Simplifying Flow Matching Transformations with Low-Rank Mixture Models

Liam A. Kruse, Houjun Liu, Alexandros E. Tzikas, Mansur M. Arief, and Mykel J. Kochenderfer

***Abstract*— Normalizing flows are powerful generative models that learn an invertible mapping between complex data distributions and simple latent distributions, typically a standard normal density. However, this choice of latent density can impose unnecessary complexity on the learned flow transformation due to the topological mismatch between the latent and data densities, leading to slower training and suboptimal performance. In this work, we propose using mixtures of probabilistic principal component analyzers (MPPCA) as the latent density for normalizing flows. We simplify the learned flow transformation by learning a latent distribution that more closely aligns with the data distribution in terms of KL divergence, thus enabling faster convergence and improved generative performance. Critically, MPPCA models can be fit quickly and cheaply using the expectation-maximization algorithm, making them a practical choice for initializing latent distributions even in high-dimensional generative tasks. We validate our method on both tabular and image datasets, demonstrating consistent gains in training efficiency and generation quality compared to baselines.**

I. Introduction

Normalizing flows are a class of generative models that learn a deterministic, invertible mapping between a complex data density and a simpler *base* density, enabling both sampling and density estimation [1]. *Continuous* normalizing flows (CNFs) extend the framework by parameterizing the transformation as the solution to an ordinary differential equation (ODE) [2]. The learned vector field, e.g., the *integration map*, induces a pushforward probability distribution as it transports samples from the base density to the target data density. Early CNF models were trained via maximum likelihood estimation, requiring expensive simulation through the learned ODE dynamics [3]. Recently, the development of the *flow matching objective* allowed flows to be trained in a simulation-free approach by directly regressing to the ODE vector field, akin to the approach used to train diffusion models [4]. Flow matching (FM) has enabled scalable flows to perform high-dimensional tasks such as image generation, image translation, and learning single-cell dynamics [5].

Unfortunately, the learned integration map is not unique, as different dynamics and integration paths can still induce the same pushforward density. Transformations with unnecessarily complex dynamics can increase the computational cost at inference time. Poorly conditioned dynamics require more time steps for an adaptive-step solver to solve the ODE [6] and result in larger time-discretization errors for fixed-step solvers, ultimately degrading the quality of the generated samples [7]. To address this shortcoming of CNF training, inductive biases have been introduced to encourage simpler and more efficient transformations. For example, optimal transport (OT) principles have been incorporated to regularize the learned dynamics [6] and to encourage straighter paths between the data and base densities [5], [8].

In this work, we learn expressive base distributions to simplify the learned flow transformation and provide an inductive bias on the training procedure, as shown in Figure 1. While flow models typically rely on simple standard normal base densities to permit tractable density estimation, we propose mixtures of probabilistic principal component analyzers (MPPCA) [9]—a type of low-rank Gaussian mixture model—as a more flexible alternative. MPPCA models offer two key characteristics: 1) they have an analytical likelihood expression, preserving the computational efficiency required for density estimation, and 2) they can be fit efficiently even on high-dimensional data using the expectation-maximization (EM) algorithm. By initializing the base distribution to closely approximate the target data distribution, our approach reduces the complexity of the learned CNF transformation, leading to faster convergence and improved generative performance. In practice, our approach *warm-starts* the flow training procedure, trading many iterations of gradient-based optimization for a small number of efficient EM steps. Our specific contributions include the following:

- We construct expressive base distributions for flows using MPPCA models and demonstrate that the MPPCA fitting procedure is computationally inexpensive compared to the flow training procedure.
- We show that MPPCA base distributions can be combined with optimal transport-based flow matching objectives to generate higher-quality samples while reducing inference-time integration steps.
- We quantitatively evaluate our approach on challenging density estimation and generative tasks, including training on high-dimensional image datasets.

II. Related Work

Several prior works have investigated the role of the base density on flow performance and training characteristics. Papamakarios et al. [10] compared standard normal base densities against a Masked Autoencoder for Distribution Estimation [11], finding that the standard normal density was

L. A. Kruse (corresponding author), H. Liu, A. E. Tzikas, and M. J. Kochenderfer are with the Stanford Intelligent Systems Laboratory in the Department of Aeronautics and Astronautics at Stanford University, Stanford, CA 94305, USA (email: {lkruse, houjun, alextzik, mykel}@stanford.edu).

M. Arief is with the Industrial and Systems Engineering Department at King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, 31261, KSA (email: mansur.arief@kfupm.edu.sa).

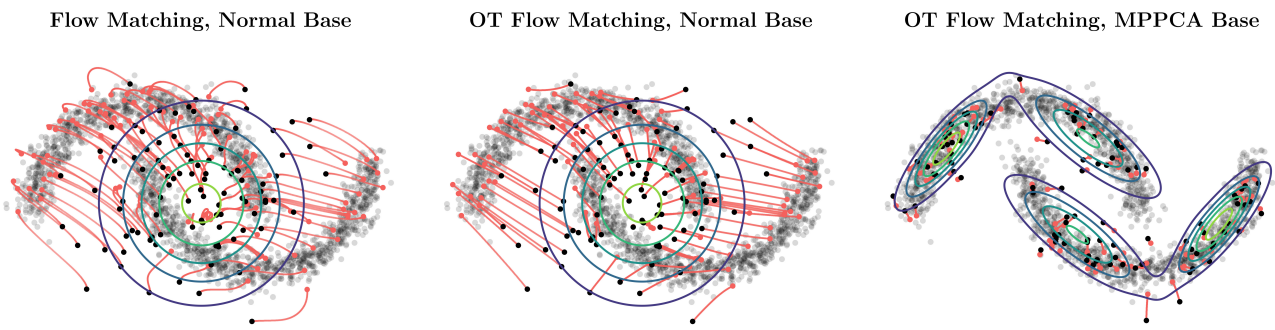


Fig. 1: The integration paths through the flow model are not unique and might be unnecessarily complex (left). Flow matching with OT principles produces straighter paths (center). We propose low-rank base densities to further simplify the learned flow transformation (right).

competitive on most datasets. Researchers have noted that Gaussian mixture model (GMM) base densities can improve classification performance, as data samples from a particular class are mapped to a specific latent mixture component [12], [13]. In this work we are not concerned with classification, but rather unconditional density estimation and sampling. Furthermore, Ardizzone et al. [12] assume that the mixture components have unit covariance matrices, while Izmailov et al. [13] initialize the mixture means randomly and do not fit them along with the flow parameters. We learn base densities that are close to the true data density, and fit the MPPCA models using the EM algorithm prior to training the flow. Hagemann and Neumayer [14] also use GMMs as base densities, and demonstrate that splitting the base density into multiple modes can control the Lipschitz constants of invertible neural networks. They use heuristics to choose the GMM component means, whereas we fit ours directly to the data density. Stimper et al. [15] develop a base distribution for normalizing flows based on learned rejection sampling. Their objective is to better model target distributions with complex topological structure supports. However, their base distribution cannot be directly evaluated since the normalization constant is unknown. In contrast, MPPCA models admit exact likelihood calculations, preserving the density estimation capabilities of the entire flow model.

Intuitively, an expressive base density serves as an inductive bias that simplifes the learned flow transformation by reducing the learning requirements of the flow. Researchers have investigated other strategies to simplify the flow transformation, notably focusing on the connections between optimal transport and the integration paths. Regularizing CNFs with a transport cost can straighten the integration paths and reduce the computational cost of simulation-based maximum likelihood training procedures [6], [16]. More recently, research groups have exploited minibatch approximations of the OT map between two distributions to produce straighter flows, enhancing both training and inference performance [5], [8]. Meanwhile, Liu et al. [7] solve a nonlinear least squares optimization problem to find *straight* couplings—rather than *optimal* couplings—which can be integrated in a single Euler step.

Learning expressive mixture models in high dimensions is a challenging task. High-dimensional covariance estimates risk overfitting to noise, and the matrices can become ill-conditioned or singular [17]. Furthermore, storing full-rank matrices requires memory that scales quadratically with the number of dimensions. One solution is to constrain the learned covariance matrices to be low-rank, effectively modeling the covariances as full-rank matrices on a learned low-dimensional subspace [9]. Two common frameworks for modeling low-rank GMMs include mixtures of probabilistic component analyzers [9] and mixtures of factor analyzers (MFAs) [18]. In this work, we focus on MPPCA models since they can be fit with closed-form expectation-maximization updates. Richardson and Weiss [19] use MFA models for image generation, demonstrating that low-rank mixture models can be trained on full-sized images despite the high dimensionality. Covariance matrix adaptation (CMA) is an evolutionary optimization algorithm that iteratively adapts a covariance matrix to improve sample efficiency [20]. Low-rank updates make the optimization process robust and sample efficient [21].

III. Continuous Normalizing Flows

We outline the fundamental theory behind continuous normalizing flows, the flow matching objective, and flow-based generative modeling.

A. Continuous Normalizing Flows

A smooth, time-varying vector field $u_t : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ defines an ordinary differential equation (ODE):

$$dx = u_t(x)dt \tag{1}$$

We denote the ODE solution as $\phi_t(x)$, which is a diffeomorphic map that transports a point x along the vector field from time 0 up to time t and has initial conditions $\phi_0(x) = x$. Given a probability distribution $p_0(x)$, the map $\phi_t(x)$ induces a *pushforward* density

$$p_t(x) = p_0(\phi_t^{-1}(x)) \left| \det \left[\frac{\partial \phi_t^{-1}}{\partial x}(x) \right] \right| \tag{2}$$

This time-varying density is characterized by the *continuity equation* [5]

$$\frac{\partial p}{\partial t} = -\nabla \cdot (p_t u_t) \tag{3}$$

Chen et al. [2] modeled the vector field u_t with a neural network, creating a deep parametric generative model called a continuous normalizing flow (CNF). Samples from $p_t(x)$ can be obtained by drawing samples $x \sim p_0(x)$ and transporting them along the vector field u. Furthermore, the density $p_t(x)$ can be evaluated using the change-of-variables formula from Equation (2). Previously, normalizing flow generative models were created by stacking discrete sets of diffeomorphic transformations [1]. Often, the transformations had heavily restricted architectures to ensure invertibility or tractable Jacobian calculations [22]. However, CNFs permit unrestricted neural network architectures [3].

B. Flow Matching

Early CNF models were trained by maximum likelihood—a computationally expensive procedure that requires simulating the ODE dynamics at every training step to obtain the instantaneous changes in log density [3]. The *flow matching objective* offers a simulation-free approach to training CNFs by regressing to u directly, in a similar fashion to regressing the drift for stochastic differential equations in diffusion models [23]. Consider a mixture of probability paths $p_t(x \mid z)$ that vary according to a conditional variable z. We can rewrite the marginal probability path $p_t(x)$ as

$$p_t(x) = \int p_t(x \mid z)\, q(z) dz \tag{4}$$

where $q(z)$ is a distribution over the conditioning variable. If the vector field $u_t(x \mid z)$ generates the path $p_t(x \mid z)$ from initial conditions $p_0(x \mid z)$, then the vector field

$$u_t(x) = \mathbb{E}_{q(z)} \frac{u_t(x \mid z)\, p_t(x \mid z)}{p_t(x)} \tag{5}$$

generates the path $p_t(x)$, provided that p_t and u_t satisfy Equation (3). When the conditional probability paths $p_t(x \mid z)$ and vector fields $u_t(x \mid z)$ are known and have simple forms, we can obtain an unbiased stochastic objective for regressing the vector field in Equation (5) [4], [5], [23], [24]. Let $v_\theta(t, x) : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ denote a time-varying vector field parameterized by a neural network with weights θ. The *conditional* flow matching objective

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,\ q(z),\ p_t(x|z)} \|v_\theta(t, x) - u_t(x \mid z)\|^2 \tag{6}$$

allows us to regress the marginal vector field from Equation (5) using only samples from $p_t(x \mid z)$ and $u_t(x \mid z)$. Please refer to Tong et al. [5] for a discussion on common forms for the conditional probability paths and vector fields.

IV. MPPCA Base Distributions

The initial probability distribution p_0 introduced in Section III-A is typically a standard normal density, which allows for tractable density estimation using Equation (2) [1], [4], [16]. Tong et al. [5] showed how to conduct flow matching with samples from an arbitrary p_0; however, density estimation is no longer feasible if p_0 does not have an analytical likelihood expression. In this work, we propose representing p_0 with MPPCA models, which we introduce in the following section.

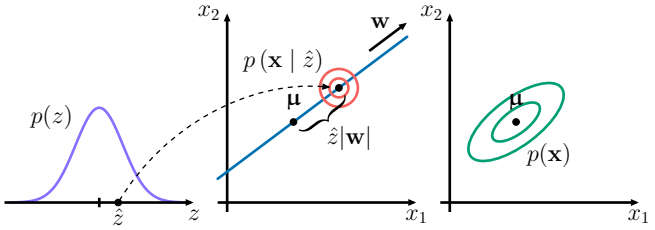


Fig. 2: In the probabilistic PCA framework, samples from a low-dimensional latent space are projected to the data space by the factor loading matrix, which defines the principal component directions.

A. Low-Rank Mixture Models

Principal component analysis (PCA) is a classic statistical technique for dimensionality reduction. Tipping and Bishop [9] reformulate PCA within a maximum likelihood framework, resulting in an associated probability density. Consider the following latent variable model:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{7}$$

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{8}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \tag{9}$$

where $\mathbf{W}$ is a rectangular *factor loading* matrix of size $d \times \ell$ and ℓ is a latent dimension such that $\ell \ll d$. The latent vector $\mathbf{z}$ is of length ℓ, the mean vector $\boldsymbol{\mu}$ is of length d, and $\boldsymbol{\epsilon}$ is added noise with diagonal covariance $\sigma^2\mathbf{I}$. For MPPCA, the noise is assumed to be isotropic; in the more general MFA framework, the noise assumption is relaxed to merely be diagonal. In the case of isotropic noise, the implied conditional distribution is

$$p(\mathbf{x} \mid \mathbf{z}) = \left(2\pi\sigma^2\right)^{-d/2} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{W}\mathbf{z} - \boldsymbol{\mu}\|^2\right\} \tag{10}$$

The Gaussian prior over the latent variables is given by

$$p(\mathbf{z}) = (2\pi)^{-\ell/2} \exp\left\{-\frac{1}{2}\mathbf{z}^\top \mathbf{z}\right\} \tag{11}$$

Thus, the marginal density over $\mathbf{x}$ can be expressed as

$$p(\mathbf{x}) = \eta |\mathbf{C}|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{C}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right\} \tag{12}$$

with normalizing constant $\eta = (2\pi)^{-d/2}$ and model covariance $\mathbf{C} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}$. The likelihood of a dataset $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_N\}$ is maximized when the columns of the factor loading matrix $\mathbf{W}$ span the principal subspace of the data [9]. Equation (12) defines the marginal likelihood for a single probabilistic principal component analyzer (PPCA); multiple PPCA models can be combined in a mixture to learn local principal axes and model more complex distributions

[9]. Figure 2 shows how a low-dimensional latent variable is projected to the data space for a single PPCA model, defining a Gaussian marginal distribution $p(\mathbf{x})$.

B. Fitting MPPCA Models

Consider a mixture of K probabilistic principal component analyzers as presented in Equation (7). The log-likelihood of a dataset $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_N\}$ is

$$\mathcal{L} = \sum_{i=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k p(\mathbf{x}_n \mid k) \right\} \tag{13}$$

where $p(\mathbf{x} \mid k)$ represents the density of the kth PPCA model and π_k is the corresponding mixing proportion, with $\pi_k \geq 0$ and $\sum_{k=1}^{K} \pi_k = 1$. Tipping and Bishop [9] derive an iterative expectation-maximization procedure with closed-form updates for parameters $\mathbf{W}_k$, μ_k, π_k, and σ_k^2 for components $k = 1, \ldots, K$ that is guaranteed to find a local maximum of Equation (13). The EM procedure can also be performed in a memory-efficient fashion for large or high-dimensional datasets by accumulating sufficient statistics over minibatches [19].

V. Experiments

This section first presents our datasets and evaluation metrics before discussing experimental results.

A. Datasets

We perform density estimation on three tabular datasets from the UC Irvine Machine Learning Repository, following the preprocessing steps of Papamakarios et al. [10]. These datasets are widely used in flow-based generative modeling benchmarks [3], [16], [25]. Furthermore, we train flows on three datasets of natural images: FashionMNIST [26], CelebA [27], and CIFAR-10 [28]. We crop, normalize, and resize the CelebA dataset to both 32×32 and 64×64.

B. Metrics

We compute the following metrics to evaluate the quality of the learned generative models and quantify the impact of learning an MPPCA base distribution:

- *Average log-likelihood*: A held-out test dataset is transformed by integrating backwards through the learned map ϕ and the log-likelihood of each data point is computed according to Equation (2). A *higher* value indicates that the learned flow density more closely matches the samples.
- *Number of statistically different bins (NDB)*: The NDB metric is a simple method to evaluate generative models based on relative proportions of samples that fall into predetermined bins [19]. A *lower* value indicates that the learned distribution more closely represents the data distribution. We divide by the number of bins C and report the *proportion* of statistically different bins.
- *Fréchet inception distance metric (FID)*: FID is a standard metric for evaluating the quality of generated images commonly found in generative adversarial network literature [29]. In particular, we use the embeddings generated by the third layer of a pretrained InceptionV3 model, as is standard, to compute the FID metric. A *lower* value indicates that the learned distribution more closely represents the data distribution.
- *Number of function evaluations (NFE)*: We report the average number of function evaluations required by an adaptive-step ODE solver to produce the integration map ϕ at a prespecified numerical tolerance. A *higher* NFE value indicates that the learned ODE dynamics are more complex, resulting in more computational burden at inference time due to the increased number of solver steps [4], [6].

TABLE I: Hyperparameters

	K	ℓ	EM iter	α_0	batch
Hepmass	10	4	10	5×10^{-4}	256
Miniboone	30	6	50	5×10^{-4}	64
Bsds300	30	6	10	5×10^{-4}	256
Fashion	50	6	10	2×10^{-4}	128
Cifar-10	50	10	10	2×10^{-4}	128
Celeba (32×32)	50	10	10	2×10^{-4}	128
Celeba (64×64)	64	16	16	2×10^{-4}	128

We report the log-likelihood metric for the UCI datasets and the NDB metric for the image datasets. In all experiments, we report the average required NFE as well as the total training time. The total training time includes the time required to fit the MPPCA base distributions.

C. Experimental Setup

We parameterize the vector field u_t with a simple multilayer perception (MLP) neural network for the tabular density estimation tasks and with a U-net [30], [31] for the image generation tasks. The U-net weights are updated with an exponential moving average to smooth out noisy parameter updates. We formulate p_0 as an MPPCA model for our proposed technique and benchmark against a standard choice of $p_0 = \mathcal{N}(\mathbf{0}, \mathbf{I})$. Flows are trained via stochastic gradient descent using the variance-preserving (VP) flow matching objective [23]. Furthermore, to quantify the simplifying effect of the MPPCA base, we also ablate the flow matching objective by considering a minibatch optimal transport objective [5]. The minibatch approximations to the OT map have been shown to create simpler flows that are more stable to train. For experiments where p_0 is an MPPCA model, we first fit the model on the training dataset before training the flow separately. Hyperparameters are listed in Table I, where α_0 is the initial learning rate. MPPCA hyperparameter values are similar to the values used by Richardson and Weiss [19].

D. Results

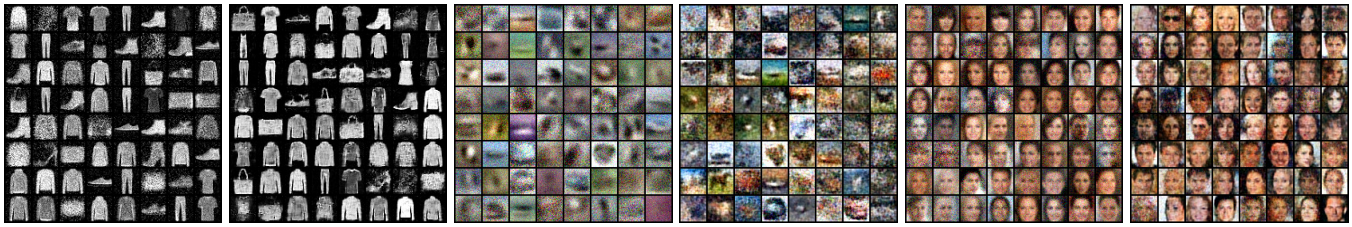
We implement early stopping for the density estimation tasks, stopping training if there is no improvement in validation loss for 10 epochs. For the image generation tasks, we train for a fixed number of epochs across all trials,

TABLE II: Density estimation results on tabular UCI datasets.

Dataset	Model	Training		Testing	
		epochs (↓)	time (min) (↓)	log-likelihood (↑)	NFE (↓)
HEPMASS $d = 21$	VP-MPPCA	**34.6 ± 5.9**	**1.7 ± 0.3**	−27.00 ± 0.03	**50.0 ± 0.2**
	OT-MPPCA	48.6 ± 7.2	8.5 ± 1.2	**−26.93 ± 0.01**	52.5 ± 1.4
	VP-Normal	40.8 ± 5.7	1.8 ± 0.2	−27.95 ± 0.01	61.2 ± 0.6
	OT-Normal	52.6 ± 6.2	9.1 ± 1.0	−27.91 ± 0.01	63.2 ± 0.7
MINIBOONE $d = 43$	VP-MPPCA	24.4 ± 7.1	**0.3 ± 0.1**	**−30.97 ± 0.46**	**43.1 ± 1.0**
	OT-MPPCA	**21.0 ± 7.2**	0.4 ± 0.1	−31.42 ± 0.45	43.5 ± 0.7
	VP-Normal	53.6 ± 14.2	0.5 ± 0.1	−50.37 ± 0.99	57.0 ± 1.0
	OT-Normal	82.2 ± 12.0	1.1 ± 0.2	−48.40 ± 0.55	62.4 ± 1.4
BSDS300 $d = 63$	VP-MPPCA	**31.6 ± 7.1**	**10.6 ± 1.0**	126.64 ± 1.68	50.9 ± 1.1
	OT-MPPCA	32.8 ± 5.6	25.0 ± 3.2	**127.25 ± 1.11**	**50.7 ± 0.6**
	VP-Normal	140.6 ± 21.0	17.2 ± 2.7	69.76 ± 1.67	102.8 ± 0.5
	OT-Normal	171.0 ± 15.6	102.4 ± 9.3	86.14 ± 1.39	113.8 ± 0.5

TABLE III: Generative modeling results on image datasets.

Dataset	Model	Training		Testing		
		epochs	time (min)	NDB/C (↓)	NFE (↓)	FID (↓)
FASHION $[28 \times 28 \times 1]$	VP-MPPCA	100	172.3 ± 6.0	0.52 ± 0.03	32.8 ± 0.3	131.7 ± 2.2
	OT-MPPCA	100	187.5 ± 7.9	**0.37 ± 0.02**	**26.0 ± 0.0**	**92.5 ± 4.9**
	VP-Normal	100	299.8 ± 1.9	0.92 ± 0.01	38.2 ± 0.2	133.3 ± 3.3
	OT-Normal	100	183.9 ± 7.6	0.89 ± 0.01	38.0 ± 0.0	136.2 ± 7.6
CIFAR-10 $[32 \times 32 \times 3]$	VP-MPPCA	100	224.9 ± 2.2	0.33 ± 0.00	**26.0 ± 0.0**	289.9 ± 2.5
	OT-MPPCA	100	172.2 ± 1.4	**0.30 ± 0.02**	**26.0 ± 0.0**	**268.4 ± 2.4**
	VP-Normal	100	246.5 ± 1.4	0.88 ± 0.00	38.0 ± 0.0	303.2 ± 8.5
	OT-Normal	100	199.1 ± 2.6	0.84 ± 0.00	32.0 ± 0.0	305.9 ± 3.4
CELEBA $[32 \times 32 \times 3]$	VP-MPPCA	50	319.6 ± 1.3	0.44 ± 0.03	26.1 ± 0.1	304.7 ± 1.2
	OT-MPPCA	50	501.5 ± 1.8	**0.32 ± 0.02**	**26.0 ± 0.0**	**237.1 ± 3.8**
	VP-Normal	50	327.2 ± 1.5	0.83 ± 0.01	50.2 ± 4.6	289.9 ± 2.4
	OT-Normal	50	495.0 ± 1.8	0.86 ± 0.01	37.9 ± 0.1	290.9 ± 2.3
CELEBA $[64 \times 64 \times 3]$	VP-MPPCA	50	451.3 ± 0.0	0.68 ± 0.00	26.7 ± 0.0	285.3 ± 0.0
	OT-MPPCA	50	351.3 ± 0.0	**0.38 ± 0.00**	**26.0 ± 0.0**	**239.6 ± 0.0**
	VP-Normal	50	540.6 ± 2.4	0.89 ± 0.02	56.0 ± 0.0	387.9 ± 7.5
	OT-Normal	50	407.6 ± 1.1	0.89 ± 0.01	44.0 ± 0.0	377.5 ± 4.2



(a) Samples from OT-MPPCA.

(b) Samples from OT-Normal.

Fig. 3: For each image dataset, sample pairs show samples from the learned model after the first (left) and last (right) training epoch.

hypothesizing that the warm-start provided by the MPPCA base density will lead to improved performance given a constrained training budget. We sweep across five random seeds for the density estimation experiments and three random seeds for the image generation experiments. Code to reproduce the experimental results is available at `https://github.com/sisl/LowRankLatentGMMs`.

Table II shows the results for density estimation on the UCI datasets. Early stopping generally triggers earliest when training flows with an MPPCA base, as the average number of required training epochs decreases dramatically for VP-MPPCA and OT-MPPCA models on the MINIBOONE and BSDS300 datasets. The computational savings obtained from early stopping outweigh the additional overhead cost of fitting the more expressive base distribution, as evidenced by the relatively short overall training times for models with an MPPCA base. Flows with MPPCA base densities obtain the highest average log-likelihood values on all three datasets. Interestingly, VP-MPPCA achieves the lowest average NFE on two of the three datasets, implying that the simplifying effect of learning an expressive base distribution does not necessarily benefit from the additional application of optimal-transport-based inductive biases.

Table III shows the results for the image generation experiments. Once again, flows with MPPCA base densities require the fewest average NFEs at inference time, implying that VP-MPPCA and OT-MPPCA learn simpler integration paths. Furthermore, the OT-MPPCA architecture achieves the lowest NDB/C and FID scores across all image datasets given a fixed number of training epochs, which indicates that learning an expressive base density helps the model generate samples that are a closer match to the ground-truth images. Learning the MPPCA base density warm-starts the training procedure—a phenomenon clearly visible in Figure 3. Each row shows image pairs of generated samples from the FashionMNIST, CIFAR-10, and CelebA datasets. The leftmost image in each pair shows samples generated after the first epoch, while the rightmost image shows samples generated after the final epoch. By initializing the normalizing flow with an MPPCA base density, the model starts with a structured prior that closely approximates the data distribution. This warm-start significantly reduces the burden on the flow to learn complex transformations, allowing it to refine rather than entirely reshape the base distribution. As a result, even after a single training epoch, generated samples already exhibit meaningful resemblance to ground truth images.

VI. MPPCA Considerations

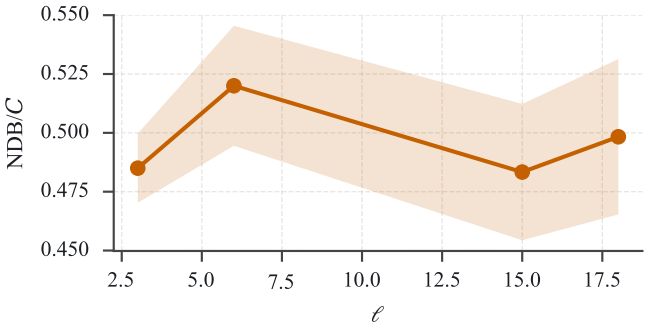
We next examine computational considerations and the importance of the latent factors hyperparameter.

A. Computational Overhead

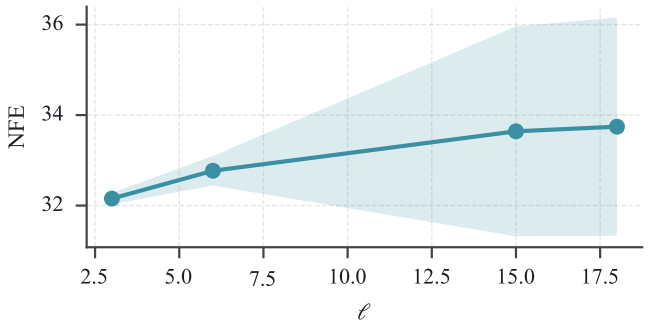
Learning an MPPCA base introduces additional computational overhead due to the need to fit the mixture model separately before flow training. Unlike a standard normal base, MPPCA requires learning cluster means, factor loading matrices, noise variances, and mixing coefficients. The total number of parameters for an MPPCA model with K components is $K(d\ell+d+1)+(K-1)$. The time required to fit the MPPCA model is minimal compared to the overall training process, accounting for less than 3% of the total training time across all image generation experiments, as shown in Table IV.

TABLE IV: Time required to fit MPPCA via EM versus total training times.

Dataset	Model	EM time (min)	Total (min)
Fashion	VP-MPPCA	0.3 ± 0.0	93.5 ± 0.1
	OT-MPPCA	0.3 ± 0.0	94.3 ± 0.3
Cifar-10	VP-MPPCA	0.7 ± 0.0	85.3 ± 0.0
	OT-MPPCA	0.7 ± 0.0	86.3 ± 0.0
Celeba $[32 \times 32 \times 3]$	VP-MPPCA	4.7 ± 0.1	192.3 ± 0.1
	OT-MPPCA	4.7 ± 0.0	194.6 ± 0.5



(a) NDB/C versus number of latent factors



(b) NFEs versus number of latent factors

Fig. 4: Performance metrics on the FashionMNIST dataset as a function of latent factors.

B. The Effect of Latent Factors on Model Performance

We consider the FashionMNIST dataset and sweep over a range of latent factor counts for the latent MPPCA distribution, recording the average NDB/C and NFE scores across test batches for three random seeds. The ranges of performance metrics are visualized in Figure 4. Overall, performance is relatively steady across the sweep of latent factors, indicating that MPPCA models with even a small number of latent factors can provide a meaningful inductive

bias for the learning process. Richardson and Weiss [19] showed that MPPCA models with 10 or fewer latent factors can adequately approximate high-dimensional distributions, which is consistent with Figure 4.

VII. Discussion

We present a technique to simplify flow matching transformations using low-rank mixture base distributions. Mixtures of probabilistic principal component analyzers are a parameter-efficient model that can be updated analytically via expectation-maximization, even in high-dimensional spaces. Initializing the flow matching with an MPPCA base density serves as an inductive bias on the training process, reducing the burden on the flow to learn complex transformations. We show that MPPCA base distributions can be combined with continuous normalizing flows to generate higher-quality samples and reduce the number of required integration steps at inference time compared to standard normal base densities.

Future work will explore techniques to scale the MPPCA fitting procedure to even higher dimensions, such as feature sampling or initializing mixture component clusters with constrained K-means. We will also evaluate the simplifying effect of MPPCA base densities when combined with other flow matching objectives such as action matching [24] and Schrödinger Bridge flow matching [5].

Acknowledgments

Toyota Research Institute (TRI) provided funds to assist the authors with their research, but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

References

[1] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International Conference on Machine Learning (ICML)*, 2015, pp. 1530–1538.

[2] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[3] W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "FFJORD: Free-form continuous dynamics for scalable reversible generative models," in *International Conference on Learning Representations (ICLR)*, 2018.

[4] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *International Conference on Learning Representations (ICLR)*, 2023.

[5] A. Tong et al., "Improving and generalizing flow-based generative models with minibatch optimal transport," *Transactions on Machine Learning Research*, 2024.

[6] C. Finlay, J.-H. Jacobsen, L. Nurbekyan, and A. Oberman, "How to train your neural ode: The world of Jacobian and kinetic regularization," in *International Conference on Machine Learning (ICML)*, 2020, pp. 3154–3164.

[7] X. Liu, C. Gong, et al., "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *International Conference on Learning Representations (ICLR)*, 2023.

[8] A.-A. Pooladian, H. Ben-Hamu, C. Domingo-Enrich, B. Amos, Y. Lipman, and R. T. Chen, "Multisample flow matching: Straightening flows with minibatch couplings," in *International Conference on Machine Learning (ICML)*, 2023, pp. 28 100–28 127.

[9] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, 1999.

[10] G. Papamakarios, T. Pavlakou, and I. Murray, "Masked autoregressive flow for density estimation," *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, 2017.

[11] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *International Conference on Machine Learning (ICML)*, 2015, pp. 881–889.

[12] L. Ardizzone, R. Mackowiak, C. Rother, and U. Köthe, "Training normalizing flows with the information bottleneck for competitive generative classification," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 7828–7840, 2020.

[13] P. Izmailov, P. Kirichenko, M. Finzi, and A. G. Wilson, "Semi-supervised learning with normalizing flows," in *International Conference on Machine Learning (ICML)*, 2020, pp. 4615–4630.

[14] P. Hagemann and S. Neumayer, "Stabilizing invertible neural networks using mixture models," *Inverse Problems*, vol. 37, no. 8, 2021.

[15] V. Stimper, B. Schölkopf, and J. M. Hernández-Lobato, "Resampling base distributions of normalizing flows," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022, pp. 4915–4936.

[16] D. Onken, S. W. Fung, X. Li, and L. Ruthotto, "OT-Flow: Fast and accurate continuous normalizing flows via optimal transport," in *AAAI Conference on Artificial Intelligence (AAAI)*, vol. 35, 2021, pp. 9223–9232.

[17] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365–411, 2004.

[18] Z. Ghahramani, G. E. Hinton, et al., "The EM algorithm for mixtures of factor analyzers," University of Toronto, Tech. Rep. CRG-TR-96-1, 1996.

[19] E. Richardson and Y. Weiss, "On GANs and GMMs," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[20] N. Hansen, "The CMA evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.

[21] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for Optimization*. MIT Press, 2019.

[22] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.

[23] M. S. Albergo, N. M. Boffi, and E. Vanden-Eijnden, "Stochastic interpolants: A unifying framework for flows and diffusions," *arXiv preprint arXiv:2303.08797*, 2023.

[24] K. Neklyudov, R. Brekelmans, D. Severo, and A. Makhzani, "Action matching: Learning stochastic dynamics from samples," in *International Conference on Machine Learning (ICML)*, 2023, pp. 25 858–25 889.

[25] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[26] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[27] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *International Conference on Computer Vision (ICCV)*, 2015, pp. 3730–3738.

[28] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[29] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANstrained by a two time-scale update rule converge to a local Nash equilibrium," *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, 2017.

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer, 2015, pp. 234–241.

[31] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning (ICML)*, 2021, pp. 8162–8171.