.

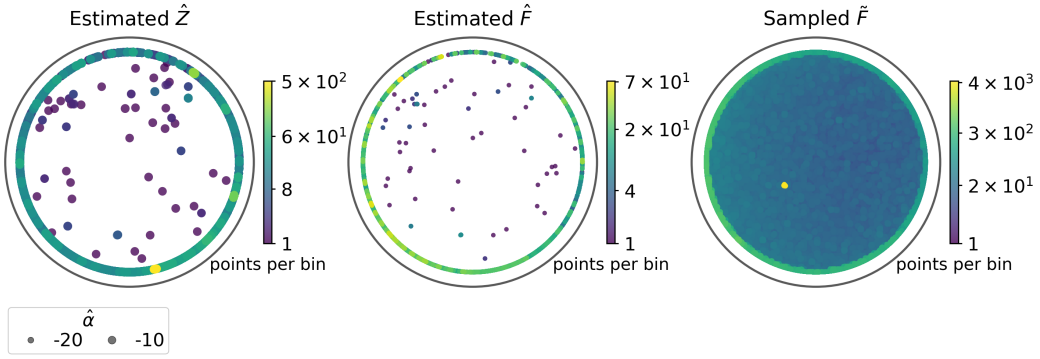

Figure 24: Latent representations learned on CORUM and generated by H2G-RD with the validation-selected curvature $K = 1$. Left: estimated node embeddings $\widehat{\mathbf{Z}}$ in the Poincaré disk, where marker size increases with the estimated node effect $\widehat{\alpha}$. Middle: estimated hyperlink embeddings $\widehat{\mathbf{F}}$ obtained from the observed training hypergraph. Right: sampled hyperlink embeddings $\widetilde{\mathbf{F}}$ generated by the fitted Riemannian diffusion model, shown for sampling seed 0. Colors indicate local point density on a logarithmic scale. The generated embeddings extend to previously unseen latent locations while retaining the broad radial organization of the fitted hyperlink embeddings.

D.7.2 Latent geometry of generated hyperlinks

As with MathOverflow-Answers, H2G-EB and H2G-RD select the same curvature on CORUM, so we visualize only the H2G-RD generation results; the empirical-bootstrap samples reuse the fitted latent support by construction and therefore provide limited additional geometric information.

Figure 24 shows the learned and generated latent geometry on CORUM. As in MIMIC-III and MathOverflow-Answers, the estimated hyperlink embeddings $\widehat{\mathbf{F}}$ exhibit a pronounced annular structure, reflecting concentration at similar hyperbolic radii. The estimated node embeddings $\widehat{\mathbf{Z}}$ are likewise concentrated toward the outer region of the disk.

H2G-RD generates latent hyperlinks beyond the empirical support of $\widehat{\mathbf{F}}$, populating previously unseen locations throughout the disk while retaining substantial mass toward the outer radial region. Thus, the diffusion-based generator is not restricted to the finite support of the fitted hyperlink embeddings.

A localized high-density region is also visible near the center of the generated embeddings. As in the previous datasets, this concentration is more plausibly associated with the diffusion sampling dynamics than with dataset-specific semantic structure. Despite this localized mode, the generated embeddings remain broadly distributed across the latent space and yield strong fidelity after decoding.

D.7.3 Biological interpretation of generated protein complexes

Figure 4 shows three representative protein complexes generated by H2G-RD on CORUM. Their constituent proteins are associated with three well-studied biological functions: gene regulation, cell adhesion and migration, and DNA repair. The labels are descriptive summaries of the generated protein sets. These generated protein sets are absent from the CORUM dataset and therefore represent novel combinations under our generative model. We examine whether their constituent proteins exhibit biologically related functions or known interactions, providing evidence that the generated hyperlinks may serve as plausible candidates for downstream protein-complex investigation.

Gene regulation. The first generated complex contains AR (androgen receptor), RBBP4 (retinoblastoma-binding protein 4), HDAC1 and HDAC2 (histone deacetylases 1 and 2), TAF10 (TFIID subunit 10), and SETDB1 (SET domain bifurcated histone lysine methyltransferase 1). These proteins are broadly involved in controlling which genes are turned on or off. In particular, HDAC1, HDAC2, and RBBP4 are known to act together in protein complexes that regulate gene expression by modifying chromatin (Kelly & Cowley, 2013; Torchy et al., 2015; Millard et al., 2016). AR is a transcription factor whose activity depends on regulatory proteins that control access to

DNA, while SETDB1 also contributes to gene repression through histone modification (Chng et al., 2012; Cho et al., 2014). Thus, although the full generated set is not claimed to be a known complex, its members share a clear role in gene regulation.

Cell adhesion and migration. The second generated complex contains EGFR (epidermal growth factor receptor), ITGB1 (integrin beta-1), ITGB3 (integrin beta-3), ITGAV (integrin alpha-V), and STRN4 (striatin-4). Several of these proteins directly participate in cell adhesion, which allows cells to attach to their surroundings and also affects how cells move. For example, ITGAV and ITGB3 form an integrin receptor, while ITGB1 is also a major component of integrin-mediated adhesion. Integrin signaling is known to interact with EGFR signaling, and this interaction helps regulate cell attachment, movement, and growth (Cabodi et al., 2004; Thomas et al., 2019). STRN4 is a striatin-family scaffold protein with reported roles in regulating cellular morphology through PP2A-associated signaling. The generated set therefore combines proteins with related roles in cell adhesion, signaling, and migration.

DNA repair. The third generated complex contains PARP1 (poly[ADP-ribose] polymerase 1), XRCC6 (DNA repair protein Ku70), MRE11 (double-strand break repair protein MRE11), and KPNA1 (importin subunit alpha-1). PARP1, XRCC6, and MRE11 all have established roles in detecting or repairing damaged DNA. PARP1 helps coordinate repair after DNA strand breaks, XRCC6 is part of the Ku complex used to repair broken DNA ends, and MRE11 is a core component of the machinery that detects and processes DNA double-strand breaks (Ray Chaudhuri & Nussenzweig, 2017; D'amours & Jackson, 2002; Reginato & Cejka, 2020). KPNA1 is primarily involved in transporting proteins into the nucleus and is less specific to DNA repair than the other three proteins. Nevertheless, the generated set contains a clear DNA-repair core supported by established biological studies.

E Proofs of theoretical results

This section provides proofs and supporting results for the theoretical claims in the main text. We first prove the hyperbolic-versus-Euclidean expressivity result and the higher-order dependence property, followed by the curvature-identifiability theorem. We then develop the consistency theory for the fitted generator, including bounds for parameter estimation, latent distribution recovery, and the Riemannian diffusion approximation used in H2G-RD.

E.1 Proof of Proposition A.1

Proof. We prove the hyperbolic and Euclidean statements separately.

Hyperbolic embedding. The $(1+\varepsilon)$-distortion construction of Sarkar (2011, Theorem 5) implies that, for every finite weighted tree, there exist an embedding $\psi : T_{b,L} \to \mathbb{H}_2^{-1}$ and a common scale $\tau > 0$ satisfying

$$\frac{\tau}{1+\varepsilon} d_T(u,v) \le d_1\{\psi(u), \psi(v)\} \le \tau d_T(u,v)$$

for all vertices u, v. Under our normalized curvature convention, $d_K = K^{-1/2} d_1$. Hence, setting $s = \tau/\sqrt{K}$ gives Equation 2 and therefore $D(\psi) \le 1+\varepsilon$.

Euclidean lower bound. Let $\phi : T_{b,L} \to \mathbb{R}^q$ be injective and write $D_\phi = D(\phi)$. Since distortion is invariant to global rescaling, we may assume $\mathrm{coLip}(\phi) = 1$. Thus,

$$d_T(u,v) \le \|\phi(u) - \phi(v)\|_2 \le D_\phi d_T(u,v).$$

Let o be the root and $\mathcal{L}_L$ the set of b^L leaves. Every $u \in \mathcal{L}_L$ satisfies $\|\phi(u) - \phi(o)\|_2 \le D_\phi L$, while any two distinct leaves satisfy $\|\phi(u) - \phi(v)\|_2 \ge d_T(u,v) \ge 2$. Therefore, the open Euclidean unit balls centered at the leaf images are pairwise disjoint and are all contained in the radius-$(D_\phi L + 1)$ ball centered at $\phi(o)$. Writing $v_q = \mathrm{Vol}\{B_q(0,1)\}$, volume comparison gives

$$b^L v_q \le (D_\phi L + 1)^q v_q.$$

Hence

$$D_\phi \geq \frac{b^{L/q} - 1}{L}.$$

Combining this with $D_\phi \geq 1$ proves Equation 3. If $D_\phi \leq D_0$, then $b^{L/q} \leq 1 + D_0 L$, which rearranges to Equation 4. Finally, $N_{b,L} \asymp b^L$ for fixed b, so $L = \Theta\{\log(N_{b,L})\}$ and $\log(1+D_0 L) = \Theta\{\log\log(N_{b,L})\}$, yielding the stated asymptotic dimension bound. □

E.2 Proof of Proposition 1

Proof. Fix $\Theta_n = (\mathcal{Z}_n, \boldsymbol{\alpha}_n, K, \mathbb{P}_{\mathbf{F}})$ and write $Y_i = \mathbf{1}_{\{i \in E\}}$ for each $i \in [n]$. By Equation 1, conditional on $\mathbf{F}$ and Θ_n, the variables $Y_1, \ldots, Y_n$ are independent, with

$$\mathbb{E}[Y_i \mid \mathbf{F}, \Theta_n] = p_i(\mathbf{F}).$$

For any nonempty $e \subseteq [n]$, the event $e \subseteq E$ is equivalent to $Y_i = 1$ for every $i \in e$. Conditional independence and the law of iterated expectations therefore give

$$\begin{aligned}
\mathbb{P}(e \subseteq E \mid \Theta_n) &= \mathbb{E}\left[\prod_{i \in e} Y_i \,\middle|\, \Theta_n\right] \\
&= \mathbb{E}_{\mathbf{F}}\left[\mathbb{E}\left[\prod_{i \in e} Y_i \,\middle|\, \mathbf{F}, \Theta_n\right]\right] \\
&= \mathbb{E}_{\mathbf{F}}\left[\prod_{i \in e} p_i(\mathbf{F})\right] \\
&= \int_{\mathbb{H}_2^{-K}} \prod_{i \in e} p_i(\boldsymbol{f})\, \mathbb{P}_{\mathbf{F}}(\mathrm{d}\boldsymbol{f}).
\end{aligned}$$

This integral is well defined for any probability measure $\mathbb{P}_{\mathbf{F}}$ supported on $\mathbb{H}_2^{-K}$, since the integrand is bounded between zero and one. In particular, no density assumption on $\mathbb{P}_{\mathbf{F}}$ is required.

For distinct $i, k \in [n]$, applying the preceding identity to $\{i\}$, $\{k\}$, and $\{i, k\}$ yields

$$\begin{aligned}
\mathrm{Cov}(Y_i, Y_k \mid \Theta_n) &= \mathbb{E}[Y_i Y_k \mid \Theta_n] - \mathbb{E}[Y_i \mid \Theta_n]\, \mathbb{E}[Y_k \mid \Theta_n] \\
&= \mathbb{E}_{\mathbf{F}}[p_i(\mathbf{F}) p_k(\mathbf{F})] - \mathbb{E}_{\mathbf{F}}[p_i(\mathbf{F})]\, \mathbb{E}_{\mathbf{F}}[p_k(\mathbf{F})] \\
&= \mathrm{Cov}_{\mathbf{F}}\{p_i(\mathbf{F}), p_k(\mathbf{F})\}.
\end{aligned}$$

This completes the proof. □

E.3 Proof of Theorem 1

We first specify the configurations and explain the holomorphic compatibility condition, and then prove the quantitative bound. The proof adapts the Schwarz–Pick argument of Li et al. (2026) to the hyperlink–node distance matrix with fixed node effects.

Fix the node effects $\boldsymbol{\alpha}_n = (\alpha_1, \ldots, \alpha_n)^\top$. Consider two configurations

$$(K, \mathcal{F}_m, \mathcal{Z}_n), \qquad (K', \mathcal{F}'_m, \mathcal{Z}'_n),$$

where

$$\mathcal{F}_m = \{\boldsymbol{f}_j\}_{j=1}^m, \qquad \mathcal{Z}_n = \{\boldsymbol{z}_i\}_{i=1}^n$$

are indexed collections of points in $\mathbb{H}_2^{-K}$, and

$$\mathcal{F}'_m = \{\boldsymbol{f}'_j\}_{j=1}^m, \qquad \mathcal{Z}'_n = \{\boldsymbol{z}'_i\}_{i=1}^n$$

are the corresponding collections in $\mathbb{H}_2^{-K'}$. The node effects are identical across these configurations.

For $\kappa > 0$, write d_κ for the distance on $\mathbb{H}_2^{-\kappa}$, so that $d_K(\boldsymbol{x}, \boldsymbol{y}) = d_{\mathbb{H}_2^{-K}}(\boldsymbol{x}, \boldsymbol{y})$. Define the hyperlink–node distance matrices by

$$D_{ji} = d_K(\boldsymbol{f}_j, \boldsymbol{z}_i), \qquad D'_{ji} = d_{K'}(\boldsymbol{f}'_j, \boldsymbol{z}'_i), \qquad \boldsymbol{D} = (D_{ji}), \quad \boldsymbol{D}' = (D'_{ji}).$$

The corresponding conditional inclusion matrices $\boldsymbol{P} = (P_{ji})$ and $\boldsymbol{P'} = (P'_{ji})$ have entries

$$P_{ji} = \sigma(\alpha_i - D_{ji}), \qquad P'_{ji} = \sigma(\alpha_i - D'_{ji}).$$

All entries belong to $(0, 1)$, so their logits are well defined. The theorem assumes that

$$K, K' \in [c_K^{-1}, c_K], \qquad D_{ji}, D'_{ji} \geq c_l \quad \text{for all } j \in [m],\ i \in [n],$$

where $c_K \geq 1$ and $c_l > 0$.

Identify the Poincaré disk with $\mathbb{D} = \{w \in \mathbb{C} : |w| < 1\}$. Under the normalized hyperboloid parametrization in Section 2.1, the complex-coordinate projection is

$$\zeta(\boldsymbol{x}) = \frac{x_1 + \mathrm{i}x_2}{1 + x_3}, \qquad \boldsymbol{x} = (x_1, x_2, x_3)^\top, \qquad \mathrm{i}^2 = -1.$$

Write

$$u_j = \zeta(\boldsymbol{f}_j), \qquad v_i = \zeta(\boldsymbol{z}_i), \qquad u'_j = \zeta(\boldsymbol{f}'_j), \qquad v'_i = \zeta(\boldsymbol{z}'_i).$$

For $u, v \in \mathbb{D}$, let

$$\rho(u, v) = 2 \operatorname{arctanh} \left| \frac{u - v}{1 - \overline{u}v} \right|,$$

where $\overline{u}$ denotes complex conjugation. This is the curvature--1 hyperbolic distance on the disk. Our curvature convention gives

$$d_\kappa(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{\kappa}} \rho\big(\zeta(\boldsymbol{x}), \zeta(\boldsymbol{y})\big), \qquad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{H}_2^{-\kappa}. \tag{5}$$

Consequently,

$$\sqrt{K}\, D_{ji} = \rho(u_j, v_i), \qquad \sqrt{K'}\, D'_{ji} = \rho(u'_j, v'_i).$$

A holomorphic self-map $\phi : \mathbb{D} \to \mathbb{D}$ is complex differentiable at every point of $\mathbb{D}$ and takes values in $\mathbb{D}$; it need not be surjective or invertible. When $K < K'$, the theorem requires a single such map satisfying

$$\phi(u_j) = u'_j, \quad j \in [m], \qquad \phi(v_i) = v'_i, \quad i \in [n].$$

The same map must therefore match both the hyperlink and node embeddings simultaneously. Separate maps for the two embedding sets do not satisfy this condition. When $K' < K$, the corresponding map is required in the reverse direction. The map may depend on the pair of configurations being compared.

Proof. If $K = K'$, the claimed inequality is immediate. Otherwise, by interchanging the two configurations if necessary, we may assume that $K < K'$.

Let $\phi : \mathbb{D} \to \mathbb{D}$ be the holomorphic map supplied by the compatibility condition. The Schwarz–Pick theorem states that

$$\left| \frac{\phi(u) - \phi(v)}{1 - \overline{\phi(u)}\phi(v)} \right| \leq \left| \frac{u - v}{1 - \overline{u}v} \right|, \qquad u, v \in \mathbb{D}.$$

Since $\operatorname{arctanh}$ is increasing on $[0, 1)$,

$$\rho\big(\phi(u), \phi(v)\big) \leq \rho(u, v).$$

Applying this inequality to each hyperlink–node pair and using Equation 5, we obtain

$$\begin{aligned} \sqrt{K'}\, D'_{ji} &= \rho(u'_j, v'_i) \\ &= \rho\big(\phi(u_j), \phi(v_i)\big) \\ &\leq \rho(u_j, v_i) = \sqrt{K}\, D_{ji}. \end{aligned}$$

Therefore,

$$D'_{ji} \leq \sqrt{\frac{K}{K'}}\, D_{ji}, \qquad j \in [m],\ i \in [n]. \tag{6}$$

Combining Equation 6 with $D_{ji} \geq c_l$ gives

$$\begin{aligned} D_{ji} - D'_{ji} &\geq \left(1 - \sqrt{\frac{K}{K'}}\right) D_{ji} \\ &\geq c_l \left(1 - \sqrt{\frac{K}{K'}}\right) \\ &= \frac{c_l(K' - K)}{\sqrt{K'}(\sqrt{K'} + \sqrt{K})} \\ &\geq \frac{c_l}{2c_K}(K' - K). \end{aligned}$$

For the last inequality, we used $K, K' \leq c_K$, which implies $\sqrt{K'}(\sqrt{K'} + \sqrt{K}) \leq 2c_K$. Both sides are nonnegative. Squaring and summing over all hyperlink–node pairs yields

$$\frac{1}{mn}\|\boldsymbol{D} - \boldsymbol{D}'\|_F^2 \geq \frac{c_l^2}{4c_K^2}(K - K')^2. \tag{7}$$

Since $\operatorname{logit}\{\sigma(x)\} = x$ and the node effects are identical across configurations,

$$\begin{aligned} \operatorname{logit}(P_{ji}) - \operatorname{logit}(P'_{ji}) &= (\alpha_i - D_{ji}) - (\alpha_i - D'_{ji}) \\ &= -(D_{ji} - D'_{ji}). \end{aligned} \tag{8}$$

Thus,

$$\|\operatorname{logit}(\boldsymbol{P}) - \operatorname{logit}(\boldsymbol{P}')\|_F^2 = \|\boldsymbol{D} - \boldsymbol{D}'\|_F^2.$$

Combining this identity with Equation 7 gives

$$\frac{1}{mn}\|\operatorname{logit}(\boldsymbol{P}) - \operatorname{logit}(\boldsymbol{P}')\|_F^2 \geq \frac{c_l^2}{4c_K^2}(K - K')^2.$$

Taking square roots and rearranging proves

$$|K - K'| \leq \frac{2c_K}{c_l\sqrt{mn}}\|\operatorname{logit}(\boldsymbol{P}) - \operatorname{logit}(\boldsymbol{P}')\|_F.$$

In particular, $\boldsymbol{P} = \boldsymbol{P}'$ implies $K = K'$. The case $K' < K$ follows by reversing the roles of the two configurations. □

E.4 Consistency of the hyperbolic hypergraph generator

We develop the consistency argument in three steps. First, we place the true and fitted models in common coordinates and derive a general bound that decomposes hyperlink-generation error into parameter-estimation error, empirical latent-distribution error, and latent-generator approximation error. We then give sufficient conditions under which the likelihood-based embedding stage is consistent. Finally, we relate the approximation error of H2G-RD to initialization, score-estimation, smoothing, and numerical discretization errors of the Riemannian diffusion model.

Write $\mathcal{M} = \mathbb{H}_2^{-1}$ and d_1 for its distance. Under our normalized hyperboloid convention, all curvatures use the same point set, with $d_K = K^{-1/2}d_1$; the metrics are not identified as isometric when the curvatures differ. Set $\vartheta = (K, \boldsymbol{\alpha}_n, \mathcal{Z}_n)$ and

$$q_\vartheta(e \mid \mathbf{f}) = \prod_{i \in e} \sigma\{\alpha_i - d_K(\mathbf{f}, \mathbf{z}_i)\} \prod_{i \notin e} [1 - \sigma\{\alpha_i - d_K(\mathbf{f}, \mathbf{z}_i)\}], \qquad e \subseteq [n].$$

Let $P = P_{\mathbf{F}}$ and $P_E(e) = \int q_\vartheta(e \mid \mathbf{f}) P(d\mathbf{f})$. The fitted generator defines the random probability measure

$$\widehat{P}_E(e) = \int q_{\widehat{\vartheta}}(e \mid \mathbf{f}) \widehat{P}_{\mathbf{F}}(d\mathbf{f}).$$

All fitted laws below are conditional on the training data and any training randomness; stochastic orders include this randomness. All latent Wasserstein distances use d_1, and

$$W_2^2(\mu, \nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int d_1(\mathbf{f}, \mathbf{g})^2 \, \gamma(d\mathbf{f}, d\mathbf{g}),$$

where $\Pi(\mu, \nu)$ denotes the set of couplings.

Condition 1 (Geometric boundedness)**.** *The true hyperlink embeddings $\mathbf{f}_1, \ldots, \mathbf{f}_m$ are independent draws from P. For constants $R < \infty$ and $0 < k_- < k_+ < \infty$, the support of P and all true node embeddings lie in $B_{d_1}(o, R)$, and $K, \widehat{K} \in [k_-, k_+]$. The fitted latent distribution has a finite second moment under d_1. These constants do not depend on m, n.*

Let $\mathcal{I}$ be the full isometry group of $(\mathcal{M}, d_1)$, and define

$$r_{m,n}^2 = \frac{1}{n}\sum_{i=1}^{n}(\widehat{\alpha}_i - \alpha_i)^2 + |\widehat{K}^{-1/2} - K^{-1/2}|^2 + \inf_{T\in\mathcal{I}}\left\{\frac{1}{m}\sum_{j=1}^{m} d_1(T\widehat{\mathbf{f}}_j, \mathbf{f}_j)^2 + \frac{1}{n}\sum_{i=1}^{n} d_1(T\widehat{\mathbf{z}}_i, \mathbf{z}_i)^2\right\}, \tag{9}$$

$$P_m = \frac{1}{m}\sum_{j=1}^{m}\delta_{\mathbf{f}_j}, \qquad \widehat{\mu}_m = \frac{1}{m}\sum_{j=1}^{m}\delta_{\widehat{\mathbf{f}}_j}, \qquad \varepsilon_m = W_2(\widehat{P}_{\mathbf{F}}, \widehat{\mu}_m). \tag{10}$$

The same isometry must align both sets of embeddings and the fitted latent distribution. The reconstruction error ε_m is unchanged by this joint transformation. If node effects are known, the first term in Equation 9 is zero.

The following identity is the KL chain rule underlying the DDE reduction (Wu et al., 2025); it does not depend on Euclidean geometry. After a common alignment, write $Q = T_{\#}\widehat{P}_{\mathbf{F}}$ and

$$q_{\widehat{\vartheta}}^{T}(e \mid \mathbf{f}) = q_{\widehat{\vartheta}}(e \mid T^{-1}\mathbf{f}).$$

For the joint laws $\mathbb{P}(d\mathbf{f}, e) = P(d\mathbf{f})q_{\vartheta}(e \mid \mathbf{f})$ and $\widehat{\mathbb{P}}(d\mathbf{f}, e) = Q(d\mathbf{f})q_{\widehat{\vartheta}}^{T}(e \mid \mathbf{f})$, we have

$$D_{\mathrm{KL}}(\mathbb{P}\|\widehat{\mathbb{P}}) = D_{\mathrm{KL}}(P\|Q) + \int D_{\mathrm{KL}}\{q_{\vartheta}(\cdot \mid \mathbf{f})\|q_{\widehat{\vartheta}}^{T}(\cdot \mid \mathbf{f})\}P(d\mathbf{f}). \tag{11}$$

When $P \ll Q$, factor the joint Radon–Nikodym derivative into its latent and conditional factors and integrate its logarithm. If $P \not\ll Q$, both sides are infinite, so the identity holds in the extended-real sense. Marginalization gives

$$D_{\mathrm{KL}}(P_E\|\widehat{P}_E) \le D_{\mathrm{KL}}(\mathbb{P}\|\widehat{\mathbb{P}}).$$

With the true decoder, the second term is zero, giving the ideal DDE identity for the joint laws and an inequality for the hyperlink laws. For empirical bootstrap and nonatomic P, however, $D_{\mathrm{KL}}(P\|Q) = \infty$. We therefore use the following coupling bound, which does not require latent absolute continuity.

Proposition E.1 (A bound valid for both EB and RD)**.** *Under Condition 1, for a constant C depending only on R, k_-, k_+,*

$$\frac{1}{n}D_{\mathrm{KL}}(P_E\|\widehat{P}_E) \le C\{r_{m,n}^2 + W_2^2(P, P_m) + \varepsilon_m^2\}. \tag{12}$$

Moreover, $\mathbb{E}W_2^2(P, P_m) \le C_R m^{-1/2}$. Consequently, Theorem 2 holds.

Proof. Set $b(u) = \log(1 + e^u)$. Since $0 < b''(u) \le 1/4$,

$$D_{\mathrm{KL}}\{\mathrm{Ber}(\sigma(u))\|\,\mathrm{Ber}(\sigma(v))\} = b(v) - b(u) - b'(u)(v - u) \le \frac{1}{8}(u - v)^2. \tag{13}$$

Fix $T \in \mathcal{I}$ and any $\gamma \in \Pi(P, Q)$. On $(\mathbf{f}, \mathbf{g}, e)$, define two laws with common $(\mathbf{f}, \mathbf{g})$ marginal γ, and respective conditional laws $q_{\vartheta}(e \mid \mathbf{f})$ and $q_{\widehat{\vartheta}}^{T}(e \mid \mathbf{g})$. Their hyperlink marginals are P_E and $\widehat{P}_E$. The KL chain rule, marginalization, and conditional product structure give

$$D_{\mathrm{KL}}(P_E\|\widehat{P}_E) \le \frac{1}{8}\int\sum_{i=1}^{n}\left[\alpha_i - K^{-1/2}d_1(\mathbf{f}, \mathbf{z}_i) - \widehat{\alpha}_i + \widehat{K}^{-1/2}d_1(\mathbf{g}, T\widehat{\mathbf{z}}_i)\right]^2 d\gamma. \tag{14}$$

Write $c = K^{-1/2}$ and $\widehat{c} = \widehat{K}^{-1/2}$. The absolute value of the ith summand before squaring is at most

$$|\widehat{\alpha}_i - \alpha_i| + \widehat{c}\,d_1(\mathbf{g}, \mathbf{f}) + \widehat{c}\,d_1(T\widehat{\mathbf{z}}_i, \mathbf{z}_i) + 2R|\widehat{c} - c|.$$

This follows from the triangle inequality and $d_1(\mathbf{f}, \mathbf{z}_i) \le 2R$ for P-almost every $\mathbf{f}$. In particular, no bounded-support assumption on generated embeddings is needed. Apply $(a_1 + \cdots + a_4)^2 \le 4\sum_{k=1}^4 a_k^2$ in Equation 14, then minimize over γ. Furthermore, by the triangle inequality and the empirical pairing,

$$W_2(P, Q) \le W_2(P, P_m) + \left\{ \frac{1}{m} \sum_{j=1}^{m} d_1(\mathbf{f}_j, T\widehat{\mathbf{f}}_j)^2 \right\}^{1/2} + \varepsilon_m.$$

Squaring this bound and taking the infimum over T proves Equation 12.

For the empirical term, use the global normal-coordinate map $\log_o : \mathcal{M} \to \mathbb{R}^2$. The exponential map is Lipschitz on the Euclidean radius-R ball, so transporting the Euclidean empirical measures gives

$$W_2^2(P, P_m) \le L_R^2 W_{2,\mathbb{R}^2}^2\{(\log_o)_\# P, (\log_o)_\# P_m\}.$$

The transformed distribution has bounded support. Apply Fournier & Guillin (2015, Theorem 1, with $p = 2, d = 2$) and then Markov's inequality. This proves the asserted expectation bound and Theorem 2. □

Here consistency means $n^{-1} D_{\mathrm{KL}}(P_E \| \widehat{P}_E) \to 0$ in probability. It must not be equated with full total-variation consistency when n grows. By Pinsker's inequality, the stronger condition

$$n\{r_{m,n}^2 + \varepsilon_m^2 + m^{-1/2}\} = o_p(1)$$

is sufficient for $d_{\mathrm{TV}}(P_E, \widehat{P}_E) = o_p(1)$, where $d_{\mathrm{TV}}(\mu, \nu) = \sup_A |\mu(A) - \nu(A)|$. The theorem concerns the fitted distribution of one new hyperlink, not its finite Monte Carlo approximation. For $\widetilde{m}$ conditionally independent generated hyperlinks, the KL divergence of the ordered joint laws is $\widetilde{m}$ times the single-hyperlink KL divergence; increasing the number of generated samples does not remove estimation error.

E.4.1 A sufficient condition for embedding consistency

The following result supplies a conditional likelihood-estimation rate rather than importing the Euclidean inner-product rate from DDE. Its quantitative identifiability assumption is additional and must be verified for the latent configurations under consideration.

Let $N = mn$, $\xi = (K, \boldsymbol{\alpha}_n, \mathcal{F}_m, \mathcal{Z}_n)$, and write ξ_0 for the truth. Define $\eta_{ji}(\xi) = \alpha_i - K^{-1/2} d_1(\mathbf{f}_j, \mathbf{z}_i)$. Let $\mathcal{T}_{m,n}$ be the parameter class with all embeddings in $B_{d_1}(o, R)$, $K \in [k_-, k_+]$, and $|\alpha_i - a_{m,n}| \le A$, where A is fixed and $a_{m,n} \le a_+$ is a deterministic sequence. Set $\rho_{m,n} = \min\{1, e^{a_{m,n}}\}$. The dense case is included by taking $a_{m,n}$ bounded. Let $r^2(\xi, \xi_0)$ denote Equation 9 with ξ in place of the estimates.

Condition 2 (Quantitative identifiability)**.** *The true parameter belongs to $\mathcal{T}_{m,n}$. On events whose probabilities tend to one under the latent sample, a constant $c_{\mathrm{id}} > 0$, independent of m, n, satisfies*

$$r^2(\xi, \xi_0) \le \frac{c_{\mathrm{id}}}{mn} \sum_{j=1}^{m} \sum_{i=1}^{n} \{\eta_{ji}(\xi) - \eta_{ji}(\xi_0)\}^2, \qquad \xi \in \mathcal{T}_{m,n}. \tag{15}$$

This condition is stronger than identification up to a common isometry: it excludes configurations for which curvature, node effects, or latent positions are arbitrarily difficult to distinguish.

Proposition E.2 (Conditional MLE consistency)**.** *Assume Conditions 1 and 2, and conditional independence of the incidence entries given the latent embeddings. Suppose $\widehat{\xi} \in \mathcal{T}_{m,n}$ satisfies*

$$\ell(\widehat{\xi}) \ge \sup_{\xi \in \mathcal{T}_{m,n}} \ell(\xi) - mn\omega_{m,n}, \qquad \omega_{m,n} \ge 0.$$

Then

$$r_{m,n}^2 = O_p\left\{ \frac{(m+n)\log(mn)}{mn\rho_{m,n}} + \frac{\omega_{m,n}}{\rho_{m,n}} \right\}. \tag{16}$$

In particular, the embeddings, node effects, and curvature are consistent in this averaged, aligned sense if $(m \wedge n)\rho_{m,n} / \log(mn) \to \infty$ and $\omega_{m,n} = o_p(\rho_{m,n})$.

Proof. Condition on the true latent sample. Put

$$S(\xi) = \sum_{j,i} \{\eta_{ji}(\xi) - \eta_{ji}(\xi_0)\}^2, \qquad Z(\xi) = \sum_{j,i} (H_{ji} - p^0_{ji})\{\eta_{ji}(\xi) - \eta_{ji}(\xi_0)\}.$$

The parameter constraints imply that all natural-parameter differences are bounded by a constant and that $b''(u) \asymp \rho_{m,n}$ between each pair of admissible natural parameters. Hence, for constants $c, C > 0$,

$$\ell(\xi) - \ell(\xi_0) \le Z(\xi) - c\rho_{m,n} S(\xi), \qquad \mathrm{Var}\{Z(\xi) \mid \mathcal{F}_m\} \le C\rho_{m,n} S(\xi).$$

A radius-R hyperbolic ball has an ϵ-net of cardinality at most $C\epsilon^{-2}$. The triangle inequality and the bounded curvature interval therefore give a net of natural-parameter arrays with maximum-entry accuracy N^{-1} and log-cardinality $L_N \le C(m+n)\log N$. Bernstein's inequality and a union bound show that, for every $t \ge 1$, with conditional probability at least $1 - e^{-t}$, simultaneously over the net,

$$Z(\xi) \le \tfrac{c}{2}\rho_{m,n} S(\xi) + C(L_N + t).$$

Indeed, the Bernstein square-root term is absorbed into $c\rho_{m,n}S(\xi)/2 + C(L_N + t)$ by Young's inequality. For each H_{ji}, the function $u \mapsto H_{ji}u - b(u)$ is 1-Lipschitz, so replacing $\widehat{\xi}$ by its net representative changes ℓ by at most one. Also, $S(\widehat{\xi}) \le 2S(\xi_{\mathrm{net}}) + 2/N$. The approximate maximization inequality consequently yields

$$\frac{S(\widehat{\xi})}{N} \le C\left\{\frac{L_N + t + 1}{N\rho_{m,n}} + \frac{\omega_{m,n}}{\rho_{m,n}}\right\}.$$

Apply Equation 15 and remove the conditioning. Since $K \mapsto K^{-1/2}$ has a Lipschitz inverse on the specified compact interval, the curvature-error term also gives consistency of $\widehat{K}$ itself. □

When both true and estimated node effects satisfy the above band constraint, $b''(u) \le C\rho_{m,n}$ between the true and generated natural parameters, even for generated embeddings outside the ball. Thus the right side of Equation 12 gains a factor $\rho_{m,n}$. Combining this refinement with Proposition E.2 gives

$$\frac{1}{n} D_{\mathrm{KL}}(P_E \| \widehat{P}_E) = O_p\left\{\frac{(m+n)\log(mn)}{mn} + \omega_{m,n} + \rho_{m,n}(m^{-1/2} + \varepsilon_m^2)\right\}.$$

This is analogous to the sparsity cancellation in DDE's normalized decoder-error bound. The likelihood-gap condition concerns an approximately global maximizer. A small change between successive optimization iterates does not establish this condition. Curvature-grid approximation and numerical optimization error must both be included in $\omega_{m,n}$ when applying the result to a grid-based implementation.

E.4.2 Riemannian diffusion: existing results and a sufficient bound

De Bortoli et al. (2022, Theorem 4.1) establish a W_1 sampling bound on compact manifolds with a smooth positive target and uniformly accurate scores. Xu et al. (2026, Theorem 1) obtain an early-stopped TV bound of the form

$$\epsilon + C\epsilon_{\mathrm{score}} + \sqrt{hT}\,\mathrm{poly}(d, G, \tau^{-1}),$$

under compactness and their geometric, score, and sampler assumptions. Here h is the step size, $\tau > 0$ is the early-stopping time, d is the intrinsic dimension, and G collects geometric bounds; the horizon is chosen large enough to control initialization by ϵ. These results do not directly cover the noncompact hyperbolic space. Huang et al. (2022) develop the Riemannian score-matching framework on general manifolds, including hyperbolic examples. For our setting, we give a separate sufficient bound below, with nonexplosion, integrability, and numerical approximation stated explicitly.

Condition on the fitted embeddings and curvature. Work on $\mathcal{M}_{\widehat{K}} = \mathbb{H}_2^{-\widehat{K}}$ with metric $g_{\widehat{K}}$. Let $X_0 \sim \widehat{\mu}_m$ and

$$\mathrm{d}X_t = -\tfrac{1}{2}\beta(t)\nabla_{\mathcal{M}} U(X_t)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}B_t^{\mathcal{M}}.$$

Assume β is continuous and positive, and

$$\pi_U(dx) = Z_U^{-1} e^{-U(x)} \operatorname{vol}_{\widehat{K}}(dx)$$

is a probability measure. Write $\mu_t = \mathcal{L}(X_t)$ and p_t for its Riemannian density at $t > 0$. For $0 < \tau < T$, let $Q_{\tau,T}$ be the law at time $T - \tau$ of the learned reverse diffusion

$$\mathrm{d}Y_u = \beta(T-u)\left\{\tfrac{1}{2}\nabla_{\mathcal{M}} U(Y_u) + s_\theta(Y_u, T-u)\right\}\mathrm{d}u + \sqrt{\beta(T-u)}\,\mathrm{d}\overline{B}_u^{\mathcal{M}}, \qquad Y_0 \sim \pi_U.$$

Define the initialization, score, smoothing, and discretization errors by

$$\begin{aligned} I_T &= D_{\mathrm{KL}}(\mu_T \| \pi_U), \\ S_{\tau,T} &= \int_\tau^T \beta(t)\mathbb{E}_{X_t\sim\mu_t}\|s_\theta(X_t,t) - \nabla_{\mathcal{M}} \log p_t(X_t)\|^2_{g_{\widehat{K}}}\,\mathrm{d}t, \\ b_\tau^2 &= \mathbb{E}[d_1(X_\tau, X_0)^2], \qquad \delta_{\mathrm{disc}} = W_2(\widehat{P}_{\mathbf{F}}, Q_{\tau,T}). \end{aligned}$$

The Wasserstein distances and radial moments here still use d_1. Their conversion to the fitted metric only changes constants because $\widehat{K} \in [k_-, k_+]$.

Proposition E.3 (A sufficient RD consistency bound)**.** *Assume the forward diffusion, its exact time reversal on $[\tau, T]$, and the learned reverse diffusion are nonexplosive and well posed. Assume p_t is smooth and positive for $t > 0$, $I_T, S_{\tau,T} < \infty$, and the Girsanov change-of-drift exponential is a true martingale with integrable log-likelihood (for example, under the corresponding Novikov condition). Then*

$$D_{\mathrm{KL}}(\mu_\tau \| Q_{\tau,T}) \le I_T + \tfrac{1}{2} S_{\tau,T}. \tag{17}$$

If, in addition,

$$\int d_1(o,x)^4 \mu_\tau(dx) + \int d_1(o,x)^4 Q_{\tau,T}(dx) \le M_4$$

for a uniform constant M_4, then

$$\varepsilon_m^2 \le 3b_\tau^2 + C_{M_4}(I_T + S_{\tau,T})^{1/4} + 3\delta_{\mathrm{disc}}^2. \tag{18}$$

Consequently, $b_\tau + I_T + S_{\tau,T} + \delta_{\mathrm{disc}} = o_p(1)$ and $r_{m,n} = o_p(1)$ imply normalized KL consistency of H2G-RD.

Proof. The exact reverse process starts from μ_T and replaces s_θ by $\nabla_{\mathcal{M}} \log p_t$. Let $\mathbb{R}$ and $\mathbb{Q}$ be its path law and the learned reverse path law, respectively. Their diffusion coefficients agree, and their drift difference is $\beta(t)(\nabla_{\mathcal{M}} \log p_t - s_\theta)$. Girsanov's formula, applied on the orthonormal frame bundle, gives

$$D_{\mathrm{KL}}(\mathbb{R}\|\mathbb{Q}) = D_{\mathrm{KL}}(\mu_T\|\pi_U) + \frac{1}{2}\int_\tau^T \beta(t)\mathbb{E}_{\mu_t}\|\nabla_{\mathcal{M}} \log p_t - s_\theta\|^2_{g_{\widehat{K}}}\,\mathrm{d}t.$$

One may equivalently apply Girsanov first with common initial law μ_T, and then change the initial law to π_U. Marginalization at the terminal time proves Equation 17.

For completeness, let $\Delta = d_{\mathrm{TV}}(\mu_\tau, Q_{\tau,T})$. A maximal coupling (V, W) satisfies $\mathbb{P}(V \ne W) = \Delta$. The triangle inequality and Cauchy–Schwarz yield

$$W_2^2(\mu_\tau, Q_{\tau,T}) \le \mathbb{E}[d_1(V,W)^2 \mathbf{1}_{\{V\ne W\}}] \le 2\sqrt{2M_4}\,\Delta^{1/2}.$$

Pinsker's inequality and Equation 17 therefore give

$$W_2^2(\mu_\tau, Q_{\tau,T}) \le C_{M_4}(I_T + S_{\tau,T})^{1/4}.$$

Finally,

$$W_2(\widehat{\mu}_m, \widehat{P}_{\mathbf{F}}) \le b_\tau + W_2(\mu_\tau, Q_{\tau,T}) + \delta_{\mathrm{disc}},$$

and squaring proves Equation 18. The final assertion follows from Theorem 2. □

Suppose π_U satisfies

$$D_{\mathrm{KL}}(\nu\|\pi_U) \le (2\lambda)^{-1} \int \|\nabla \log(d\nu/d\pi_U)\|^2 \, d\nu,$$

a logarithmic Sobolev inequality. Under integration-by-parts conditions, entropy dissipation for the forward generator gives, for $0 < t_0 < T$,

$$I_T \le \exp\left\{-\lambda \int_{t_0}^{T} \beta(t)\,\mathrm{d}t\right\} D_{\mathrm{KL}}(\mu_{t_0}\|\pi_U).$$

Indeed, the entropy derivative is minus $\beta(t)/2$ times relative Fisher information. The initial time $t_0 > 0$ avoids assigning a finite KL divergence to an empirical initial measure relative to a smooth reference law.

The score error in Proposition E.3 is relative to the *exact* marginal score of the noised empirical embedding law. It includes approximation bias from any approximate transition-score target, such as a Varadhan approximation, as well as statistical and optimization errors. Decreasing the numerical step size alone does not remove these errors. The proposition does not assert a discretization rate for an unspecified numerical scheme. Conditioning on $\widehat{\mu}_m$ avoids assuming that the jointly estimated embeddings are independent.