Simplex-to-Euclidean Bijections for Categorical Flow Matching

Bernardo Williams[1] **Victor M. Yeom-Song**[2,3] **Marcelo Hartmann**[1] **Arto Klami**[1]
[1]Department of Computer Science, University of Helsinki
[2]Department of Computer Science, Aalto University
[3]ELLIS Institute Finland

Abstract

We propose a method for learning and sampling from probability distributions supported on the simplex. Our approach maps the open simplex to Euclidean space via smooth bijections, leveraging the Aitchison geometry to define the mappings, and supports modeling categorical data by a Dirichlet interpolation that dequantizes discrete observations into continuous ones. This enables density modeling in Euclidean space through the bijection while still allowing exact recovery of the original discrete distribution. Compared to previous methods that operate on the simplex using Riemannian geometry or custom noise processes, our approach works in Euclidean space while respecting the Aitchison geometry, and achieves competitive performance on both synthetic and real-world data sets.

1 INTRODUCTION

We study the problem of learning and generating samples from probability distributions supported on the unit simplex, a setting that naturally arises when working with compositional data (vectors of non-negative components that sum 1o one). Learning distributions on the simplex provides a natural framework for many real-world applications. For example, in computational biology, sequences can be encoded as categorical variables e.g. for conditional generation of DNA sequences (Avdeyev et al., 2023) or proteins (Campbell et al., 2024), and compositional data naturally arises in geology, chemistry, design and economics (Aitchison, 1982; Chereddy and Femiani, 2025; Diederen and Zamboni, 2025).

There are two main ways of modeling categorical data. The first are discrete-state models that manipulate directly the categorical states, often by modeling transition dynamics. This family includes discrete flow and diffusion models (Austin et al., 2021; Campbell et al., 2024; Gat et al., 2024; Sahoo et al., 2025) as well as masked diffusion models (Sahoo et al., 2024). The other category adapts continuous-state models, such as standard diffusion and flow models, to work with categorical observations by continuous relaxations. We work within the latter category, to ease the use of established continuous models with well-understood learning dynamics and good implementations (Karras et al., 2022; Lipman et al., 2024) for discrete data.

The continuous relaxation models can be further divided into two broad categories: Some operate directly on the simplex, while others avoid the simplex entirely by working in the ambient space $\mathbb{R}^K$, gradually moving continuous samples toward the vertices to recover discreteness (Chen et al., 2023; Eijkelboom et al., 2024). Within the simplex-based models, two challenges arise: (i) handling the boundary, where discrete data lie, and (ii) accounting for the simplex's non-Euclidean geometry. Existing approaches tackle these in different ways: Some define distributions directly on the simplex, either via custom marginals for diffusion processes (Avdeyev et al., 2023; Floto et al., 2023; Tae et al., 2025) or vector fields (Stark et al., 2024; Dunn and Koes, 2024; Tang et al., 2025), or by mapping the simplex to the sphere through bijections (Davis et al., 2024; Cheng et al., 2024, 2025). We build on the methods that operate on the simplex, focusing on two related aspects: How to account for the geometry of the simplex and the discrete data at its boundary in a principled way.

Concretely, we propose a new approach for modeling categorical data with continuous generative models, aiming for conceptual and implementation simplicity to best leverage the existing continuous tools. We map

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

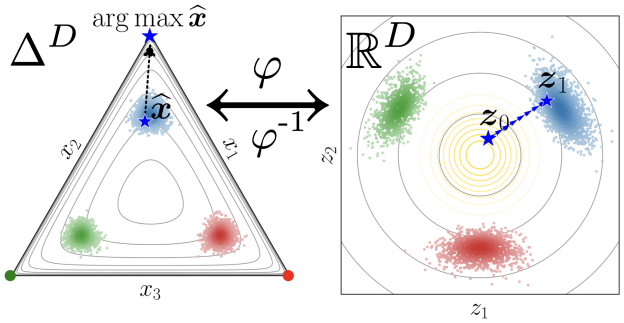


Figure 1: We stochastically interpolate categorical observations (color) to distributions on the interior of a simplex (left). The resulting Dirichlet mixture is transformed to Euclidean space (right) with a bijection φ, enabling use of standard continuous generative models, like conditional flow matching. Discrete samples are obtained by composition of the inverse transformation φ^{-1} and arg max operation.

the *interior* of the simplex to Euclidean space using a smooth bijection drawn from compositional data analysis and the Aitchison geometry, a geometry induced by the logratio of the components (Aitchison, 1982). We can then train a standard continuous generative model in Euclidean space. For compositional data this is enough to account for the simplex geometry, but for categorical observations that lie at the boundary of the simplex we need additional tools. They are not covered by our bijection, but for training the model we can map them into interior points via a Dirichlet interpolation scheme, a stochastic generalization of a recent dequantization approach by Chereddy and Femiani (2025). We do this so that we can retrieve the original category by a simple arg max-operation, which means we can also easily convert continuous samples into discrete ones. Figure 1 illustrates the method.

We demonstrate the approach using Flow Matching (Lipman et al., 2023; Albergo and Vanden-Eijnden, 2023) as the continuous generative model, and propose two alternative bijections: the stick-breaking transform (Aitchison, 1982) and the isometric logratio transform (Egozcue et al., 2003). Both define a geometry on the simplex, and explicitly connect this geometry to Euclidean space. Moreover, they are computationally lightweight and easy to implement.

We establish several theoretical results and empirically validate the method on standard benchmarks, demonstrating highly competitive performance within the continuous relaxations and outperforming discrete-state models in low-dimensional problems.

2 BACKGROUND

For K classes, the simplex is a space of $K-1$ dimensions. For notational simplicity, we denote this by $D = K-1$. Discrete data is denoted $\boldsymbol{c} \in \{\boldsymbol{e}_1, \dots, \boldsymbol{e}_K\}$ and generated samples with a hat, e.g. $\hat{\boldsymbol{c}}$, to distinguish them from true data samples. The (closed) simplex is defined $\Delta^D := \{\boldsymbol{x} \in \mathbb{R}^K : x_i \geq 0,\ \sum_{i=1}^K x_i = 1\}$, and the open simplex as $\mathring{\Delta}^D := \{\boldsymbol{x} \in \Delta^D : x_i > 0\ \forall i\}$. The boundary $\partial\Delta^D := \Delta^D \setminus \mathring{\Delta}^D$ consists of all borders where at least one coordinate is zero. We will also make reference to the unit sphere, defined as $\mathbb{S}^D := \{\boldsymbol{x} \in \mathbb{R}^K : \sum_{k=1}^K x_k^2 = 1\}$, and the positive orthant of the sphere $\mathbb{S}_+^D := \{\boldsymbol{x} \in \mathbb{S}^D : x_i \geq 0\ \forall i\}$. We use the notation $\arg\max \boldsymbol{x} := \boldsymbol{e}_j$ with $j = \arg\max_i x_i$ to denote the one-hot vector at the maximizing index.

Equipping $\mathring{\Delta}^D$ with the Fisher information metric yields a D-dimensional Riemannian manifold. Intuitively, a Riemannian metric specifies an inner product on the tangent space at each point, which in turn induces geodesics (shortest paths) and distances (see e.g. Lee (2018)). For the Fisher metric on the simplex, both geodesics and geodesic distances admit closed-form expressions. Similarly, the unit sphere $\mathbb{S}^D$ is a D-dimensional Riemannian manifold endowed with the canonical metric induced by its embedding in $\mathbb{R}^K$.

2.1 Conditional Flow Matching

We use Conditional Flow Matching (CFM) (Lipman et al., 2023; Albergo and Vanden-Eijnden, 2023) as the example (Euclidean) generative model, and hence briefly review it here.

CFM is a continuous-time normalizing flow for 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 $\boldsymbol{v}_t^\theta(\boldsymbol{x}_t)$ 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, which leads to the CFM objective

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{t\sim\text{Unif}(0,1) \\ \boldsymbol{x}_0,\boldsymbol{x}_1\sim\pi(\boldsymbol{x}_0,\boldsymbol{x}_1)}} \left\| \boldsymbol{v}_t^\theta(\boldsymbol{x}_t) - \boldsymbol{u}_t(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{x}_1) \right\|_2^2. \tag{1}$$

There are multiple interpolation schemes Lipman et al. (2024). A simple and commonly used choice is the linear interpolation, $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$, where $\boldsymbol{x}_0 \sim p_0$ and $\boldsymbol{x}_1 \sim p_1$, resulting in target conditional velocity $\boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0,\boldsymbol{x}_1) = \dot{\boldsymbol{x}}_t = \boldsymbol{x}_1 - \boldsymbol{x}_0$.

Two common choices for the coupling π of p_0 and p_1 are the following. The independent coupling draws

$(\boldsymbol{x}_0, \boldsymbol{x}_1)$ independently from $\pi(\boldsymbol{x}_0, \boldsymbol{x}_1) = p_0(\boldsymbol{x}_0)p_1(\boldsymbol{x}_1)$ and the optimal transport coupling (OT) constructs an approximation of the optimal transport plan on each minibatch of samples of p_0 and p_1 (Tong et al., 2024). Concretely, given a batch of samples from each distribution, a cost matrix is computed using Euclidean distances, and a pairing of the samples (transport plan) is obtained that approximately minimizes the total cost while preserving the empirical marginals. The OT coupling can improve training stability and accelerate inference.

2.2 Riemannian Flow Matching on the Simplex

To illustrate how previous methods handle the geometry of the simplex, we briefly outline Statistical Flow Matching (SFM) (Cheng et al., 2024; Davis et al., 2024). The basic idea is to map the simplex to a space that is "easier" in some computational sense, and learn the flow there, eventually mapping the continuous samples from the model back to discrete observations with another transformation.

SFM transforms the simplex Δ^D to the positive orthant of the unit sphere $\mathbb{S}^D_+$, another D-dimensional manifold. 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:

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

The transformation gives the change of volume

$$\det \boldsymbol{G}_\varphi = \frac{1}{2^D} \prod_{i=1}^{K} \frac{1}{\sqrt{x_i}}, \tag{2}$$

where $\boldsymbol{G}_\varphi = \boldsymbol{J}_\varphi^\top \boldsymbol{J}_\varphi$ is the pullback metric of the transformation. Unlike the Fisher–Rao geometry on the simplex, the spherical geometry (exponential–log maps) is well-defined on the boundary of the positive orthant. However, the change-of-variables volume term (Eq. 2) becomes singular whenever at least one coordinate is zero, so likelihood evaluation is only possible on the open simplex. To address this, SFM uses a lower bound for the categorical likelihood (Cheng et al., 2024, Eq. (14)). Even though working on the sphere avoids some boundary issues, training still requires a Riemannian variant of Flow Matching (Chen and Lipman, 2024) with associated geometric machinery. The training objective is

$$\mathbb{E}_{\substack{t\sim\mathrm{Unif}(0,1) \\ \boldsymbol{x}_0,\boldsymbol{x}_1\sim\pi(\boldsymbol{x}_0,\boldsymbol{x}_1)}} \left\| \boldsymbol{v}_t^\theta(\boldsymbol{x}_t) - \mathrm{Log}_{\boldsymbol{x}_t}(\boldsymbol{x}_1)/(1-t) \right\|_g^2,$$

where $\|\cdot\|_g$ is the Riemannian norm, and $\boldsymbol{x}_t = \mathrm{Exp}_{\boldsymbol{x}_0}(t\boldsymbol{v})$ and $\boldsymbol{v} = \mathrm{Log}_{\boldsymbol{x}_0}(\boldsymbol{x}_1)$ are the exponential and logarithmic maps.

3 METHOD

We build on the same idea as SFM, of transforming the simplex to a space that is easier to work on. Instead of mapping it to another space that still requires Riemannian machinery, we consider transformations that take us to Euclidean space, where all computations are easy and we can directly leverage standard continuous models. A full bijection from the closed simplex to $\mathbb{R}^D$ obviously does not exist, but our innovation is to construct such a mapping on $\mathring{\Delta}^D$, the interior of the simplex. The method operates entirely within $\mathring{\Delta}^D$, with information-preserving mappings between the $\mathring{\Delta}^D$ and the discrete categories.

The method, coined *Simplex-to-Euclidean Flow Matching* (FM-$\mathring{\Delta}$), has two main components: (i) A bijection from the open simplex to Euclidean space, for which we provide two concrete alternatives, and (ii) Dirichlet interpolation for handling discrete observations on the boundary, both for lifting them into the open simplex and eventually for mapping generated samples back to discrete categories.

3.1 Simplex-to-Euclidean bijections

Compositional Data Analysis Aitchison and Shen (1980); Aitchison (1981, 1982) noted that for compositional data (vectors in the open simplex), the logratio between the components provides a principled method for constructing smooth bijections from the open simplex to Euclidean space. Two classical examples, which will serve as building blocks of the transforms we later propose, are the additive logratio (ALR), for $1 \le k \le D$

$$\mathrm{alr}(\boldsymbol{x})_k := \log \tfrac{x_k}{x_K}, \ \mathrm{alr}^{-1}(\boldsymbol{z}) = \mathrm{softmax}([\boldsymbol{z}, 0]), \tag{3}$$

and the multiplicative logratio (MLR)

$$\mathrm{mlr}(\boldsymbol{x})_k := \log \tfrac{x_k}{1-\sum_{i=1}^k x_i}, \ \mathrm{mlr}^{-1}(\boldsymbol{z})_k = \frac{e^{z_k}}{\prod_{i=1}^k \left(1+e^{z_i}\right)}.$$

The last entry depends on the others, with $x_K = 1 - \sum_{i=1}^D x_i$. Note that both transformations depend on the ordering of the components of $\boldsymbol{x}$, best seen by the explicit reference to an arbitrary (last) component x_K in $\mathrm{alr}(\boldsymbol{x})$. Permuting the inputs produces a different map and consequently also a different geometry.

These transformations induce a geometry on $\mathring{\Delta}^D$ that differs from the Fisher–Rao metric. The Aitchison geometry equips the open simplex with the inner product

(Aitchison, 1983)

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_A := \frac{1}{2K} \sum_{i,j=1}^{K} \log \frac{x_i}{x_j} \log \frac{y_i}{y_j},$$

which captures the relative structure of compositions. Addition in this geometry is defined through a perturbation, $\boldsymbol{x} \oplus \boldsymbol{y} = C(x_1 y_1, \ldots, x_K y_K)$, where $C(\boldsymbol{z}) = \boldsymbol{z}/\sum_k z_k$ ensures the result remains in the simplex. Distances and inner products are invariant under such perturbation, highlighting that the information is carried by the ratios. These properties make the Aitchison geometry particularly well-suited for data in $\mathring{\Delta}^D$.

The naive ALR and MLR mappings could in principle be used as such (see the Supplement C.4), and recently Chereddy and Femiani (2025) indeed used ALR for modeling discrete data (for CAD generation) with diffusion models, without accounting for the order invariance in any way. We will next introduce two concrete mappings; one is invariant to the order, resolving the fundamental limitation, whereas the other improves on MLR by aligning it to the center point on each space.

Isometric logratio transform (ILR) We consider the isometric logratio transform (Egozcue et al., 2003)

$$\begin{aligned} \varphi &: \mathring{\Delta}^D \to \mathbb{R}^D, \quad \boldsymbol{x} \mapsto \boldsymbol{z} = \boldsymbol{H} \log \boldsymbol{x}, \\ \varphi^{-1} &: \mathbb{R}^D \to \mathring{\Delta}^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x} = \operatorname{softmax}(\boldsymbol{H}^\top \boldsymbol{z}), \end{aligned} \tag{4}$$

where $\boldsymbol{H} \in \mathbb{R}^{D \times K}$ a Helmert matrix (Lancaster, 1965) and the rows of $\boldsymbol{H}$ form an orthonormal basis that spans the tangent space of the simplex $T_{\boldsymbol{x}}\Delta^D = \{\boldsymbol{x} \in \mathbb{R}^K : \sum_{k=1}^K x_k = 0\}$. Characterization of the Helmert matrix, proof of $\mathcal{O}(K)$ complexity and other details are provided in the Supplement A.6. The volume change is

$$\det \boldsymbol{J}_\varphi = \det(\boldsymbol{H}_{1:D,1:D}) \prod_{i=1}^{K} \frac{1}{x_i}.$$

We use ILR because it is invariant to the category order, always giving the same mapping and hence geometry. Moreover it is an isometry from the simplex with the Aitchison geometry to Euclidean space as stated in Theorem 1. A direct consequence is that the paths traced by a Flow Matching model in Euclidean space are geometrically consistent with the Aitchison geometry, following the Aitchison geodesics.

Theorem 1 (Isometry, (Egozcue et al., 2003))**.** *Let $\langle \cdot, \cdot \rangle_A$ denote the Aitchison inner product on $\mathring{\Delta}^D$ and $\langle \cdot, \cdot \rangle_2$ the standard inner product on $\mathbb{R}^D$. For a Helmert matrix $\boldsymbol{H} \in \mathbb{R}^{D \times K}$, the ILR map satisfies*

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_A \;=\; \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{y}) \rangle_2 \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathring{\Delta}^D,$$

and, in particular, the ILR map is an isometry between $(\mathring{\Delta}^D, \langle \cdot, \cdot \rangle_A)$ and $(\mathbb{R}^D, \langle \cdot, \cdot \rangle_2)$.

Stick-breaking transform (SB) We additionally consider one order-dependent transformation that improves over the multiplicative logratio by the composition with a shift which centers the transformation (Carpenter et al., 2017). For $1 \le k \le D$, it is defined as:

$$\begin{aligned} \varphi &: \mathring{\Delta}^D \to \mathbb{R}^D, \quad \boldsymbol{x} \mapsto \boldsymbol{z}, \\ z_k &= \operatorname{mlr}(\boldsymbol{x})_k - \log\left(\frac{1}{K-k}\right), \\ \varphi^{-1} &: \mathbb{R}^D \to \mathring{\Delta}^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x}, \\ \boldsymbol{x} &= \operatorname{mlr}^{-1}(\boldsymbol{y}),\ y_k = z_k + \log\left(\frac{1}{K-k}\right). \end{aligned} \tag{5}$$

The last entry is $x_K = 1 - \sum_{k=1}^D x_k$. The term $\frac{1}{K-k}$ centers the transformation such that the zero vector in $\mathbb{R}^D$ maps to the vector $[\frac{1}{K}, .., \frac{1}{K}] \in \Delta^D$. For example a Gaussian with zero mean will correspond to a distribution centered at the midpoint of the simplex. SB has simple Jacobian determinant, $\det \boldsymbol{J}_\varphi = \prod_{i=1}^K \frac{1}{x_i}$, and is widely used in probabilistic modeling (Linderman et al., 2015; Carpenter et al., 2017).

3.2 Handling discrete data

Categorical observations $\boldsymbol{c} \in \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_K\}$ lie on the boundary $\partial\Delta^D$, requiring two additional tools: a way to map discrete observations into $\mathring{\Delta}^D$ for training, and a way to map generated continuous samples back to discrete observations at inference.

We do both using a stochastic interpolation scheme

$$\boldsymbol{x} = \lambda \boldsymbol{c} + (1 - \lambda)\boldsymbol{\varepsilon},$$

with $\boldsymbol{\varepsilon} \sim \operatorname{Dir}(\boldsymbol{\alpha})$. It associates each one-hot vector $\boldsymbol{c}$ with a continuous representation $\boldsymbol{x}$ on the simplex. The scheme is inspired by two distinct previous works. On one hand, it generalizes the deterministic interpolation of Chereddy and Femiani (2025) that represents $\boldsymbol{c}$ with $\boldsymbol{x}$ that is pulled from the corner of the simplex slightly towards the centroid. On the other hand, we leverage a theoretical result of Stark et al. (2024) to prove that we can exactly recover the discrete observations from the continuous relaxation.

Mapping observations to continuous space. During training, we sample $\boldsymbol{\varepsilon}$ for each categorical observation at every iteration. This transforms the discrete data into a Dirichlet-interpolated mixture $q_\lambda(\boldsymbol{x})$ characterized formally in Proposition 1, and the continuous generative model is used to approximate this density. The proposition effectively states that modeling the mixture distribution accurately recovers the true categorical distribution in the total variation sense.

Proposition 1. ***Categorical probabilities bound:*** *Let $\lambda \geq \frac{1}{2}$ so that the Dirichlet–interpolated mixture $q_\lambda(\boldsymbol{x}) = \sum_{k=1}^{K} p_k\, q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k)$ (see Eq. 7) has a.s. disjoint component supports contained in the strict* arg max *regions $\mathcal{R}_k := \{\boldsymbol{x} \in \mathring{\Delta}^D : x_k > x_j \ \forall j \neq k\}$. For any density $\tilde{q}$ on $\mathring{\Delta}^D$ define the induced (generated) categorical probabilities $\hat{p}_k := \int_{\mathcal{R}_k} \tilde{q}(\boldsymbol{x})\, d\boldsymbol{x}$. Then the total variation between true and generated categorical laws is bounded by the distance between the distributions:*

$$\mathrm{TV}(\boldsymbol{p}, \hat{\boldsymbol{p}}) = \tfrac{1}{2} \sum_{k=1}^{K} |\hat{p}_k - p_k| \ \leq \ \tfrac{1}{2}\, \|\tilde{q} - q_\lambda\|_1.$$

In particular, $\|\tilde{q} - q_\lambda\|_1 \to 0$ implies $\hat{\boldsymbol{p}} \to \boldsymbol{p}$ in total variation, and the arg max *discretization of $\tilde{q}$ recovers exactly the true categorical distribution.*

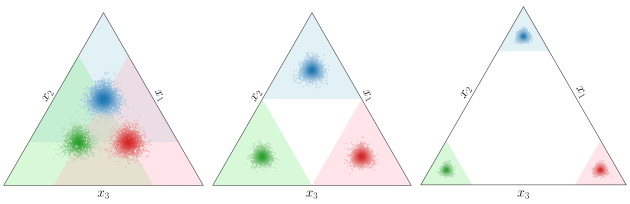
Mapping continuous samples to discrete observations. Samples drawn from the continuous generative model can be transformed back to discrete categories by selecting the category with the largest entry. Proposition 2, generalized from Stark et al. (2024), shows that given an interpolated point $\boldsymbol{x}$ for $\lambda > \frac{1}{2}$ the operator $\arg\max \boldsymbol{x}$ recovers the original category $\boldsymbol{c}$ exactly.

Proposition 2. ***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 $\boldsymbol{\varepsilon}$.*

Effect of the parameters. The interpolation scheme has two parameters, the interpolation constant λ and the Dirichlet concentration $\boldsymbol{\alpha}$. We next explain how they can be chosen without resorting to hyperparameter optimization.

Figure 2 illustrates the induced mixture for different λ; for large λ the probability mass remains close to the vertices where the geometry is more curved geometry. We characterize the density as a continuous transformation of a unimodal base distribution at the center of the space (e.g. $\boldsymbol{z}_0 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_D)$), and moving more of the mass towards the centroid likely makes this slightly easier. Consequently, we recommend using the smallest valid choice (Proposition 2). That is, we use $\lambda = \frac{1}{2}$.

In absence of prior information, the symmetric Dirichlet $\mathrm{Dir}(\alpha, .., \alpha)$ with $\alpha \in (0, \infty)$ is the only reasonable choice, with the marginal variance $\mathrm{Var}(x_k) = \frac{D}{K^2(\alpha K+1)}$. For $\alpha < 1$ the mass concentrates near the boundary, whereas for $\alpha > 1$ it concentrates at the interior. Following the above reasoning, we prefer having the mass away from the border, implying $\alpha \gg 1$.



$\lambda = \frac{1}{4}$ $\lambda = \frac{1}{2}$ $\lambda = \frac{3}{4}$

Figure 2: Dirichlet interpolation. The λ parameter controls the mixture distribution. For $\lambda \geq \frac{1}{2}$ the supports do not overlap and we can recover the categories. Large λ unnecessarily concentrates the mass around the simplex borders we want to avoid.

One way to reason about the optimal value is to investigate the variance more closely. For constant α, it decreases with K, suggesting a possible dependency with the dimensionality. However, under the ILR transform, the variance in the Euclidean coordinates is constant and independent of K, as formulated in Proposition 3. That is, even though the simplex marginals become tighter with K, the variance of the transformed variables $\boldsymbol{z} = \varphi(\boldsymbol{x})$ remains constant. The choice hence does not dramatically influence the difficulty of fitting the continuous model, and we settle with a uniform choice of sufficiently large $\alpha = 100$. For SB we do not have a similar theoretical result, but nevertheless make the same choice.

Proposition 3. ***Euclidean covariance:*** *Let $\boldsymbol{x} \sim \mathrm{Dir}(\alpha, .., \alpha)$ with $\alpha > 0$. Let $\boldsymbol{H} \in \mathbb{R}^{D \times K}$ be a Helmert matrix, and φ the ILR transform. Then the covariance of $\boldsymbol{z} = \varphi(\boldsymbol{x})$ is*

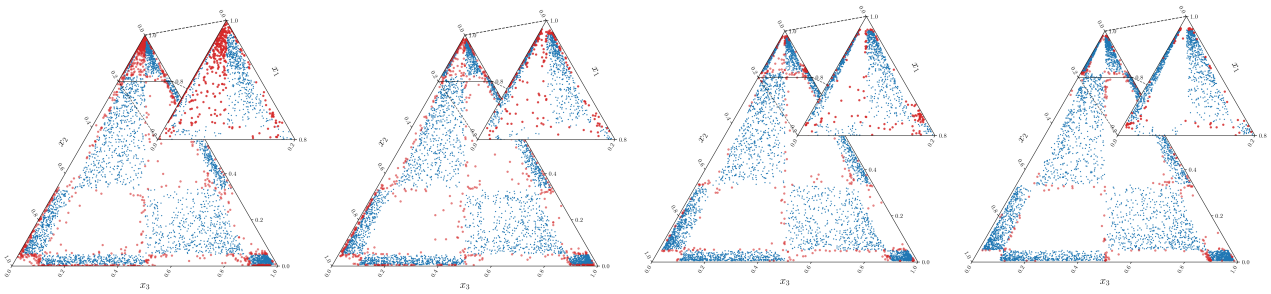
$$\mathrm{Cov}(\boldsymbol{z}) = \psi'(\alpha)\, \boldsymbol{I}_D,$$

where ψ' is the trigamma function. In particular, the covariance of $\boldsymbol{z}$ is independent of K.

3.3 Training, sampling and evaluation

The full method is characterized in Algorithms 1 and 2. The method can be used in two ways: When the data are discrete categorical observations, we apply Dirichlet interpolation during training and recover categories at sampling via the arg max operator. For compositional data (e.g. Fig. 3) the interpolation step is not needed. In both cases, sampling from the underlying continuous model requires solving an ODE as usual; we use standard numerical solvers such as Euler or Dormand–Prince.

We cannot directly evaluate the categorical probability $\Pr(C = k)$, since the model only defines a continuous density on the interior. We do not need it for learning (training uses the interpolated points) or sampling (obtained by the arg max-operation) and it is not provided



(a) LinearFM, 25.9% (b) SFM, 12.9% (c) FM-$\mathring{\Delta}$ (ILR), 6.8% (d) FM-$\mathring{\Delta}$ (SB), 5.4%

Figure 3: Samples from Checkerboard on the simplex. Red points indicate samples not aligned with the true density (% indicated in caption). The zoomed area shows the top region $x_1 \geq \frac{4}{5}$, emphasizing the differences.

Algorithm 1 Training of Simplex-to-Euclidean Flow Matching (FM-$\mathring{\Delta}$)

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 $\min \|\boldsymbol{v}_\theta(\boldsymbol{z}_t, t) - \boldsymbol{u}_t\|^2$ (Eq. (1))

Algorithm 2 Sampling with FM-$\mathring{\Delta}$

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

by many alternatives either. For example, Cheng et al. (2024) only provides an overly loose lower bound (see Supplement C.1 for a demonstration). Nonetheless, for evaluation it is useful to estimate $\Pr(C = k)$

Proposition 2 implies that for any $\alpha > 1$ the true density of the interpolated data $q_{\text{true}}(\boldsymbol{x})$ is a mixture of K Dirichlets, with the modes at the expected values $\boldsymbol{\mu}^{(k)} := \lambda \boldsymbol{e}_k + (1-\lambda)\frac{1}{K}$. 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

$$\widehat{\Pr}(C = k) = \frac{q_\theta(\boldsymbol{\mu}^{(k)})}{q_\lambda(\boldsymbol{\mu}^{(k)}|\boldsymbol{e}_k)}. \tag{6}$$

The model's density is computed by combining the change of variables given by φ and the instantaneous change of variables given by the flow model,

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

Each mixture component is supported on the region of the simplex where its category is the largest, $\{\boldsymbol{x} \in \mathring{\Delta}^D : x_k > x_j\ \forall j \neq k\}$, and is obtained by shifting and rescaling a Dirichlet distribution:

$$q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k) = \frac{1}{(1-\lambda)^D} \mathrm{Dir}\left(\frac{\boldsymbol{x} - \lambda \boldsymbol{e}_k}{1-\lambda}; \alpha\right). \tag{7}$$

4 EXPERIMENTS

We evaluate our approach on five tasks. We compare against other continuous models DirichletFM (Stark et al., 2024), DDSM (Avdeyev et al., 2023), Bit-Diffusion (Chen et al., 2023), Gumbel-Softmax FM (Tang et al., 2025) and α-Flow (Cheng et al., 2025). As additional baselines, we report results with some methods working directly in the discrete space, DFM (Gat et al., 2024), D3PM (Austin et al., 2021), MDLM (Sahoo et al., 2024), SEDD (Lou et al., 2024) and MultiFlow (Campbell et al., 2024), to measure the gap between the two families.

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. As explained in Section 2.1, our method is used with minibatch OT (w/OT) and without OT, we report both results. We fix $\lambda = \frac{1}{2}$ and $\alpha = 100$ in all cases. Our code implementation is available at `github.com/williwilliams3/simplexfm`.

Table 1: For NLL, $\leq$ is an upper bound and $\approx$ means estimate using Eq. 6. The results above the dotted line are from Cheng et al. (2024).

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

Compositional data The training data consists of continuous samples in $\mathring{\Delta}^2$, drawn from a checkerboard distribution in $\mathbb{R}^2$ and projected on the simplex with the inverse stick-breaking transform. We train directly on these continuous samples rather than on discrete observations. As shown in Fig. 3, the generated data from FM-$\mathring{\Delta}$ aligns more closely with the true distribution, whereas LinearFM and SFM produce many poor samples near the vertices. The number of invalid samples (points falling in regions of zero density) is more than twice for both, compared to our method.

Binarized MNIST The Binarized MNIST sets each pixel of MNIST to 1 with probability given by its intensity and 0 otherwise (Salakhutdinov and Murray, 2008); each pixel takes value in Δ^1. The velocity field is modeled using a convolutional neural network (Song and Ermon, 2020; Cheng et al., 2024). 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-$\mathring{\Delta}$ has the lowest NLL and FID, with relatively similar performance for all four variants. Some of the baselines (D3PM, DFM and DirichletFM) are substantially worse.

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 (on the signal) generation of the four symbols in Δ^3. 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. The velocity field is modeled with the same architecture as in Avdeyev et al. (2023), and following their setup, we evaluate generations by mapping both generated and test samples through a pretrained Sei model (Chen et al., 2022); the SP-MSE loss is the average Euclidean distance between these embeddings. Table 2 shows our method is again the best.

Table 2: DNA sequence generation. The results for the baselines are 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
Gumbel-Softmax FM	0.0290
DirichletFM	0.0269
LinearFM	0.0282
SFM	0.0258
FM-$\mathring{\Delta}$(SB)	0.0278
FM-$\mathring{\Delta}$(SB)w/OT	**0.0214**
FM-$\mathring{\Delta}$(ILR)	0.0259
FM-$\mathring{\Delta}$(ILR)w/OT	0.0224

Text8 We use the Text8 dataset (Mahoney, 2011), which models individual letters as elements of Δ^{26}. It contains 27 symbols (26 letters plus a blank space) and follows the standard 95K/5K/5K split, with each sample being a random chunk of length 256. The velocity field is modeled through a 12-layer diffusion transformer (Lou et al., 2024; Cheng et al., 2024), and the hyperparameters are selected based on the lowest validation error. As done by Campbell et al. (2024); Cheng et al. (2025), the evaluation is done generating 4000 samples which are tokenized according to the vocabulary of the GPT-J-6B model (Wang and Komatsuzaki, 2021), from the tokens the entropy is obtained and the GPT-J-6B model computes the reported NLL. We desire an entropy close to that of the data distribution and the NLL to be low. Table 3 shows that the discrete-space models are the best in terms of NLL, but our method is the best within the continuous relaxations. In terms of entropy, all methods behave fairly similarly.

Scalability The previous experiments all considered problems of fixed dimensionality. To study the performance as a function of K, we consider a setup where 10^6 samples are drawn from discrete distributions of $K = 2^1, \ldots, 2^9$ categories, with some probability vector $\boldsymbol{p} \in \Delta^D$. After training the model, we draw 10^5 samples and compute the KL divergence between the empirical density of the samples and the true distribution. Fig. 4 shows our method generally outperforms SFM and LinearFM especially for medium dimensionalities, and is comparable to the discrete-state SEDD for dimensionality up to $K = 2^7$. The Supplement C.2

Table 3: Text8: NLL and Entropy on the GPT-J-6B model. The baseline results are from the respective papers.

	Model	NLL ↓	Entropy	Diff
Discrete	MDLM	6.76	7.55	+0.07
	DFM	6.78	7.58	+0.10
	D3PM	6.93	7.38	-0.10
	SEDD	**6.49**	7.17	-0.31
	MultiFlow	6.73	7.39	-0.09
Continuous	LinearFM	7.35	7.62	+0.14
	α-Flow(α=-0.5)	7.14	7.37	-0.11
	α-Flow(α=0.5)	7.00	7.42	-0.06
	α-Flow(α=1)	7.09	7.51	**+0.03**
	SFM	6.85	7.38	-0.10
	FM-$\mathring{\Delta}$(ILR)	6.81	7.39	-0.09
	FM-$\mathring{\Delta}$(ILR)w/OT	6.89	7.38	-0.10
	Data	4.10	7.48	0

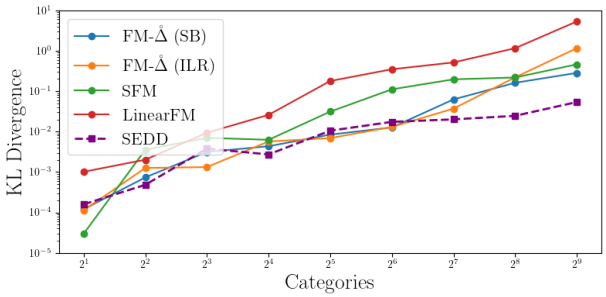


Figure 4: Divergence between the ground truth and estimated categorical probabilities, for problems of varying number of categories.

has additional experiments under the same setup on the effect of α and λ, which were kept fixed in all of the main experiments.

5 DISCUSSION

The progress of continuous relaxations for generative modeling of discrete data is extremely rapid. We briefly detail the connections to the most recent works.

Diederen and Zamboni (2025) used the ILR transformation in modeling compositional data, as one transformation within a composition of multiple ones. They did not, however, consider discrete observations at all. Potapczynski et al. (2020), in turn, considered an alternative bijection building on softmax$_{++}$ that also allows recovering discrete data with arg max, but their solution is designed specifically for a variational autoencoder as the generative model, with no obvious generalization for other models.

Chereddy and Femiani (2025) proposed a diffusion model that leverages special cases of our machinery. Inspired by logistic-normal models (Aitchison and Shen, 1980), their Gaussian-Softmax diffusion model uses the push-forward of a Gaussian with the additive logratio transform (Eq. 3), using a deterministic interpolation scheme similar to ours ($\lambda \boldsymbol{e}_k + (1-\lambda)\frac{1}{K}$ with $\lambda = 0.99$) to handle discrete observations. However, their transformation leaves the mass very close to the borders and the solution is specifically geared towards use of the order-dependent ALR transformation. Cheng et al. (2025) proposed an alternative bijection on the logit-simplex space, as a limiting case of the information geometry, resulting in a geometry that is intrinsically different from ours. Closer inspection of the differences would be highly interesting future work. Mahabadi et al. (2024) and Tae et al. (2025) also used the logit-simplex space, but treated each discrete token as a point in the probability simplex close to a vertex, rather than making a formal connection with the vertices and well-defined points in the interior.

We specifically focused on accounting for the simplex geometry, but some methods also operate directly in a Euclidean space, by adding random noise directly for the one-hot observations in $\mathbb{R}^K$. For example, Hoogeboom et al. (2020, 2021) proposed a variational objective for learning a generative model and also used the arg max operation like us. These models miss all connection to the simplex and its natural geometry. Truncated Flows (Tan et al., 2022) extend this idea by learning normalizing flows defined over bounded regions of $\mathbb{R}^K$ (or a latent space), recovering the exact categories if the regions are disjoint, and Categorical Normalizing Flows (Lippe and Gavves, 2020) model categories using mixtures of logistic distributions, with a particular focus on structured data such as graphs.

Our method aligns with these concurrent works, pushing forward the boundary of how to best use established tools for continuous generation for discrete data, especially in terms of generality. Our results indicate the method is competitive with the rest while achieving a certain degree of conceptual and computational elegance due to accounting for the simplex geometry without needing complex tools.

6 CONCLUSION

We proposed a principled method that constructs a bridge between the generation of continuous data in 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 using flow matching with highly promising results, constraining to this choice to keep

the presentational and empirical complexity at bay, but other generative models like (Karras et al., 2022; Song et al., 2023; Geng et al., 2025) could directly be plugged in as future work.

Acknowledgements

The authors were supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI, and additionally by grants: 363317 (BW, AK), 345811 (BW, AK), 348952 (MH), 369502 (MH), 359207 and the Finland Fellowship granted by Aalto University (VYS). The authors acknowledge the research environment provided by ELLIS Institute Finland, and then CSC - IT Center for Science, Finland, and the Aalto Science-IT project for computational resources.

References

John Aitchison. A new approach to null correlations of proportions. *Journal of the International Association for Mathematical Geology*, 13(2):175–189, 1981.

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

John Aitchison. Principal component analysis of compositional data. *Biometrika*, 70(1):57–65, 1983.

John Aitchison and Sheng M Shen. Logistic-normal distributions: Some properties and uses. *Biometrika*, 67(2):261–272, 1980.

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. *Advances in Neural Information Processing Systems*, 37:54787–54819, 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.

Sathvik Chereddy and John Femiani. Sketchdnn: Joint continuous-discrete diffusion for cad sketch generation. *arXiv preprint arXiv:2507.11579*, 2025.

Oscar Davis, Samuel Kessler, Mircea Petrache, İsmail İ Ceylan, Michael Bronstein, and Avishek J Bose. Fisher flow matching for generative modeling over discrete data. *Advances in Neural Information Processing Systems*, 37:139054–139084, 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.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Emiel Hoogeboom, Taco S Cohen, and Jakub M Tomczak. Learning discrete distributions by dequantization. *arXiv preprint arXiv:2001.11235*, 2020.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

HO Lancaster. The helmert matrices. *The American Mathematical Monthly*, 72(1):4–12, 1965.

John M Lee. *Introduction to Riemannian manifolds*, volume 2. Springer, 2018.

Scott Linderman, Matthew J Johnson, and Ryan P Adams. Dependent multinomial models made easy: Stick-breaking with the pólya-gamma augmentation. *Advances in neural information processing systems*, 28, 2015.

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.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Phillip Lippe and Efstratios Gavves. Categorical normalizing flows via continuous transformations. *arXiv preprint arXiv:2006.09790*, 2020.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, pages 32819–32848, 2024.

Rabeeh Karimi Mahabadi, Hamish Ivison, Jaesung Tae, James Henderson, Iz Beltagy, Matthew E Peters, and Arman Cohan. Tess: Text-to-text self-conditioned simplex diffusion. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2347–2361, 2024.

Matt Mahoney. Large text compression benchmark, 2011.

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.

Andres Potapczynski, Gabriel Loaiza-Ganem, and John P Cunningham. Invertible gaussian reparameterization: Revisiting the gumbel-softmax. *Advances in Neural Information Processing Systems*, 33:12311–12321, 2020.

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 T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *International Conference on Machine Learning*, pages 52584–52619. PMLR, 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.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023.

Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to dna sequence design. In *International Conference on Machine Learning*, pages 46495–46513. PMLR, 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.

Jaesung Tae, Hamish Ivison, Sachin Kumar, and Arman Cohan. Tess 2: A large-scale generalist diffusion language model. *arXiv preprint arXiv:2502.13917*, 2025.

Shawn Tan, Chin-Wei Huang, Alessandro Sordoni, and Aaron Courville. Learning to dequantise with truncated flows. In *International conference on learning representations*, 2022.

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.

Ben Wang and Aran Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Simplex-to-Euclidean Bijections for Categorical Flow Matching: Supplementary Materials

A MATHEMATICAL DERIVATIONS

A.1 Proof of proposition 1

Proposition 1. ***Categorical probabilities bound:*** *Let $\lambda \geq \frac{1}{2}$ so that the Dirichlet–interpolated mixture $q_\lambda(\boldsymbol{x}) = \sum_{k=1}^{K} p_k\, q_\lambda(\boldsymbol{x} \mid \boldsymbol{e}_k)$ has a.s. disjoint component supports contained in the strict* $\arg\max$ *regions $\mathcal{R}_k := \{\boldsymbol{x} \in \mathring{\Delta}^D : x_k > x_j \ \forall j \neq k\}$. For any density $\tilde{q}$ on $\mathring{\Delta}^D$ define the induced (generated) categorical probabilities $\hat{p}_k := \int_{\mathcal{R}_k} \tilde{q}(\boldsymbol{x})\, d\boldsymbol{x}$. Then the total variation between true and generated categorical laws is bounded by the distance between the distributions:*

$$\mathrm{TV}(p, \hat{p}) = \tfrac{1}{2} \sum_{k=1}^{K} |\hat{p}_k - p_k| \ \leq \ \tfrac{1}{2}\, \|\tilde{q} - q_\lambda\|_1.$$

In particular, $\|\tilde{q} - q_\lambda\|_1 \to 0$ implies $\hat{p} \to p$ in total variation, and the $\arg\max$ *discretization of $\tilde{q}$ recovers exactly the true categorical distribution.*

Proof. Because $\lambda \geq \frac{1}{2}$ and Prop. 2 the mixture places almost surely all mass of component k inside $\mathcal{R}_k$, hence $p_k = \int_{\mathcal{R}_k} q_\lambda(\boldsymbol{x})\, d\boldsymbol{x}$. For any measurable set B, $\left| \int_B (\tilde{q}(\boldsymbol{x}) - q_\lambda(\boldsymbol{x}))\, d\boldsymbol{x} \right| \leq \int_B |\tilde{q}(\boldsymbol{x}) - q_\lambda(\boldsymbol{x})|\, d\boldsymbol{x}$, giving a point-wise bound. Summing and using $\sum_k \int_{\mathcal{R}_k} |\tilde{q}(\boldsymbol{x}) - q_\lambda(\boldsymbol{x})|\, d\boldsymbol{x} \leq \int |\tilde{q}(\boldsymbol{x}) - q_\lambda(\boldsymbol{x})|\, d\boldsymbol{x}$ yields

$$\tfrac{1}{2} \sum_{k=1}^{K} |\hat{p}_k - p_k| \ \leq \ \tfrac{1}{2}\, \|\tilde{q} - q_\lambda\|_1.$$

□

A.2 Proof of proposition 2

Proposition 2. ***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 $\boldsymbol{\varepsilon}$.*

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. □

A.3 Proof of proposition 3

Proposition 3. ***Euclidean covariance:*** *Let $\boldsymbol{x} \sim \mathrm{Dir}(\alpha, .., \alpha)$ with $\alpha > 0$. Let $\boldsymbol{H} \in \mathbb{R}^{D\times K}$ be a Helmert matrix, and φ the ILR transform. Then the covariance of $\boldsymbol{z} = \varphi(\boldsymbol{x})$ is*

$$\mathrm{Cov}(\boldsymbol{z}) = \psi'(\alpha)\, \boldsymbol{I}_D,$$

where ψ' is the trigamma function. In particular, the covariance of $\boldsymbol{z}$ is independent of K.

Proof. Recall $D := K - 1$ and $\boldsymbol{x} \sim \mathrm{Dir}(\alpha, .., \alpha)$. Let ψ denote the digamma function and $\psi' = \frac{d}{dx}\psi(x)$ the trigamma function. The first and second moments of the log-random variable are:

$$\mathbb{E}[\log x_k] = \psi(\alpha) - \psi(K\alpha), \qquad \mathrm{Cov}(\log \boldsymbol{x}) = \psi'(\alpha)\, \boldsymbol{I}_K - \psi'(K\alpha)\, \mathbf{1}\mathbf{1}^\top.$$

Therefore the covariance of $\boldsymbol{z} = \boldsymbol{H} \log \boldsymbol{x}$ is

$$\mathrm{Cov}(\boldsymbol{z}) = \boldsymbol{H}\ \mathrm{Cov}(\log \boldsymbol{x})\, \boldsymbol{H}^\top = \psi'(\alpha)\, \boldsymbol{H}\, \boldsymbol{H}^\top - \psi'(K\alpha)\, \boldsymbol{H}\, \mathbf{1}\mathbf{1}^\top\, \boldsymbol{H}^\top.$$

Use the orthonormality of $\boldsymbol{H}$ and the sum-to-zero property of each row: $\boldsymbol{H}\, \boldsymbol{H}^\top = \boldsymbol{I}_D$, and by construction of the ILR basis $\boldsymbol{H}\mathbf{1} = 0$. Substituting these simplifies the covariance to

$$\mathrm{Cov}(\boldsymbol{z}) = \psi'(\alpha)\, \boldsymbol{I}_D.$$

Hence, the coordinates z_k are uncorrelated and each has constant variance $\psi'(\alpha)$ independent of K, proving the proposition. □

A.4 Proof of Theorem 1

Theorem 1 (Isometry, (Egozcue et al., 2003))**.** *Let $\langle \cdot, \cdot \rangle_A$ denote the Aitchison inner product on $\mathring{\Delta}^D$ and $\langle \cdot, \cdot \rangle_2$ the standard inner product on $\mathbb{R}^D$. For a Helmert matrix $\boldsymbol{H} \in \mathbb{R}^{D\times K}$, the ILR map satisfies*

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_A \ = \ \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{y}) \rangle_2 \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathring{\Delta}^D,$$

and, in particular, the ILR map is an isometry between $(\mathring{\Delta}^D, \langle \cdot, \cdot \rangle_A)$ and $(\mathbb{R}^D, \langle \cdot, \cdot \rangle_2)$.

We provide the proof where the Helmert matrix is the orthonormal basis of the tangent space of the simplex $T_{\boldsymbol{x}}\mathring{\Delta}^D = \{\boldsymbol{v} \in \mathbb{R}^K : \sum_{i=1}^K v_i = 0\}$. Refer to Egozcue et al. (2003) for a proof independent of the choice of basis for $T_{\boldsymbol{x}}\mathring{\Delta}^D$.

Proof. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathring{\Delta}^D$, and set $\boldsymbol{u} := \log \boldsymbol{x}$ and $\boldsymbol{v} := \log \boldsymbol{y}$, the Aitchison inner product is

$$\begin{aligned}\langle \boldsymbol{x}, \boldsymbol{y} \rangle_A &= \frac{1}{2K} \sum_{i,j=1}^{K} \log \frac{x_i}{x_j} \log \frac{y_i}{y_j} = \frac{1}{2K} \sum_{i,j=1}^{K} (u_i - u_j)(v_i - v_j), \\ &= \frac{1}{2K} \sum_{i,j=1}^{K} (u_i v_i - u_i v_j - u_j v_i + u_j v_j) = \langle \boldsymbol{u}, \boldsymbol{v} \rangle_2 - \tfrac{1}{K} \langle \boldsymbol{u}, \mathbf{1} \rangle_2 \langle \boldsymbol{v}, \mathbf{1} \rangle_2.\end{aligned}$$

On the other hand, the rows of $\boldsymbol{H}$ form a basis of $T_{\boldsymbol{x}}\mathring{\Delta}^D$, then the orthogonal projection of a vector $\boldsymbol{y} \in \mathbb{R}^K$ onto $T_{\boldsymbol{x}}\mathring{\Delta}^D$ is given by $\mathrm{proj}_{T_{\boldsymbol{x}}\mathring{\Delta}^D}(\boldsymbol{y}) = \boldsymbol{H}^\top \boldsymbol{H}\, \boldsymbol{y}$. This projection operator is equivalent to the matrix $\boldsymbol{P} = \boldsymbol{I}_K - \frac{1}{K}\mathbf{1}\mathbf{1}^\top$, so that $\mathrm{proj}_{T_{\boldsymbol{x}}\mathring{\Delta}^D}(\boldsymbol{y}) = \left(\boldsymbol{I}_K - \frac{1}{K}\mathbf{1}\mathbf{1}^\top\right) \boldsymbol{y}$. The norm in the Euclidean space is

$$\langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{y}) \rangle_2 = \boldsymbol{u}^\top \boldsymbol{H}^\top \boldsymbol{H}\, \boldsymbol{v} = \boldsymbol{u}^\top (\boldsymbol{I}_K - \frac{1}{K}\mathbf{1}\mathbf{1}^\top)\, \boldsymbol{v} = \langle \boldsymbol{u}, \boldsymbol{v} \rangle_2 - \tfrac{1}{K} \langle \boldsymbol{u}, \mathbf{1} \rangle_2 \langle \boldsymbol{v}, \mathbf{1} \rangle_2.$$

Therefore both the inner products coincide and in particular for all $\boldsymbol{x} \in \mathring{\Delta}^D$

$$\|\boldsymbol{x}\|_A = \|\varphi(\boldsymbol{x})\|_2.$$

□

A.5 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 $1 \leq k \leq D$

$$\varphi^{-1} : \mathbb{R}^D \to \Delta^D, \quad \boldsymbol{z} \mapsto \boldsymbol{x}, \quad x_k = \left(1 - \sum_{i=1}^{k-1} x_i\right) \sigma(y_k),\ y_k = z_k + \log \frac{1}{K-k}, \tag{8}$$

where $\sigma(\cdot)$ denotes the sigmoid function and the last entry is $x_K = 1 - \sum_{i=1}^{D} x_i$. Proposition 4 shows the equivalence between the stick-breaking and unit-simplex inverse transforms.

Proposition 4. *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 4 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

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

A.6 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

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

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

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

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 $\det(\boldsymbol{H}_{1:D,1:D}) = \frac{1}{\sqrt{K}}$.

Computational complexity We derive the linear computational complexity of the ILR bijection. The (non-full) Helmert matrix $\boldsymbol{H} \in \mathbb{R}^{D\times K}$ has a simple recursive structure. For $i = 1,\ldots,D$, the i-th row contains i entries equal to $1/\sqrt{i(i+1)}$, one entry equal to $-i/\sqrt{i(i+1)}$ in column $i{+}1$, and zeros elsewhere:

$$\boldsymbol{H} = \begin{bmatrix}
\frac{1}{\sqrt{1\cdot 2}} & -\frac{1}{\sqrt{1\cdot 2}} & 0 & 0 & \cdots & 0\\
\frac{1}{\sqrt{2\cdot 3}} & \frac{1}{\sqrt{2\cdot 3}} & -\frac{2}{\sqrt{2\cdot 3}} & 0 & \cdots & 0\\
\frac{1}{\sqrt{3\cdot 4}} & \frac{1}{\sqrt{3\cdot 4}} & \frac{1}{\sqrt{3\cdot 4}} & -\frac{3}{\sqrt{3\cdot 4}} & \cdots & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\
\frac{1}{\sqrt{D\cdot K}} & \frac{1}{\sqrt{D\cdot K}} & \cdots & \frac{1}{\sqrt{D\cdot K}} & \frac{1}{\sqrt{D\cdot K}} & -\frac{D}{\sqrt{D\cdot K}}
\end{bmatrix}.$$

Let $\boldsymbol{v} = \log \boldsymbol{x}$. Then $\boldsymbol{z} = \boldsymbol{H}\boldsymbol{v}$ has components

$$z_i = \frac{1}{\sqrt{i(i+1)}}\left(\sum_{j=1}^{i} v_j - i\,v_{i+1}\right).$$

Defining the cumulative sum $S_i = \sum_{j=1}^{i} v_j$ (computed in $\mathcal{O}(K)$), we obtain

$$z_i = \frac{S_i - i\,v_{i+1}}{\sqrt{i(i+1)}}, \qquad i = 1,\ldots,D,$$

Thus the full matrix–vector product $\boldsymbol{z} = \boldsymbol{H}\boldsymbol{v}$ can be computed in $\mathcal{O}(K)$ time without explicitly forming $\boldsymbol{H}$.

A.7 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 $\boldsymbol{G}_\varphi = \boldsymbol{J}_\varphi^\top \boldsymbol{J}_\varphi \in \mathbb{R}^{D\times D}$;

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

This is a rank-1 update of a diagonal matrix, set $\boldsymbol{A} = \operatorname{diag}\left(\frac{1}{x_1},\dots,\frac{1}{x_D}\right)$, 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:

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

A.8 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 $\boldsymbol{x}\in\mathring{\Delta}^D$ 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 $\boldsymbol{x}\in\mathring{\Delta}^D$ 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:

$$\log p_1(\boldsymbol{z}_1) = \log p_0(\boldsymbol{z}_0) - \int_0^1 \operatorname{div}(\boldsymbol{v}_s^\theta)(\boldsymbol{z}_s)\,\mathrm{d}s. \tag{9}$$

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

$$\log q_\theta(\boldsymbol{x}) = \log p_1(\varphi(\boldsymbol{x})) + \log\left|\frac{\partial\varphi(\boldsymbol{x})}{\partial\boldsymbol{x}}\right|. \tag{10}$$

Combining Equations (9) and (10) 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 $p_0(\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_0\mid\mathbf{0},\boldsymbol{I})$. If the base distribution is the uniform distribution, $\boldsymbol{x}_0\sim\operatorname{Unif}(\Delta^D)$, 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 $\boldsymbol{\varepsilon}\sim\operatorname{Dir}(\alpha)$ 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,

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

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

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

A.9 Standardize the data

To ensure that the mean compositions of different mixture components are comparable across varying dimensions, we examine how their Aitchison norms scale with the number of categories. The scaling determines how far each component is from the zero vector in $\mathbb{R}^D$. Recall $D = K - 1$ and $\boldsymbol{\mu}^{(k)} = \lambda \boldsymbol{e}_k + (1-\lambda)\frac{1}{K}$. The squared Aitchison norm of $\boldsymbol{\mu}^{(k)}$ is

$$\|\boldsymbol{\mu}^{(k)}\|_A^2 = \frac{1}{2K} \sum_{i=1}^{K} \sum_{j=1}^{K} \left(\log \frac{\mu_i^{(k)}}{\mu_j^{(k)}} \right)^2 = \frac{1}{K} \sum_{j \neq k} \left(\log \frac{\mu_k^{(k)}}{\mu_j^{(k)}} \right)^2 = \frac{D}{K} \left[\log \left(1 + \frac{K\lambda}{1-\lambda} \right) \right]^2. \tag{11}$$

For $\lambda = \frac{1}{2}$, this becomes

$$\|\boldsymbol{\mu}^{(k)}\|_A = \sqrt{\frac{D}{K}} \log(K+1).$$

The mean compositions therefore move logarithmically farther from the origin as the number of categories increases, indicating a dimensionality-dependent scaling. Although this effect could be removed by normalizing with $1/\|\boldsymbol{\mu}^{(k)}\|_A$, our preliminary experiments showed that this adjustment had no notable benefit (see Section B).

B EXPERIMENTAL DETAILS

The empirical experiments were done in two distinct computing environments for practical reasons, with differing hardware. Hence, we indicate the hardware separately for each experiment.

Compositional Data We evaluate the models by drawing 5000 samples using the Dopri5 solver. The velocity field is modeled with a fully connected network consisting of 4 hidden layers with 512 units each. Training is performed on AMD Rome 7H12 CPUs on a computer cluster.

Binarized MNIST We generate samples and estimate likelihoods using the Euler solver with 300 steps for all four variants of our method. Non–cherry-picked generated examples are shown in Fig. 5. Following Cheng et al. (2024), we adopt the CNN architecture of Song and Ermon (2020), modified such that each convolutional layer receives a distinct time embedding. The negative log-likelihood (NLL) approximation is standardized by evaluating the log-density on both the test image x and its flipped version $1-x$, ensuring the total sum is one. FID statistics (mean and covariance) are computed with the InceptionV3 model (Szegedy et al., 2016) over the entire training dataset, and FID scores are calculated between 1000 generated samples and the training data. We use the same hyperparameters as Cheng et al. (2024): batch size 256 and initial learning rate 3×10^{-4}. Each model is trained for approximately 500 epochs on a single NVIDIA Volta V100 GPU.

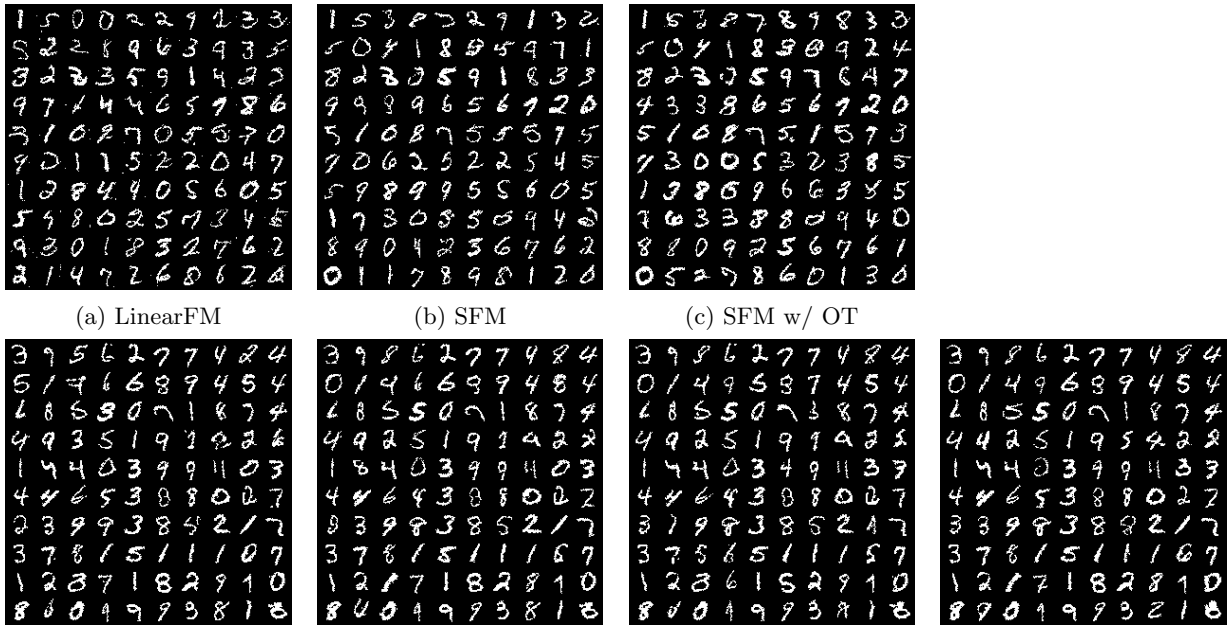
Promoter Design Following Avdeyev et al. (2023), sequences in the training data are randomly offset by up to 10 positions during training. The velocity field is parameterized by the same network as in Avdeyev et al. (2023), consisting of 20 stacks of one-dimensional convolutional layers. Training is conducted for 200K steps on a single NVIDIA Ampere A100 GPU (40GB) over the hyperparameter grid shown in Table 4. Samples are generated using the Euler solver with 300 steps for all four method variants. Evaluation follows Avdeyev et al. (2023), using the Sei model with the H3K4me3 chromatin mark to predict active promoters on both generated and test sequences. Performance is measured via the squared difference between Sei predictions on both datasets (SP-MSE). For each run, model weights are selected based on the CFM validation loss, and the best hyperparameters are chosen with respect to SP-MSE on the validation set.

Text8 Samples are generated using the Euler solver with 300 steps for all method variants. The velocity field is modeled with a 12-layer diffusion transformer, following Lou et al. (2024); Cheng et al. (2024). We conduct a grid search over hyperparameters and select the best model based on validation error. Only the ILR bijection is considered, and the hyperparameter grid is:

$$\text{OT} = [\text{False}, \text{True}], \quad \text{scaling (Eq. 11)} = [\text{True}, \text{False}], \quad \text{batch size} = [128, 216], \quad \text{lr} = [10^{-4}, 2 \times 10^{-4}].$$

Training is performed in parallel on 4 NVIDIA Ampere A100 GPUs (40GB) for approximately 400 epochs over 3 days.

Scalability We evaluate scalability by generating $N = 10^5$ samples $\{\hat{\boldsymbol{x}}^{(i)}\}_{i=1}^N$ with the Euler solver using 200 steps and estimating $\hat{\boldsymbol{p}} = \frac{1}{N}\sum_{i=1}^N \hat{\boldsymbol{x}}^{(i)}$. The true distribution $\boldsymbol{p} \in \Delta^D$ is defined as $p_1 = \frac{1}{2}$ and $[p_2, \ldots, p_K] \sim \frac{1}{2}\text{Unif}(\Delta^D)$. The velocity field is modeled by a fully connected network with 4 hidden layers of 512 units. The input dimension is D+64 for our method and K+64 for SFM, where 64 corresponds to the dimension of the sinusoidal time embedding. All models share the same fixed hyperparameters. Training is performed on AMD Rome 7H12 CPUs on a computer cluster.



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

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

Figure 5: Samples from BMNIST from the different methods. LinearFM draws samples of visually lower quality than the rest of the methods.

C ADDITIONAL EXPERIMENTS

C.1 Estimation of the categorical probabilities

We evaluate the accuracy of our estimator for categorical log-probabilities under the same setup as the *Scalability* experiment. We consider dimensions $D = 2, 2^2, \ldots, 2^8$ and compare the Stick-Breaking (SB) transform against SFM. The true probabilities $\boldsymbol{p} \in \Delta^D$ and the velocity field is explained in the Section B.

Recall the estimator for the discrete probabilities is given by $\widehat{\Pr}(C = k) = \frac{q_\theta(\boldsymbol{\mu}^{(k)})}{q_\lambda(\boldsymbol{\mu}^{(k)}|\boldsymbol{e}_k)}$. The numerator $q_\theta(\cdot)$ is computed using the Euler integrator with 200 steps, and its divergence term is approximated via the Hutchinson trace estimator.

We compare our estimator $\widehat{\Pr}(C = k)$ to the lower bound proposed by Cheng et al. (2024). Figure 6 shows that our estimator closely matches the true probabilities up to 2^5 categories in terms of KL divergence, whereas the

Table 4: Values of SP-MSE on validation and test data for the tested hyperparameters. WD is weight decay and β_1 is a parameter of the Adam optimizer.

Method	Bijection	OT	Batch	Opt	WD	β_1	Step	SP-MSE(val)	SP-MSE(test)
FM-Å	SB	False	64	Adam	0	0.85	40000	0.0251	0.0278
FM-Å	SB	False	64	Adam	0	0.95	40000	0.0321	0.0341
FM-Å	SB	False	128	Adam	0	0.85	30000	0.0327	0.0353
FM-Å	SB	False	128	Adam	0	0.95	30000	0.0406	0.0443
FM-Å	SB	False	128	Adam	10^{-5}	0.85	120000	0.0435	0.0447
FM-Å	SB	False	64	Adam	10^{-5}	0.85	200000	0.0506	0.0512
FM-Å	SB	False	128	Adam	10^{-5}	0.95	120000	0.0509	0.0514
FM-Å	SB	False	64	Adam	10^{-5}	0.95	200000	0.0549	0.0554
FM-Å	SB	True	64	Adam	0	0.85	40000	0.0213	0.0214
FM-Å	SB	True	128	Adam	0	0.85	30000	0.0314	0.0325
FM-Å	SB	True	128	Adam	10^{-5}	0.95	120000	0.0363	0.038
FM-Å	SB	True	128	Adam	0	0.95	30000	0.0387	0.0409
FM-Å	SB	True	128	Adam	10^{-5}	0.85	120000	0.0387	0.0392
FM-Å	SB	True	64	Adam	10^{-5}	0.95	190000	0.0405	0.0435
FM-Å	SB	True	64	Adam	0	0.95	40000	0.0591	0.0569
FM-Å	SB	True	64	Adam	10^{-5}	0.85	190000	0.0613	0.0642
FM-Å	ILR	False	64	Adam	0	0.85	40000	0.0252	0.0259
FM-Å	ILR	False	64	Adam	0	0.95	40000	0.0321	0.0335
FM-Å	ILR	False	128	Adam	0	0.85	30000	0.0328	0.0346
FM-Å	ILR	False	128	Adam	0	0.95	30000	0.0406	0.0443
FM-Å	ILR	False	128	Adam	10^{-5}	0.85	120000	0.0436	0.0449
FM-Å	ILR	False	64	Adam	10^{-5}	0.85	200000	0.0508	0.0526
FM-Å	ILR	False	128	Adam	10^{-5}	0.95	120000	0.0511	0.0511
FM-Å	ILR	False	64	Adam	10^{-5}	0.95	200000	0.0548	0.0553
FM-Å	ILR	True	64	Adam	0	0.85	40000	0.0213	0.0224
FM-Å	ILR	True	128	Adam	0	0.85	30000	0.0314	0.0317
FM-Å	ILR	True	128	Adam	10^{-5}	0.95	120000	0.0362	0.0384
FM-Å	ILR	True	128	Adam	0	0.95	30000	0.0387	0.0419
FM-Å	ILR	True	128	Adam	10^{-5}	0.85	120000	0.0387	0.039
FM-Å	ILR	True	64	Adam	10^{-5}	0.95	190000	0.0405	0.042
FM-Å	ILR	True	64	Adam	0	0.95	40000	0.0588	0.0581
FM-Å	ILR	True	64	Adam	10^{-5}	0.85	190000	0.0612	0.0626

Table 5: Run time measurements with varying data dimensionality D and fixed batch size $B = 512$. All results are reported in milliseconds as a mean with two standard deviations across 1000 iterations.

Method	$D = 2^2$ (ms)	$D = 2^3$ (ms)	$D = 2^4$ (ms)	$D = 2^5$ (ms)	$D = 2^6$ (ms)	$D = 2^7$ (ms)
SFM	45 ± 3	45 ± 4	48 ± 3	48 ± 3	50 ± 3	54 ± 4
FM-Å(ALR)	76 ± 12	74 ± 10	76 ± 12	82 ± 12	88 ± 5	85 ± 10
FM-Å(ILR)	81 ± 12	84 ± 12	83 ± 12	85 ± 12	88 ± 6	92 ± 8

Table 6: Run time measurements with varying batch size B and fixed data dimensionality $D = 64$. All results are reported in milliseconds as a mean with two standard deviations across 1000 iterations.

Method	$B = 2^6$ (ms)	$B = 2^7$ (ms)	$B = 2^8$ (ms)	$B = 2^9$ (ms)	$B = 2^{10}$ (ms)	$B = 2^{11}$ (ms)
SFM	15 ± 2	18 ± 1	29 ± 1	51 ± 3	93 ± 10	183 ± 36
FM-$\mathring{\Delta}$(ALR)	13 ± 4	18 ± 6	42 ± 6	89 ± 6	171 ± 18	346 ± 36
FM-$\mathring{\Delta}$(ILR)	13 ± 4	18 ± 6	43 ± 9	83 ± 12	166 ± 12	339 ± 32

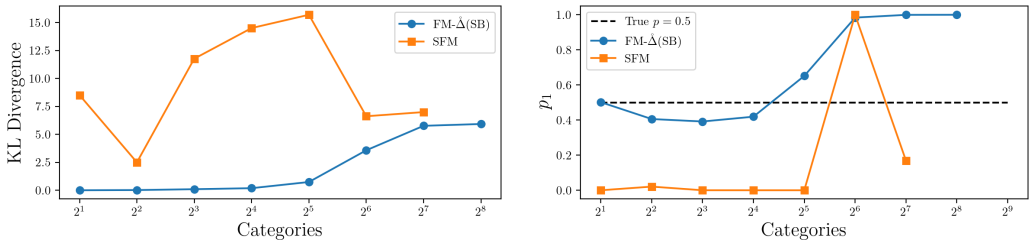


Figure 6: Left: KL divergence between the true categorical probabilities and either the estimation $\widehat{\Pr}(C = k)$ (blue) or lower-bound approximations given by SFM (orange). Right: Estimated value and lower bound of p_1 compared to the true value $p_1 = 0.5$. Missing values correspond to numerical overflows during the computation.

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.

C.2 Effect of the parameters

Using the same setup as in Scalability, we train a velocity network across the values $\alpha \in \{1, 10, 100, \infty\}$, $\lambda \in \{0.5, 0.75, 0.99\}$, with and without scaling (see Eq. 11). We only consider the ILR bijection. The case $\alpha = 1$ corresponds to the uniform distribution over each arg max region ($x_i > x_j$ for all $i \neq j$), and $\alpha = \infty$ to a deterministic interpolation, namely $\lambda \boldsymbol{c} + (1 - \lambda)\frac{1}{K}$. We generate $100,000$ samples 5 times and plot the mean values with 2 standard deviation bands.

Figure 7 shows that scaling improves the accuracy when there are only 2 categories, but scaling degrades the performance once the number of categories exceeds 2^5. The uniform distribution ($\alpha = 1$) is slightly more accurate for $\lambda \in \{0.5, 0.75\}$ than $\lambda = 0.99$. The performance of the method is more sensitive to the the values of α and λ as K increases, but we intentionally did not attempt fine-tuning α and λ to maximize the empirical performance, since we want an easy-to-use pipeline.

C.3 Run Times

We compare the run times between SFM and FM-$\mathring{\Delta}$(ALR) with the same setup as the *Scalability* experiment. To measure the effect of the number of categories, we consider dimensions $D = 2^2, \ldots, 2^7$ with a fixed batch size of $B = 512$, presented in Table 5. Then, to measure the effect of batch size, we consider batch sizes $B = 2^6, 2^7, \ldots, 2^11$ with a fixed $D = 64$, presented in Table 6. All the run times are reported with mean and two standard deviations over 1000 training iterations in machines with Xeon E5 2680 v3 2.50GHz CPUs, we do not use GPUs for these measurements.

C.4 Numerical results for Additive Logratio and Multiplicative Logratio bijections

In Section 3.1 we defined ALR and MLR as possible bijections, before introducing our proposed bijections SB and ILR. It would still be useful to clarify how well ALR and MLR would work.

We now ran the Checkerboard, BMNIST and DNA experiments with the ALR and MLR mappings, reporting the results in Table 7. SB remains the best on DNA and Checkerboard and ILR on binarized MNIST, but both

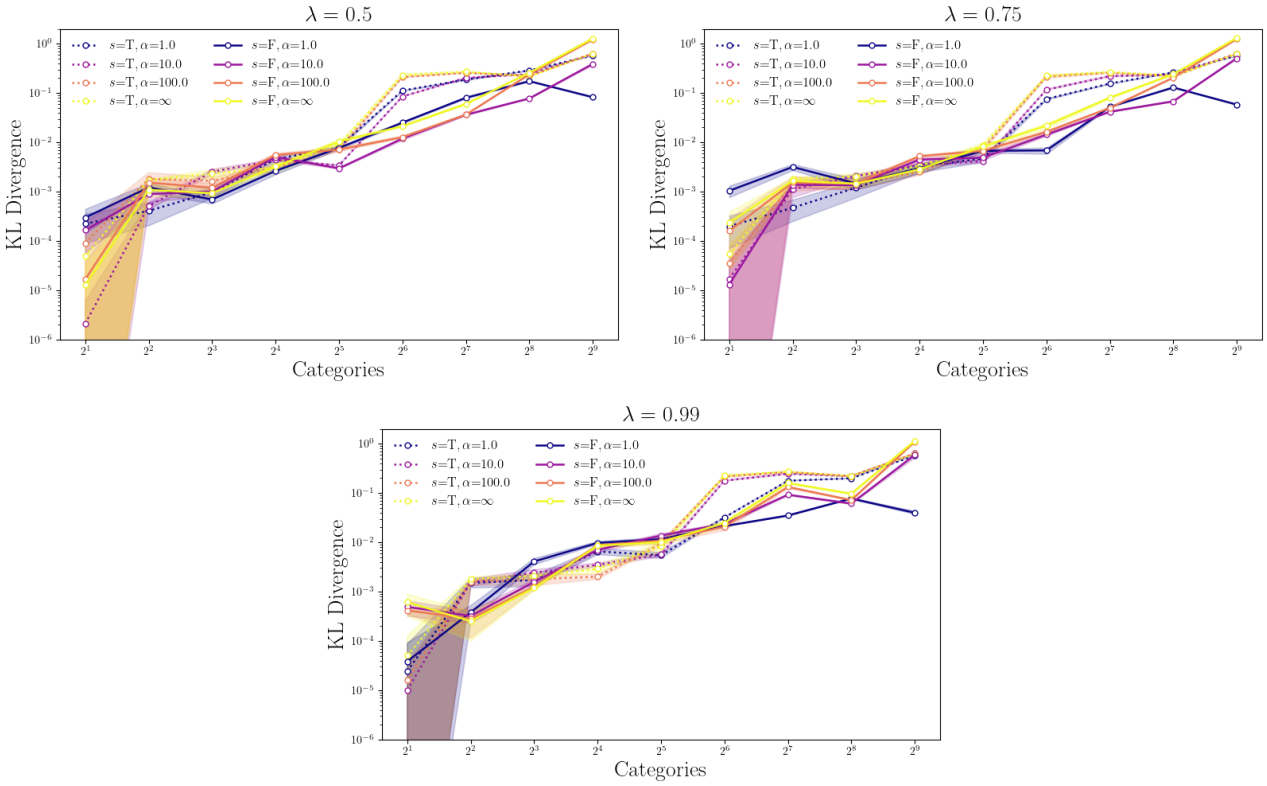


Figure 7: The KL divergence between the true and estimated categorical probabilities for $\alpha \in \{1, 10, 100, \infty\}$, $\lambda \in \{0.5, 0.75, 0.99\}$, with and without scaling. The bands are two standard deviations away from the mean.

ALR and MLR also work relatively well. SB and MLR are highly similar (for BMNIST with 2 classes they are exactly the same map), as they should because the former mostly just stabilizes the computation by centering.

Table 7: Numerical results for ALR and MLR bijections.

Method	Checkerboard	BMNIST	DNA
FM-Å(ALR)	6.4%	4.98	0.029
FM-Å(ALR)w/OT	–	5.22	0.037
FM-Å(MLR)	5.7%	4.93	0.027
FM-Å(MLR)w/OT	–	4.51	0.024
FM-Å(SB)	**5.4%**	4.93	0.028
FM-Å(SB)w/OT	–	4.51	**0.021**
FM-Å(ILR)	6.8%	**4.36**	0.026
FM-Å(ILR)w/OT	–	4.57	0.022