Categorical Flow Matching via Simplex-to-Euclidean Bijections

Bernardo Williams* University of Helsinki

Victor M. Yeom-Song Aalto University

Arto Klami University of Helsinki

Abstract

We propose a simple framework for learning and sampling from probability distributions supported on the simplex. Our approach maps the open simplex to the Euclidean space via smooth bijections, so we can model the density in the Euclidean space. This density is linked to categorical observations via a Dirichlet interpolation. Compared to previous methods that operate on the simplex using Riemannian geometry or custom noise processes, our approach is simpler while achieving competitive performance on both synthetic and real-world datasets.

1 Introduction

We study the problem of learning and generating samples from a probability distribution $\Pr(X)$ defined on the unit simplex, typically for categorical data and normalized probability vectors. We learn such distributions from categorical observations, e.g. for generation of high-dimensional discrete data.

The core difficulties in extending continuous generative models for this case relate to the geometry of the simplex, with recent solutions that can be categorized in three groups. Most relevant to our work are methods that define the density directly on the simplex, e.g. by constructing custom marginal distributions that define a diffusion process [Avdeyev et al., 2023, Floto et al., 2023] or a vector field [Stark et al., 2024, Dunn and Koes, 2024, Tang et al., 2025], or by a bijection between the simplex and the sphere [Davis et al., 2024, Cheng et al., 2024, 2025]. The second group trains models directly in the ambient space $\mathbb{R}^K$ for data with K classes, and obtains discrete observations by progressively pushing continuous samples toward the vertices of the simplex [Chen et al., 2023, Eijkelboom et al., 2024, Sahoo et al., 2025]. The third line of work models stochastic transitions between categorical states on the simplex, leading to discrete flow and diffusion models [Austin et al., 2021, Campbell et al., 2024, Gat et al., 2024], as well as masked diffusion models [Sahoo et al., 2024].

We propose a new, simpler, approach for modeling categorical data. We map the *interior* of the simplex to the Euclidean space using a smooth bijection, train a standard continuous generative model in that space, and then map the samples back. The discrete observations, that would naturally lie at the border of the simplex not covered by our bijection, are first mapped into interior points via a Dirichlet interpolation, so that the original category can be retrieved by a simple argmax-operation.

The main advantage of our approach is that it enables the direct use of standard Euclidean-space generative models, while ensuring that the resulting distribution remains supported on the simplex. This is in stark contrast to prior work, which constructs flows on simplex-constrained distributions or Riemannian manifolds. In this work we use Flow Matching [Lipman et al., 2023, Albergo and Vanden-Eijnden, 2023] as a concrete example of such a model, and we consider two alternative bijections commonly used in Bayesian modeling in other contexts, the stick-breaking transform

*bernardo.williamsmoreno@helsinki.fi

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Frontiers in Probabilistic Inference: Sampling Meets Learning.

[Aitchison, 1982] and the isometric logratio transform [Egozcue et al., 2003]. Both define a geometry on the simplex, and unlike previous methods, our approach explicitly connects this geometry to the Euclidean space. Moreover, they are computationally lightweight and easy to implement.

We validate our method on synthetic and real-world discrete data sets, showing improved density estimation and sampling over existing simplex-based generative models, and discuss future development for improving and validating the approach.

2 Background

Conditional Flow Matching Conditional Flow Matching (CFM) [Lipman et al., 2023] is a continuous-time normalizing flow for generative modeling of continuous data, where a vector field $\boldsymbol{u}_t(\boldsymbol{x}) : \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$ defines the ODE $\frac{d\boldsymbol{x}_t}{dt} = \boldsymbol{u}_t(\boldsymbol{x}_t)$, with marginals $\boldsymbol{x}_t \sim p_t$ related to $\boldsymbol{u}_t$ by the continuity equation (Eq. 26 in Lipman et al. 2023). The vector field transports samples from a simple base distribution p_0 to the target distribution $p_1 = p_{\text{data}}$. Since the true velocity field $\boldsymbol{u}_t$ is not available, training instead regresses a parametric model $\boldsymbol{v}_t^\theta(x_t)$ toward the conditional velocity $\boldsymbol{u}_t(\boldsymbol{x}_t \mid \boldsymbol{x}_0, \boldsymbol{x}_1)$ along paths interpolating between p_0 and p_1. We use the linear interpolation $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$, where $\boldsymbol{x}_0 \sim p_0$ and $\boldsymbol{x}_1 \sim p_1$, which leads to the CFM objective:

$$\mathcal{L}_{\text{CFM}}(\boldsymbol{\theta}) = \mathbb{E}_{t\sim\text{Unif}(0,1),\, \boldsymbol{x}_0,\boldsymbol{x}_1\sim\pi(\boldsymbol{x}_0,\boldsymbol{x}_1)} \left[\left\| \boldsymbol{v}_t^\theta\big(\boldsymbol{x}_t\big) - \boldsymbol{u}_t(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{x}_1) \right\|^2 \right]. \tag{1}$$

Here the target conditional velocity is $\boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0,\boldsymbol{x}_1) = \dot{\boldsymbol{x}}_t = \boldsymbol{x}_1 - \boldsymbol{x}_0$. We consider both the independent coupling $\pi(\boldsymbol{x}_0,\boldsymbol{x}_1) = p_0(\boldsymbol{x}_0)p_1(\boldsymbol{x}_1)$ and the minibatch optimal transport (OT) coupling [Tong et al., 2024], which approximates the optimal pairing and reduces variance in the learned flow.

Riemannian Flow Matching on the Simplex Cheng et al. [2024] and Davis et al. [2024] proposed a generalization of Flow Matching for distributions on the simplex, called Statistical Flow Matching (SFM). The basic idea is to map the simplex to a well-behaving space, learn the flow there, and then map the samples back.

Specifically, let $D := K-1$, they transform the simplex $\Delta^D := \{\boldsymbol{x} \in \mathbb{R}^K : \boldsymbol{x}_k \geq 0,\, \sum_{k=1}^K \boldsymbol{x}_k = 1\}$ to the unit sphere $\mathbb{S}^D := \{\boldsymbol{x} \in \mathbb{R}^K : \sum_{k=1}^K \boldsymbol{x}_k^2 = 1\}$, both of which are D-dimensional manifolds. The geometry of the simplex equipped with the Fisher Information metric is known in closed form [Miyamoto et al., 2024], and we have the following isomorphism between the simplex with the Fisher-Rao metric and the sphere with its canonical metric:

$$\begin{aligned} \varphi &: \Delta^D \to \mathbb{S}^D, \quad \boldsymbol{x} \mapsto \boldsymbol{z} = \sqrt{\boldsymbol{x}}; \\ \varphi^{-1} &: \mathbb{S}^D \to \Delta^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x} = \boldsymbol{z}^2. \end{aligned}$$

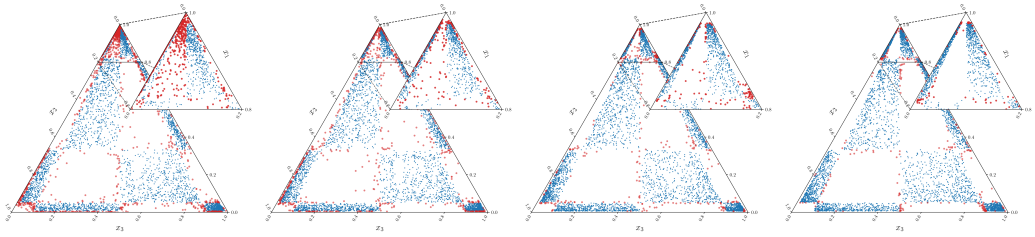
Unlike the Fisher–Rao geometry on the simplex, the spherical geometry remains well-defined on the boundary of the positive orthant. However, the change-of-variables volume term (Appendix B) is singular on the boundary, so likelihood evaluation is only possible on the open simplex; thus, SFM uses a lower bound for the categorical likelihood [Cheng et al., 2024, Eq. (14)].

Even though operating on the sphere is easier than on the simplex, we still need a Riemannian variant of flow matching [Chen and Lipman, 2024], requiring additional geometric machinery (e.g., exponential–log maps) for learning the flow. This adds both theoretical and computational overhead compared to Euclidean flow matching.

3 Method

We propose an alternative extension of conditional flow matching for discrete data. Following Chen and Lipman [2024] and Davis et al. [2024] we also transform the simplex to another space for learning the flow, but instead of mapping to another space that still needs Riemannian geometry we map it to the Euclidean space with simple geometry. A full bijection from the closed simplex to Euclidean space obviously does not exist, but our innovation is to construct such a mapping on the open simplex.

Our solution, coined *Simplex-to-Euclidean Flow Matching* (FM-Δ), has two key components: (i) a bijection from the interior of the simplex to Euclidean space, we provide two alternatives, and (ii) a new mechanism for handling discrete observations on the boundary, called Dirichlet interpolation.



(a) LinearFM (b) SFM (c) FM-Δ (ILR) (d) FM-Δ (SB)

Figure 1: Samples from Checkerboard on the simplex. Red points indicate samples not aligned with the true density. The zoomed area shows the top region $x_1 \geq \frac{4}{5}$, emphasizing the differences.

3.1 Handling discrete data

Discrete observations lie on the boundary of the simplex. We handle them via an interpolation that moves points to the interior while preserving the original category under an argmax operator; we formalize this in Proposition 1. Stark et al. [2024] noticed this property for the argmax under $\boldsymbol{\varepsilon} \sim \mathrm{Unif}(\Delta^D)$. During training we interpolate to move the data to the open simplex and during sampling the argmax operator maps continuous to discrete data (see Algorithms 1 and 2 in Appendix A).

Proposition 1. ***Dirichlet Interpolation:*** *Let $\boldsymbol{c} = \boldsymbol{e}_k$ for some $k \in \{1, .., K\}$. Let $\boldsymbol{x} := \lambda \boldsymbol{c} + (1-\lambda)\boldsymbol{\varepsilon}$ where $\boldsymbol{\varepsilon} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ with $\alpha_i > 0$. If $\lambda > \frac{1}{2}$, then $\arg\max \boldsymbol{x} = \boldsymbol{c}$. For $\lambda = \frac{1}{2}$, this holds almost surely under the distribution of $\boldsymbol{\varepsilon}$.*

3.2 Simplex-to-Euclidean bijections

Stick-breaking transform (SB) The stick-breaking transform [Aitchison, 1982, Carpenter et al., 2017] is for $1 \leq k \leq D$:

$$\begin{aligned} \varphi : \Delta^D \to \mathbb{R}^D, \quad \boldsymbol{x} \mapsto \boldsymbol{z},\ z_k &= \log\left(\frac{x_k}{1-\sum_{i=1}^k x_i}\right) - \log\left(\frac{1}{K-k}\right), \\ \varphi^{-1} : \mathbb{R}^D \to \Delta^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x},\ x_k &= \frac{\exp(y_k)}{\prod_{i=1}^k \left(1+\exp(y_i)\right)},\ y_k = z_k + \log\left(\frac{1}{K-k}\right), \end{aligned} \tag{2}$$

where $x_K = 1 - \sum_{k=1}^D x_k$. The term $\frac{1}{K-k}$ centers the transformation such that the zero vector in $\mathbb{R}^K$ maps to the vector $[\frac{1}{K}, .., \frac{1}{K}] \in \Delta^D$. The change of volumes are in Appendix B.

Isometric logratio transform (ILR) The isometric logratio transform [Egozcue et al., 2003] is an isometry between the simplex with the Aitchison geometry and the Euclidean space [Aitchison, 1982]. It has been recently used for flow models in convex polytopes by a projection into the open simplex [Diederen and Zamboni, 2025]. Given $\boldsymbol{H} \in \mathbb{R}^{D \times K}$ a Helmert matrix, the map is:

$$\begin{aligned} \varphi : \Delta^D \to \mathbb{R}^D, \quad \boldsymbol{x} \mapsto \boldsymbol{z} &= \boldsymbol{H} \log \boldsymbol{x}, \\ \varphi^{-1} : \mathbb{R}^D \to \Delta^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x} &= \mathrm{softmax}(\boldsymbol{H}^\top \boldsymbol{z}). \end{aligned} \tag{3}$$

3.3 Estimating categorical probabilities

We cannot directly evaluate the categorical probability $\Pr(C = k)$ at the vertices, since the model only defines a continuous density on the interior. We do not need the probability for learning (training uses the interpolated points) or sampling (obtained using the argmax-operation) and it is not provided by many alternatives either (e.g. Cheng et al. [2024] only provides an overly loose lower bound, see Appendix D), but nevertheless $\Pr(C = k)$ may be useful for evaluation purposes.

Proposition 1 implies that the true density of the interpolated data $q_{\text{true}}(\boldsymbol{x})$ is a mixture of K Dirichlets, for $\alpha_i > 1$ mass concentrates on the modes $\boldsymbol{\mu}_k := \mathbb{E}[\boldsymbol{x}] = \lambda \boldsymbol{e}_k + (1-\lambda)\frac{1}{K}$. Evaluating at $\boldsymbol{\mu}_k$, we have $\Pr(C{=}k) = \frac{q_{\text{true}}(\boldsymbol{\mu}_k)}{q_\lambda(\boldsymbol{\mu}_k|\boldsymbol{e}_k)}$, which naturally leads to the estimator (see Appendix B.5 for details):

$$\widehat{\Pr}(C = k) = \frac{q_\theta(\boldsymbol{\mu}_k)}{q_\lambda(\boldsymbol{\mu}_k|\boldsymbol{e}_k)}. \tag{4}$$

Table 1: NLL and FID of different discrete models on binarized MNIST. The values above the dotted line are taken from Cheng et al. [2024].

Model	NLL↓	FID↓
DirichletFM	NA	77.35
D3PM	$\leq 0.141 \pm 0.021$	67.36
DFM	0.101 ± 0.017	34.42
DDSM	$\leq 0.100 \pm 0.001$	7.79
LinearFM	NA	5.91
SFM w/ OT	NA	4.62
FM-Δ(SB)	$\approx 0.0341 \pm 0.0006$	4.93
FM-Δ(SB) w/ OT	$\approx 0.0732 \pm 0.0017$	4.51
FM-Δ(ILR)	$\approx 0.0851 \pm 0.0051$	**4.36**
FM-Δ(ILR) w/ OT	$\approx 0.0620 \pm 0.0012$	4.57

Table 2: Values taken from the respective papers.

Model	SP-MSE↓
DDSM	0.0334
D3PM-uniform	0.0375
Bit-Diffusion (one-hot)	0.0395
Bit-Diffusion (bit)	0.0414
Language Model	0.0333
DirichletFM	0.0269
LinearFM	0.0282
SFM	0.0258
FM-Δ(SB)	0.0278
FM-Δ(SB) w/ OT	**0.0214**
FM-Δ(ILR)	0.0259
FM-Δ(ILR) w/ OT	0.0224

4 Experiments

We evaluate the approach on four tasks, learning the density from a collection of samples and evaluating either the match of the learned density or the quality of the generated samples. We compare against DirichletFM [Stark et al., 2024], DDSM [Avdeyev et al., 2023], DFM [Gat et al., 2024], D3PM [Austin et al., 2021] and Bit-Diffusion [Chen et al., 2023]. We also report results for a baseline using Euclidean geometry directly on the simplex [Chen and Lipman, 2024, Stark et al., 2024], denoting it by LinearFM. For both SFM and our method, we report results with minibatch OT (w/OT), and without OT. See Appendix C for details and additional results. We fix $\lambda = \frac{1}{2}$ and $\alpha = 100$ in all cases.

Checkerboard The training data is in Δ^2 generated from the checkerboard distribution, projected to the simplex using the inverse stick-breaking transform. The generated data for FM-Δ is visually better aligned with the true distribution (Fig. 1), with LinearFM and SFM producing numerous poor samples especially near the vertices.

Binarized MNIST The Binarized MNIST dataset assigns each pixel of MNIST to 1 with probability given by its intensity and 0 otherwise [Salakhutdinov and Murray, 2008]; each of the 28×28 pixels takes value in Δ^1. We use the standard train/validation/test split and report both negative log-likelihood (NLL) and the Fréchet inception distance (FID) for the test samples in Table 1. FM-Δ is the best in both metrics, with relatively similar performance for all four variants.

DNA sequence generation We use the human Promoter DNA sequence data from Avdeyev et al. [2023], with 100,000 sequences of 1024 elements with a transcription signal. The task is conditional generation in Δ^3 (conditioned on the signal) with four categories. The training, validation, and test sets are split based on chromosomes: Chromosome 10 is used for validation, Chromosomes 8 and 9 for testing, and the remaining chromosomes for training. Following Avdeyev et al. [2023], we map the generated samples and test samples on a pretrained Sei model [Chen et al., 2022]. The SP-MSE loss is the average Euclidean norm between the two, and Table 2 again shows the proposed method achieves the best accuracy.

5 Discussion

We proposed a simple method that constructs a bridge between the generation of continuous data in the Euclidean space and the generation of discrete data in the simplex, enabling use of broad range of Euclidean generative models for categorical data. We demonstrated the approach in the context of flow matching with highly promising results, and are continuing towards evaluation of alternative generative models, problems of higher dimensionality, and improved estimates for the density itself.

References

John Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 44(2):139–160, 1982.

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, pages 1276–1301. PMLR, 2023.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *International Conference on Machine Learning*, pages 5453–5512. PMLR, 2024.

Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76:1–32, 2017.

Kathleen M Chen, Aaron K Wong, Olga G Troyanskaya, and Jian Zhou. A sequence-based global map of regulatory activity for deciphering human genetics. *Nature genetics*, 54(7):940–949, 2022.

Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *The Twelfth International Conference on Learning Representations*, 2024.

Ting Chen, Ruixiang ZHANG, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*, 2023.

Chaoran Cheng, Jiahan Li, Jian Peng, and Ge Liu. Categorical flow matching on statistical manifolds. *arXiv preprint arXiv:2405.16441*, 2024.

Chaoran Cheng, Jiahan Li, Jiajun Fan, and Ge Liu. α-flow: A unified framework for continuous-state discrete flow matching models. *arXiv preprint arXiv:2504.10283*, 2025.

Oscar Davis, Samuel Kessler, Mircea Petrache, Avishek Joey Bose, et al. Fisher flow matching for generative modeling over discrete data. *arXiv preprint arXiv:2405.14664*, 2024.

Tomek Diederen and Nicola Zamboni. Flows on convex polytopes. *arXiv preprint arXiv:2503.10232*, 2025.

Ian Dunn and David Ryan Koes. Mixed continuous and categorical flow matching for 3d de novo molecule generation. *CoRR*, 2024.

Juan José Egozcue, Vera Pawlowsky-Glahn, Glòria Mateu-Figueras, and Carles Barcelo-Vidal. Isometric logratio transformations for compositional data analysis. *Mathematical geology*, 35(3): 279–300, 2003.

Floor Eijkelboom, Grigory Bartosh, Christian Andersson Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. *Advances in Neural Information Processing Systems*, 37:11735–11764, 2024.

Griffin Floto, Thorsteinn Jonsson, Mihai Nica, Scott Sanner, and Eric Zhengyu Zhu. Diffusion on the probability simplex. *arXiv preprint arXiv:2309.02530*, 2023.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37: 133345–133385, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Henrique K Miyamoto, Fábio CC Meneghetti, Julianna Pinele, and Sueli IR Costa. On closed-form expressions for the fisher–rao distance. *Information Geometry*, 7(2):311–354, 2024.

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin Chiu, and Volodymyr Kuleshov. The diffusion duality. *arXiv preprint arXiv:2506.10892*, 2025.

Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879, 2008.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to dna sequence design. *arXiv preprint arXiv:2402.05841*, 2024.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

Sophia Tang, Yinuo Zhang, Alexander Tong, and Pranam Chatterjee. Gumbel-softmax flow matching with straight-through guidance for controllable biological sequence generation. *arXiv preprint arXiv:2503.17361*, 2025.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024.

A Algorithm

Algorithm 1 gives the training and Algorithm 2 the sampling. In all experiments we set the interpolation weight to $\lambda = \frac{1}{2}$, as this places the interpolated data maximally away from the simplex vertices and recovers the original discrete data with the argmax operator a.s. (Proposition 1). The Dirichlet concentration is chosen as $\alpha = 100$, which reduces the variance of $\varepsilon \sim \mathrm{Dir}(\alpha, .., \alpha)$ around its mode $[\frac{1}{K}, \ldots, \frac{1}{K}]$.

Algorithm 1 Training of Simplex-to-Euclidean Flow Matching (FM-Δ)

Require: Data $\boldsymbol{c} \in \Delta^D$, weight $\lambda \in (0, 1)$, Dirichlet $\alpha > 0$, bijection φ, base p_0 (e.g. $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$), coupling π (indep. or minibatch OT), isDiscrete

for each mini-batch **do**
  for each $\boldsymbol{c}$ in batch **do**
    if isDiscrete **then**
      Sample $\boldsymbol{\varepsilon} \sim \mathrm{Dir}(\alpha, .., \alpha)$
      $\boldsymbol{x} \leftarrow \lambda \boldsymbol{c} + (1 - \lambda)\boldsymbol{\varepsilon}$
    else
      $\boldsymbol{x} \leftarrow \boldsymbol{c}$
  $\boldsymbol{z}_1 \leftarrow \varphi(\boldsymbol{x})$ ▷ To Euclidean space
  Sample $\boldsymbol{z}_0 \sim p_0$; pair $(\boldsymbol{z}_0, \boldsymbol{z}_1)$ via π
  Sample $t \sim \mathrm{Unif}(0, 1)$
  $\boldsymbol{z}_t \leftarrow (1 - t)\boldsymbol{z}_0 + t\boldsymbol{z}_1, \quad \boldsymbol{u}_t \leftarrow \boldsymbol{z}_1 - \boldsymbol{z}_0$
  Update θ by $\min \|\boldsymbol{v}_\theta(\boldsymbol{z}_t, t) - \boldsymbol{u}_t\|^2$

Algorithm 2 Sampling with FM-Δ

Require: Learned $\boldsymbol{v}_\theta$, bijection φ, base p_0, isDiscrete

Sample $\boldsymbol{z}_0 \sim p_0$
Solve ODE: $\frac{d\boldsymbol{z}_t}{dt} = \boldsymbol{v}_\theta(\boldsymbol{z}_t, t), \ t \in [0, 1]$
$\hat{\boldsymbol{x}} \leftarrow \varphi^{-1}(\boldsymbol{z}_1)$
if isDiscrete **then**
  $\hat{\boldsymbol{c}} = \arg\max \hat{\boldsymbol{x}}$ ▷ Discrete sample

B Mathematical Derivations

B.1 Proof of proposition 1

Proof. Let $\boldsymbol{c} \in \Delta^D$ be a vector such that $\boldsymbol{c} = \boldsymbol{e}_k$. The noisy sample is $\boldsymbol{x} := \lambda \boldsymbol{c} + (1 - \lambda)\boldsymbol{\varepsilon}$ with entries

$$x_k = \lambda + (1 - \lambda)\varepsilon_k, \qquad x_j = (1 - \lambda)\varepsilon_j \quad \text{for } j \neq k.$$

We need to show that $x_k > x_j$ for all $j \neq k$ when $\lambda > \frac{1}{2}$. This is equivalent to

$$\lambda + (1 - \lambda)\varepsilon_k > (1 - \lambda)\varepsilon_j, \iff \lambda > (1 - \lambda)(\varepsilon_j - \varepsilon_k) \iff \frac{\lambda}{1 - \lambda} > \varepsilon_j - \varepsilon_k.$$

Since ε lies in the simplex, $\varepsilon_j - \varepsilon_k \leq 1$, and for $\lambda > \frac{1}{2}$, we have $\frac{\lambda}{1-\lambda} > 1 \geq \varepsilon_j - \varepsilon_k$. Consequently, the inequality holds, implying

$$x_k > x_j \quad \forall j \neq i, \text{ and } \arg\max_j x_j = k.$$

For the boundary case $\lambda = \frac{1}{2}$, the condition becomes

$$\frac{\lambda}{1 - \lambda} = 1, \quad \text{then} \quad 1 \geq \varepsilon_j - \varepsilon_k \quad (\text{equivalently } x_k \geq x_j).$$

Note the equality $x_k = x_j$ occurs iff $\varepsilon_j - \varepsilon_k = 1$, i.e., $\varepsilon_j = 1$ and $\varepsilon_k = 0$. Under any Dirichlet distribution with positive concentration parameters, this boundary event has probability zero. Therefore, when $\lambda = \frac{1}{2}$ we have $x_k > x_j$ for all $j \neq k$ almost surely, and thus $\arg\max_j x_j = k$ almost surely. □

B.2 Stick-breaking transform

Carpenter et al. [2017] give an alternative formulation of the SB inverse transform. We prove its equivalence to the SB inverse transform, and use this equivalence to derive a simpler expression of the Jacobian determinant.

The inverse unit-simplex transform (US) is defined for $1 \leq k \leq D$

$$\varphi^{-1} : \mathbb{R}^D \to \Delta^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x}, \quad x_k = \left(1 - \sum_{i=1}^{k-1} x_i\right) \sigma(y_k),\ y_k = z_k + \log \frac{1}{K-k}, \tag{5}$$

where $\sigma(\cdot)$ denotes the sigmoid function and the last entry is $x_K = 1 - \sum_{i=1}^{D} x_i$. Proposition 2 shows the equivalence between the stick-breaking and unit-simplex inverse transforms.

Proposition 2. *The SB inverse transform and the US inverse transform are equal.*

Proof. Proof by induction. Recall the SB inverse transform is $x_k = \prod_{i=1}^{k-1}(1 - \sigma(y_i))\sigma(y_k)$.

For the base case $k = 2$, we have $x_1 = \sigma(y_1)$ and $x_2 = (1 - \sigma(y_1))\sigma(y_2)$ in both cases.

Induction step, we assume SB and US coincide for k.

Let us prove the equality for $k + 1$,

$$\begin{aligned} x_{k+1} &= \left(1 - \sum_{i=1}^{k} x_i\right) \sigma(y_{k+1}) = \left(1 - \sum_{i=1}^{k-1} x_i - x_k\right) \sigma(y_{k+1}) \overset{1-\sum_{i=1}^{k-1} x_i = \frac{x_k}{\sigma(y_k)}}{=} \left(\frac{x_k}{\sigma(y_k)} - x_k\right) \sigma(y_{k+1}) \\ &= \left(\prod_{i=1}^{k-1}(1 - \sigma(y_i)) - \prod_{i=1}^{k-1}(1 - \sigma(y_i))\sigma(y_k)\right) \sigma(y_{k+1}) = \prod_{i=1}^{k}(1 - \sigma(y_i))\sigma(y_{k+1}). \end{aligned}$$

□

Computation of the determinant Take $y_k := z_k + \log\left(\frac{1}{K-k}\right)$, then $x_k = \frac{e^{y_k}}{\prod_{i=1}^{k}(1+e^{y_i})}$

$$\frac{\partial x_k}{\partial z_k} = \frac{e^{y_k}}{\prod_{i=1}^{k}(1 + e^{y_i})} \left(1 - \frac{e^{y_k}}{1 + e^{y_k}}\right) = \frac{e^{y_k}}{\prod_{i=1}^{k}(1 + e^{y_i})} \left(\frac{1}{1 + e^{y_k}}\right) = x_k \left(\frac{1}{1 + e^{y_k}}\right)$$

Since the Jacobian is lower triangular, the determinant is the product of the diagonal terms

$$\det \boldsymbol{J}_\varphi = \prod_{k=1}^{D} x_k \left(\frac{1}{1 + e^{y_k}}\right).$$

As a side result of Proposition 2 we have the equality $1 - \sum_{i=1}^{k} x_i = \prod_{i=1}^{k}(1 - \sigma(y_i))$, thus $x_K = \prod_{i=1}^{D}(1 - \sigma(y_i))$, and we obtain

$$\det \boldsymbol{J}_\varphi = \prod_{k=1}^{K} x_k.$$

B.3 Isometric logratio transform

We state the matrix determinant Lemma 1 which is useful throughout the derivations.

Lemma 1. *Matrix determinant. Let $\boldsymbol{A} \in \mathbb{R}^{D \times D}$ be a full rank square matrix and $u \in \mathbb{R}^D$ a vector, then*

$$\det\left(\boldsymbol{A} + uu^\top\right) = \left(1 + u^\top \boldsymbol{A}^{-1} u\right) \det(\boldsymbol{A}).$$

The ILR transform is $\boldsymbol{z} = \boldsymbol{H} \log \boldsymbol{x}$, we fix the last entry as $x_K = 1 - \sum_{i=1}^{D} x_i$. The entries of the Jacobian matrix are:

$$\frac{\partial z_i}{\partial x_j} = \frac{h_{ij}}{x_j} - \frac{h_{i,K}}{x_K}.$$

The Jacobian can be written in matrix form

$$\boldsymbol{J}_\varphi = \boldsymbol{H}_{1:D,1:D} \operatorname{diag}\left(\frac{1}{x_1}, \ldots, \frac{1}{x_D}\right) - \frac{1}{x_K} \boldsymbol{h}_K \mathbf{1}^\top,$$

where $\boldsymbol{h}_K$ is the last column of $\boldsymbol{H}$ and $\mathbf{1}$ is the vector of ones. A property of the Helmert matrix is that the columns of $\boldsymbol{H}$ sum to zero, hence $\boldsymbol{h}_K = -\boldsymbol{H}_{1:D,1:D}\mathbf{1}$ and

$$\boldsymbol{J}_\varphi = \boldsymbol{H}_{1:D,1:D}\left(\operatorname{diag}\left(\frac{1}{x_1},\ldots,\frac{1}{x_D}\right) + \frac{1}{x_K}\mathbf{1}\mathbf{1}^\top\right).$$

The determinant can be computed with the help of Lemma 1,

$$\begin{aligned}
\det(\boldsymbol{J}_\varphi) &= \det(\boldsymbol{H}_{1:D,1:D})\prod_{i=1}^{D}\tfrac{1}{x_i}\left(1 + \frac{1}{x_K}\sum_{i=1}^{D} x_i\right) \\
&= \det(\boldsymbol{H}_{1:D,1:D})\prod_{i=1}^{D}\frac{1}{x_i}\left(1 + \frac{1}{x_K}(1 - x_K)\right) \\
&= \det(\boldsymbol{H}_{1:D,1:D})\prod_{i=1}^{K}\frac{1}{x_i}.
\end{aligned}$$

The determinant of the reduced Helmert matrix is $\det(\boldsymbol{H}_{1:D,1:D}) = \frac{1}{\sqrt{K}}$.

B.4 Simplex to sphere transform

Let $\boldsymbol{z} \in \mathbb{S}_+^D$ such that the sphere bijection is $\boldsymbol{z} = \varphi(\boldsymbol{x}) = \sqrt{\boldsymbol{x}}$ for $\boldsymbol{x} \in \Delta^D$. A direct computation gives the Jacobian $\boldsymbol{J}_\varphi \in \mathbb{R}^{K\times D}$

$$\boldsymbol{J}_\varphi = \frac{dz}{dx_{1:D}} = \begin{bmatrix} \frac{1}{2\sqrt{x_1}} & 0 & \cdots & 0 \\ 0 & \frac{1}{2\sqrt{x_2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{2\sqrt{x_D}} \\ -\frac{1}{2\sqrt{x_K}} & -\frac{1}{2\sqrt{x_K}} & \cdots & -\frac{1}{2\sqrt{x_K}} \end{bmatrix}_{K\times D}.$$

The pull-back metric is $\boldsymbol{G}_\varphi = \boldsymbol{J}_\varphi^\top \boldsymbol{J}_\varphi \in \mathbb{R}^{D\times D}$;

$$\boldsymbol{G}_\varphi = \frac{1}{4}\left(\operatorname{diag}\left(\frac{1}{x_1},\ldots,\frac{1}{x_D}\right) + \frac{1}{x_K}\mathbf{1}_D\mathbf{1}_D^\top\right),$$

This is a rank-1 update of a diagonal matrix, set $\boldsymbol{A} = \operatorname{diag}\left(\frac{1}{x_1},\ldots,\frac{1}{x_D}\right)$, and $u = \sqrt{\frac{1}{x_K}}\cdot\mathbf{1}_D$, we obtain

$$1 + u^\top \boldsymbol{A}^{-1}u = \frac{1}{x_K}\sum_{i=1}^{D} x_i = 1 + \frac{1 - x_K}{x_K} = \frac{1}{x_K}.$$

Due to Lemma 1 the determinant of $\boldsymbol{G}_\varphi$ and the volume element are:

$$\det(\boldsymbol{G}_\varphi) = \frac{1}{4^D}\prod_{i=1}^{K}\frac{1}{x_i}, \quad \sqrt{\det(\boldsymbol{G}_\varphi)} = \frac{1}{2^D}\prod_{i=1}^{K}\frac{1}{\sqrt{x_i}}.$$

B.5 Categorical probabilities estimation

We have constructed the estimator of the categorical probabilities

$$\widehat{\Pr}(C{=}k) = \tfrac{q_\theta(\boldsymbol{\mu}^{(k)})}{q_\lambda(\boldsymbol{\mu}^{(k)}|\boldsymbol{e}_k)}.$$

For its computation we need two components: the log densities of our model in the simplex $q_\theta(\boldsymbol{x})$ for $\boldsymbol{x} \in \Delta^D$ and the true densities for each mixture component $q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k)$.

Computing the distribution of the model in the simplex Recall that $\boldsymbol{x} \in \Delta^D$ and $\boldsymbol{z} \in \mathbb{R}^D$ and $\boldsymbol{v}_t^\theta$ is the vector field of the flow model. The density in the Euclidean space is given by the instantaneous change of variable formula:

$$\log p_1(\boldsymbol{z}_1) = \log p_0(\boldsymbol{z}_0) - \int_0^1 \operatorname{div}(\boldsymbol{v}_s^\theta)(\boldsymbol{z}_s)\,\mathrm{d}s. \tag{6}$$

Change of variables for the transformation $\boldsymbol{x} = \varphi^{-1}(\boldsymbol{z}_1)$ gives the density on Δ^D:

$$\log q_\theta(\boldsymbol{x}) = \log p_1(\varphi(\boldsymbol{x})) + \log \left| \frac{\partial \varphi(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|. \tag{7}$$

Combining Equations (6) and (7) we obtain the density over the simplex:

$$\log q_\theta(\boldsymbol{x}) = \log p_0(\boldsymbol{z}_0) - \int_0^1 \operatorname{div}(\boldsymbol{v}_s^\theta)(\boldsymbol{z}_s)\,\mathrm{d}s + \log \left| \frac{\partial \boldsymbol{z}_1}{\partial \boldsymbol{x}} \right|.$$

If $\boldsymbol{z}_0$ is distributed as a standard Gaussian in $\mathbb{R}^D$ then $p_0(\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_0 \mid \boldsymbol{0}, \boldsymbol{I})$. If the base distribution is the uniform distribution, $\boldsymbol{x}_0 \sim \mathrm{Unif}(\Delta^D)$, on the simplex, then p_0 is computed with an additional change of variables

$$\log p_0(\boldsymbol{z}_0) = \log p_0(\varphi^{-1}(\boldsymbol{z}_0)) + \log \left| \frac{\partial \varphi^{-1}(\boldsymbol{z}_0)}{\partial \boldsymbol{z}_0} \right|.$$

Computing the distribution of the mixture components The Dirichlet interpolation moves discrete data from the vertices to a Dirichlet mixture. Each mixture component conditioned on $\boldsymbol{e}_k$ has distribution $q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k)$. Let us compute this distribution. Let $\boldsymbol{\varepsilon} \sim \mathrm{Dir}(\alpha)$ with density $p_{\boldsymbol{\varepsilon}}(\boldsymbol{v})$ and $\alpha > 0$. Define the affine map $f : \boldsymbol{\varepsilon} \mapsto \boldsymbol{x} = \lambda \boldsymbol{e}_k + (1-\lambda)\boldsymbol{\varepsilon}$. Its inverse is

$$f^{-1}(\boldsymbol{x}) = \frac{\boldsymbol{x} - \lambda \boldsymbol{e}_k}{1-\lambda}.$$

The mapping acts as a scaling by factor $(1-\lambda)$ in D dimensions, hence the Jacobian absolute determinant is

$$\left| \det \boldsymbol{J}_f^{-1} \right| = \frac{1}{(1-\lambda)^D}.$$

By change of variables,

$$q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k) = p_{\boldsymbol{\varepsilon}}\big(f^{-1}(\boldsymbol{x})\big)\,\big|\det \boldsymbol{J}_f^{-1}\big|.$$

Multiplying by the Jacobian factor $\frac{1}{(1-\lambda)^D}$ yields the final density (supported on the truncated simplex $\{\boldsymbol{x} \in \Delta^D : x_k \geq \lambda\}$)

$$q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k) = \frac{1}{(1-\lambda)^D} \mathrm{Dir}(f^{-1}(\boldsymbol{x}); \alpha).$$

C Experimental details

Checkerboards and Corners For the evaluation of checkboard we draw 5000 samples with the Dopri5 solver. For corners samples are generated with the Euler solver for 200 steps. We draw $N = 10^5$ generated samples $\{\hat{x}_i\}_{i=1}^N$, and estimate $\hat{\boldsymbol{p}} = \sum_{i=1}^N \hat{x}_i$ (normalized to sum up to one). The true distribution of $\boldsymbol{p} \in \Delta^D$ is $p_1 = \frac{1}{2}$ and $[p_2, .., p_K] \sim \frac{1}{2}\mathrm{Unif}(\Delta^{D-1})$. We train a standard fully connected network with 4 hidden layers of 512 units. The input dimension of the network is $K + 64$, where 64-dim the sinusoidal embedding is used on t.

Binarized MNIST Samples are generated with the Euler solver for 300 steps for all our methods, non cherry-picked examples of generated images are shown in Fig. 2. Our approximation of the NLL is obtained by evaluation Eq. (4) on both the test image x and its flipped version $1 - x$ standardizing so the sum is 1. The FID statistics (mean, covariance) are computed with the InceptionV3 model [Szegedy et al., 2016] over the full training data. The FID is evaluated with the statistics of 1000 generated samples for each model. Following Cheng et al. [2024] we used the CNN network from Song and Ermon [2020]. The hyperparameters are fixed for all models to the same values. Each model is trained for approximately 500 epochs on a single NVIDIA Volta V100 GPU.

Method	Bijection	OT	Batch	Opt	WD	B1	Step	SP-MSE(val)	SP-MSE(test)
FM-Δ	SB	False	64	adam	0	0.85	40000	0.0251	0.0278
FM-Δ	SB	False	64	adam	0	0.95	40000	0.0321	0.0341
FM-Δ	SB	False	128	adam	0	0.85	30000	0.0327	0.0353
FM-Δ	SB	False	128	adam	0	0.95	30000	0.0406	0.0443
FM-Δ	SB	False	128	adam	10^{-5}	0.85	120000	0.0435	0.0447
FM-Δ	SB	False	64	adam	10^{-5}	0.85	200000	0.0506	0.0512
FM-Δ	SB	False	128	adam	10^{-5}	0.95	120000	0.0509	0.0514
FM-Δ	SB	False	64	adam	10^{-5}	0.95	200000	0.0549	0.0554
FM-Δ	SB	True	64	adam	0	0.85	40000	0.0213	0.0214
FM-Δ	SB	True	128	adam	0	0.85	30000	0.0314	0.0325
FM-Δ	SB	True	128	adam	10^{-5}	0.95	120000	0.0363	0.038
FM-Δ	SB	True	128	adam	0	0.95	30000	0.0387	0.0409
FM-Δ	SB	True	128	adam	10^{-5}	0.85	120000	0.0387	0.0392
FM-Δ	SB	True	64	adam	10^{-5}	0.95	190000	0.0405	0.0435
FM-Δ	SB	True	64	adam	0	0.95	40000	0.0591	0.0569
FM-Δ	SB	True	64	adam	10^{-5}	0.85	190000	0.0613	0.0642

Table 3: Values of SP-MSE on validation and test data for tested hyperparameters SB.

Method	Bijection	OT	Batch	Opt	WD	B1	Step	SP-MSE(val)	SP-MSE(test)
FM-Δ	ILR	False	64	adam	0	0.85	40000	0.0252	0.0259
FM-Δ	ILR	False	64	adam	0	0.95	40000	0.0321	0.0335
FM-Δ	ILR	False	128	adam	0	0.85	30000	0.0328	0.0346
FM-Δ	ILR	False	128	adam	0	0.95	30000	0.0406	0.0443
FM-Δ	ILR	False	128	adam	10^{-5}	0.85	120000	0.0436	0.0449
FM-Δ	ILR	False	64	adam	10^{-5}	0.85	200000	0.0508	0.0526
FM-Δ	ILR	False	128	adam	10^{-5}	0.95	120000	0.0511	0.0511
FM-Δ	ILR	False	64	adam	10^{-5}	0.95	200000	0.0548	0.0553
FM-Δ	ILR	True	64	adam	0	0.85	40000	0.0213	0.0224
FM-Δ	ILR	True	128	adam	0	0.85	30000	0.0314	0.0317
FM-Δ	ILR	True	128	adam	10^{-5}	0.95	120000	0.0362	0.0384
FM-Δ	ILR	True	128	adam	0	0.95	30000	0.0387	0.0419
FM-Δ	ILR	True	128	adam	10^{-5}	0.85	120000	0.0387	0.039
FM-Δ	ILR	True	64	adam	10^{-5}	0.95	190000	0.0405	0.042
FM-Δ	ILR	True	64	adam	0	0.95	40000	0.0588	0.0581
FM-Δ	ILR	True	64	adam	10^{-5}	0.85	190000	0.0612	0.0626

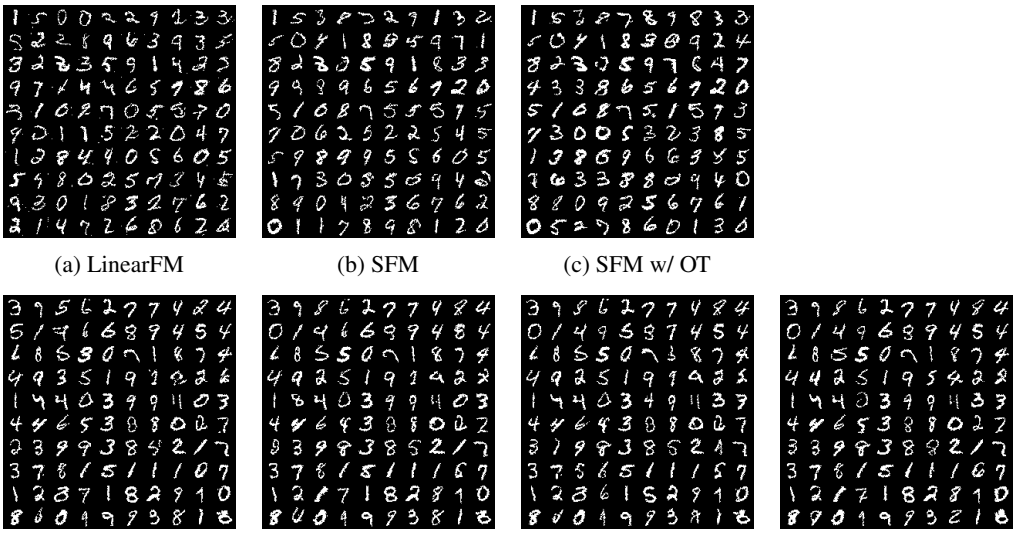
Table 4: Values of SP-MSE on validation and test data for tested hyperparameters ILR.

Promoter Design Samples are generated with the Euler solver for 300 steps for all our methods. The chromatin mark for the Sei model is H3K4me3 [Avdeyev et al., 2023], the SP-MSE is computed for all samples in the test set and the same number of generated samples. For each training run we fix the model weights as the best value in terms of validation loss (Eq. (1)). Then we select the best model with respect to the SP-MSE on the validation dataset (Table 3 and 4). WD means weight decay, B1 is the first moment of the Adam optimizer, and the Step is iteration with lowest validation loss. The network considered is the same as in Avdeyev et al. [2023]. Training is conducted for 200K steps on a single NVIDIA Ampere A100 GPU (40GB) over the hyperparameter grid shown in Tables 3 and 4.

D Additional Experiments

D.1 Corners

For the number of categories $K = 2^1, \ldots, 2^9$, we train the methods with 10^6 discrete samples distributed $\mathrm{Cat}(\boldsymbol{p})$ for fixed $\boldsymbol{p} \in \Delta^D$. The purpose is to study the scalability of the methods as the



(a) LinearFM (b) SFM (c) SFM w/ OT

(d) FM-Δ(ILR) (e) FM-Δ(ILR) w/ OT (f) FM-Δ(SB) (g) FM-Δ(SB) w/ OT

Figure 2: Samples from BMNIST from the different methods. Linear FM draws samples of visually lower quality than the rest of the methods.

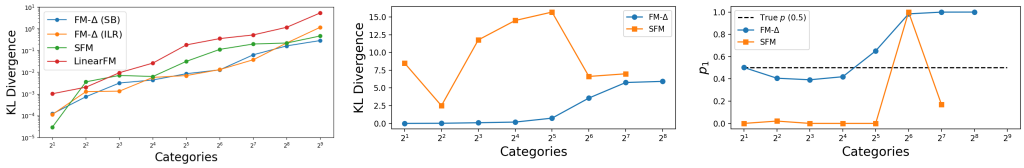


Figure 3: Left: KL Divergence between true and estimated densities, as a function of the number of categories. Middle: KL divergence between true probabilities and approximations. Right: Value of p_1 for our approximation and for the lower bound.

number of categories increases. We evaluate $\hat{\boldsymbol{p}}$ using empirical KL divergence estimated from the average of 10^5 drawn samples. Fig. 3 left shows our method outperforms SFM and LinearFM for all $K \geq 2^2$.

D.2 Estimation of the categorical probabilities

We conduct a simple experiment where we test the accuracy of the approximation of the categorical log density Eq. (4), using the same setup as the Corners experiment. We compare our estimator $\widehat{\Pr}(C = k)$ to the lower bound proposed by Cheng et al. [2024]. Figure 3 shows that our estimator closely matches the true probabilities up to 2^5 categories in terms of KL divergence, whereas the lower bound used by SFM deviates significantly for all D. Similarly, our estimator of $p_1 = 0.5$ remains accurate up to 2^5 categories, while the lower bound remains loose across all dimensions and results in numerical errors for high number of categories.