Flows for simultaneous manifold learning and density estimation (by Johann Brehmer and Kyle Cranmer)

$\mathcal{M}$-Flows

Jona Braun

ETHZ

March 2021

Outline

Introduction

- Generative modeling: Infer a probability distribution over a random variable X given observations $\{x^{(i)}\}_{i=1}^{N}$
  - Generative adversarial networks (GANs)
    - intractable density
    - lower dimensional latent space
  - Variational autoencoders (VAEs)
    - approx. tractable density
    - lower dimensional latent space
  - Normalizing Flows
    - tractable density
    - no lower dimensional latent space
- $\mathcal{M}$anifold-Flow is a combination of these methods
  1. Learns a lower dimensional data space
  2. Learns a tractable probability density on the data manifold
  - Claim: allows for dimensionality reduction, denoising and out-of-distribution detection

Normalizing Flow I

- A normalizing flow describes the transformation of a probability density through a sequence of invertible mappings[1]

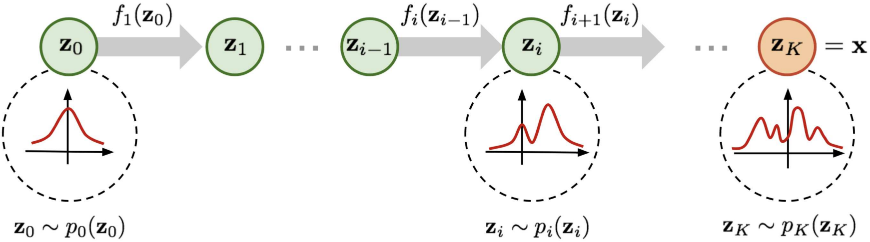


Figure: Normalizing flow[2]

Normalizing Flow II

- Let f be an invertible smooth function $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ with smooth inverse f^{-1}. Further let z_0 be a random variable $z_0 \sim p(z_0)$. Using the change-of-variable formula the random variable $z_1 = f(z_0)$ has distribution:

$$p_1(z_1) = p_0(f^{-1}(z_1)) \left|\det \frac{\partial f^{-1}}{\partial z_1}\right| = p_0(z_0) \left|\det \frac{\partial f}{\partial z_0}\right|^{-1}$$

- Through successive application of such transformations we can obtain a complex probability density

$$z_K = f_K \circ ... \circ f_2 \circ f_1(z_0)$$

$$\ln p_K(z_K) = \ln p_0(z_0) - \sum_{k=1}^{K} \ln \left|\det \frac{\partial f_k}{z_{k-1}}\right|$$

[1] Variational Inference with Noramlizing Flows, Danilo Jimenez Rezende Shakir Mohamed

[2] https://lilianweng.github.io/lil-log/2018/10/13/flow-based-deep-generative-models.html

Manifold[3]

- An n-manifold is a topological space that locally resembles an n-dimensional Euclidean space near each point
- Coordinate chart / map: An invertible map between a subset of the manifold and a simpler space (homeomorphism)
- Atlas: Collection of charts which cover a manifold
- Transition map: Mapping between overlapping charts
- Diffrentiable manifold
  1. Charts need to be diffeomorphisms
  2. Transition maps need to be diffeomorphisms
- Given two Manifolds M, N. A diffeomorphism is a smooth bijective function $f : M \mapsto N$ of which the inverse f^{-1} is also smooth.

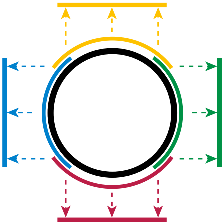


Figure: Circle with charts

[3] https://en.wikipedia.org/wiki/Manifold

Section 2: Contribution

Comparison of different approaches I

- General setting: $x \in \mathcal{M}^* \subset X = \mathbb{R}^d$ with $x \sim p^*(x)$, where
  - x is a sample
  - X is the d-dimensional data space
  - $\mathcal{M}^*$ is a n-dimensional manifold
- Given training samples $\{x_i\} \sim p^*(x)$ the goal is to estimate the
  - Density $p^*(x)$
  - Manifold $\mathcal{M}^*$
- Let $u \in U$ and $v \in V$ be latent variables, where
  - $U = \mathbb{R}^n$ is the latent space that maps to the manifold $\mathcal{M}$ (coordinates of the manifold)
  - $V = \mathbb{R}^{d-n}$ represents the remaining latent variables (direction "*off* the manifold")

Comparison of different approaches II

- Assumptions for simplification
  1. The dimension of the manifold is known and it is n
  2. The manifold is topologically equal to $\mathbb{R}^n$ $\rightarrow$ can be mapped by a single chart and it is connected

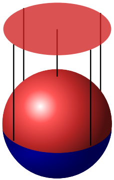


Figure: Sphere 2-Manifold

Different flow models I

$X = \mathbb{R}^d$	data space
$U = \mathbb{R}^n$	coordinates of the manifold
$V = \mathbb{R}^{d-n}$	direction "*off*" the manifold

- Normalizing flow (Ambient Flow)

$$\text{AF:} \quad x \xleftarrow{\;f\;} (u, v) \sim p_{uv}$$

- Flow on a prescribed Manifold (FOM)

$$\text{FOM:} \quad x \xleftarrow{\;g^*\;} u \xleftarrow{\;h\;} \tilde{u} \sim p_{\tilde{u}}$$

- Pseudo-invertible encoder (PIE)

$$\text{PIE:} \quad x \xleftarrow{\;f\;} (u, v) \xrightarrow{\text{Split}} \begin{cases} u \xleftarrow{\;h\;} \tilde{u} \sim p_{\tilde{u}} \\ v \sim p_v \end{cases}$$

Different flow models II

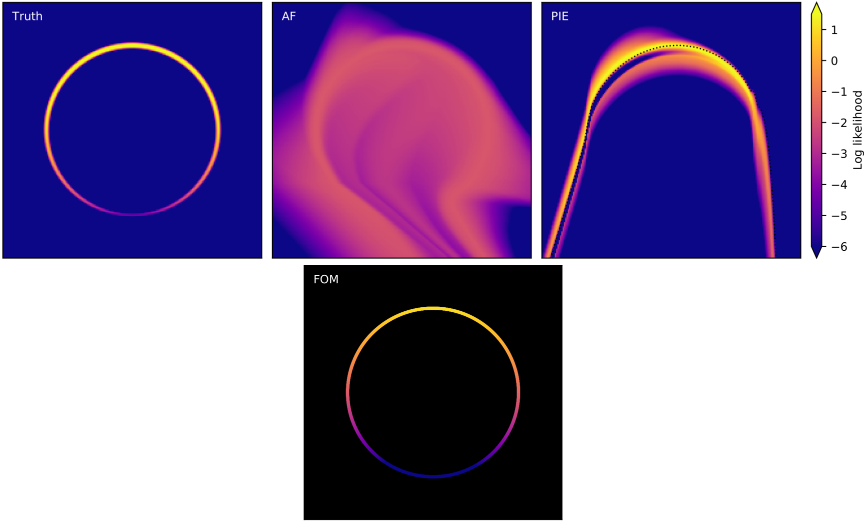


Figure: Gaussian on circle

$\mathcal{M}$-Flow I

$X = \mathbb{R}^d$		data space
$U = \mathbb{R}^n$		coordinates of the manifold
$V = \mathbb{R}^{d-n}$		direction "*off*" the manifold

$$\begin{tikzcd}
\mathcal{M}\text{-flow:} & x & (u,v) \arrow[l, "f"'] \arrow[r, dashed, shift left, "\mathrm{Proj}"] & u \arrow[l, dashed, shift left, "\mathrm{Pad}"] \arrow[ll, bend left, dashed, "g = f \circ \mathrm{Pad}"] & \tilde{u} \sim p_{\tilde{u}} \arrow[l, "h"']
\end{tikzcd}$$

- Given base density $p_{\tilde{u}}(\tilde{u})$
- Learn a n-dimensional flow h, that maps u to $\tilde{u}$

$$h : \tilde{U} \mapsto U, \qquad \tilde{U}, U = \mathbb{R}^n$$

$$p_u(u) = p_{\tilde{u}}(h^{-1}(u)) \left|\det J_h(h^{-1}(u))\right|^{-1}$$

$\mathcal{M}$-Flow II

$$
\begin{tikzcd}
\mathcal{M}\text{-flow:}\quad x & (u,v) \arrow[l, "f"'] \arrow[r, dashed, shift left, "\mathrm{Proj}"] & u \arrow[l, dashed, shift left, "\mathrm{Pad}"] \arrow[ll, dashed, bend left, "g = f \circ \mathrm{Pad}"] & \tilde{u} \sim p_{\tilde{u}} \arrow[l, "h"']
\end{tikzcd}
$$

- Learn diffeomorphism $f : U \times V \mapsto X = \mathbb{R}^d$ between latent space and data space
- Let the model manifold $\mathcal{M}$ be defined by the level set

$$
\begin{aligned}
g : U \mapsto M \subset X \text{ with } u \rightarrow g(u) &= f(u, 0) \\
g &= f_k \circ \ldots \circ f_1 \circ \mathrm{Pad}
\end{aligned}
$$

where Pad represents a n-dimensional vector with zero padding

$$
\mathrm{Pad}(u) = (u_0, \ldots, u_{n-1}, 0, \ldots, 0)^T
$$

$\mathcal{M}$-Flow III

- Than the probability density on the manifold is

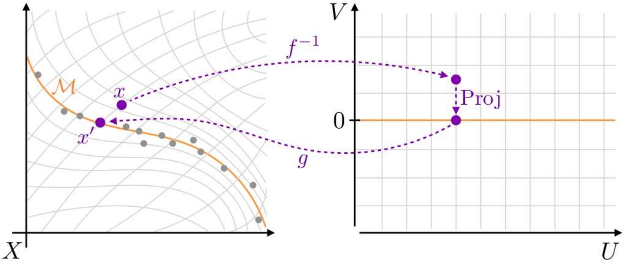
$$p_{\mathcal{M}}(x) = p_{\tilde{u}}(h^{-1}(g^{-1}(x))) \left|\det J_h(h^{-1}(g^{-1}(x)))\right|^{-1} \times \left|\det \left[J_g^T(g^{-1}(x)) J_g(g^{-1}(x))\right]\right|^{-\frac{1}{2}}$$

  where J_g is a $d \times n$ dimensional Jacobian matrix

- The density is equal to the FOM models, except the transformation g is learnable rather than given

- Sampling from an $\mathcal{M}$-flow: Draw $\tilde{u} \sim p_{\tilde{u}}(\tilde{u})$, $u = h(\tilde{u})$ and compute $x = g(u) = f(u, 0)$

$\mathcal{M}$-Flow IV



- g is a decoder: maps from a lower dimensional space to the data space
- Let g^{-1} be the encoder defined as $g^{-1} : X \mapsto U$

$$g^{-1}(x) = \text{Proj}(f^{-1}(x)), \text{ with } \text{Proj}(u, v) = u$$

Like that the inverse of g is extended to the entire data space X and not only defined for $x \in \mathcal{M}$.

$\mathcal{M}$-Flow V

- Encoder and decoder are exact inverses of each other for points on the manifold!
- Reconstruction error

$$||x - x'|| = ||x - g(g^{-1}(x))||$$

- $\mathcal{M}$-flows allow for
  - Computation of the likelihood on the manifold $p_{\mathcal{M}}(x')$
  - Denosing of the input: by using the projection $x' = g(g^{-1}(x))$
  - Anomaly or out-of-distribution detection by using the reconstruction error $||x - x'||$

$\mathcal{M}$-Flow VI

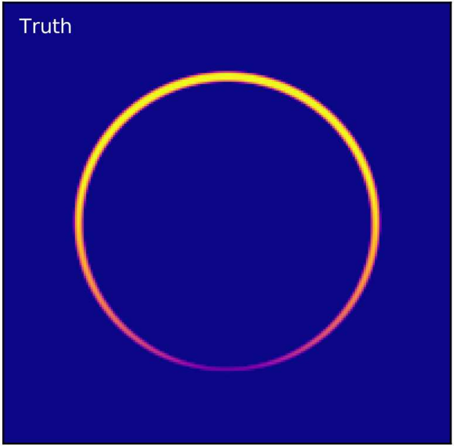


Figure: Gaussian on a circle - Truth

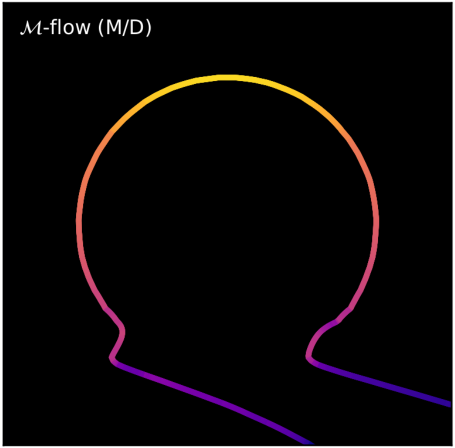


Figure: $\mathcal{M}$-Flow

Manifold-learning flows with separate encoder ($\mathcal{M}_e$-flow)

$$\mathcal{M}_e\text{-flow:}\quad \begin{tikzcd} x \arrow[rr, bend left, dotted, "e"] & (u,v) \arrow[l, "f"'] & u \arrow[l, dashed, "\mathrm{Pad}"] \arrow[ll, bend left, dashed, "g = f \circ \mathrm{Pad}"] & \tilde{u} \sim p_{\tilde{u}} \arrow[l, "h"'] \end{tikzcd}$$

- Is a variant of $\mathcal{M}$-flows
- We replace the encoder g^{-1} with a separate function

$$e : X \mapsto U$$

- The encoder is not restricted to be invertible this increases the expressiveness
- Encoder and decoder may be inconsistent, which during training has to be penalized (similar to VAEs)

Probabilistic autoencoder (PAE)

$$\text{PAE:} \quad x \underset{g}{\overset{e}{\rightleftarrows}} u \xleftarrow{h} \tilde{u} \sim p_{\tilde{u}}$$

- Like the M_e-flow but we replace the invertible transformation g with a decoder

$$e : U \mapsto X$$

  that does not need to be invertible
- This is an autoencoder where the latent space is modeled with a flow
- Pro: decoder gains expressivity
- Con: the density of the model is intractable

Relaxing simplifications

- Learn the dimensionality
  - Brute force search for the right dimension n
  - Evaluate different n with the reconstruction error
- Manifold with multiple disjiont pieces
  - Sugesstion: mixture model with separate transformations from latent to data space
- Still open research how we can learn manifolds with complex topology

Section 3: Training

Challanges training $\mathcal{M}$-Flows

1. Maximum likelihood not good enough:
   - Maximizing the likelihood $p(x \mid \phi_f, \phi_h)$ does not encourage the network to learn the correct manifold
   - Extreme case: model manifold is perpendicular to the true data manifold (might project all points to a small region whit high density)
2. Computing the determinant of $J_g^T J_g$ is not efficient ($J_g \in \mathbb{R}^{n \times d}$ is not square)

$$L[h] = -\frac{1}{n} \sum_x \left(\log p_{\tilde{u}}(h^{-1}(u)) - \log\det J_h(h^{-1}(u)) - \frac{1}{2} \log\det \left[J_g^T(u) J_g(u) \right] \right)$$

Training $\mathcal{M}$-Flows I

- Different training strategies are possible
- Separate manifold and density training (M/D) into two phases
  - In the same spirit as the EM-algorithm

  1. Manifold phase: only learn function g using the reconstruction error
  2. Density phase: only learn function h using maximum likelihood

- Loss of the density phase with $u = g^{-1}(x)$ is

$$L[h] = -\frac{1}{n}\sum_x \left(\log p_{\tilde{u}}(h^{-1}(u)) - \log\det J_h(h^{-1}(u)) - \frac{1}{2}\log\det\left[J_g^T(u)J_g(u)\right]\right)$$

- Using maximum likelihood only to update h allows for efficient computation, because we do not need to evaluate $\det\left[J_g^T(u)J_g(u)\right]$

Training $\mathcal{M}$-Flows II

- M/D-training schedule: Alternating vs Sequential update
- Likelihood evaluation during inference or testing is still problematic for high dimensional data

Mixture model on a polynomial surface I

- Comparison of $\mathcal{M}$-flow, $\mathcal{M}_e$-flow, AF, PIE and PAE

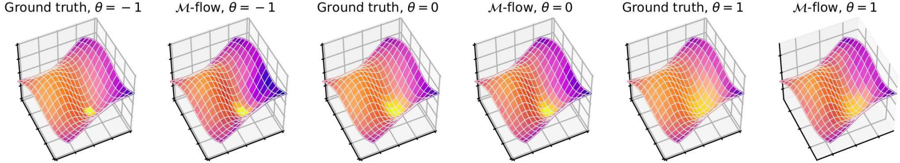


Figure: Polynomial Surface

- Synthetic example with a two-dimensional mainfold in $\mathbb{R}^3$
- Training and test data is genearted by a mixture of two Gaussians in the latent space, conditioned on the parameter $\theta \in [-1, 1]$
- In all metrics except out-of-distribution detection $\mathcal{M}$-flows provide the best results

Mixture model on a polynomial surface II

- MMD: Maximum mean discrepancies between the true and the approximate posteriors
- OOD: Out-of-distribution detection

Model	Manifold distance	Reconstruction error	Posterior MMD	OOD AUC
AF	0.005	–	0.071	**0.990**
PIE (original)	0.035	1.278	0.131	0.933
PIE (uncond. manifold)	0.006	1.253	0.075	0.972
PAE	**0.002**	**0.002**	–	**0.990**
$\mathcal{M}$-flow (alternating M/D)	**0.002**	**0.003**	**0.020**	**0.986**
$\mathcal{M}$-flow (sequential M/D)	0.009	0.013	**0.017**	0.961
$\mathcal{M}_e$-flow (alternating M/D)	**0.003**	**0.003**	0.030	0.985
$\mathcal{M}_e$-flow (sequential M/D)	**0.002**	**0.002**	**0.007**	**0.987**

Figure: Polynomial Surface Results

Image manifolds I

- Three image datasets
  - The images are generated with a Style-GAN2 model trained on the Flickr-Faces-HQ (FFHQ) dataset downsampled to 64x64
    - Sample n latent variable of the GAN while keeping others fixed
    - n = 2 and n = 64 (traing set size: 10'000 and 20'000 images)
  - Third dataset is the real-world CelebA-HQ set downsampled to 64x64 images (existence of a manifold nor its dimension is known)
    - n was set to 512
- Results
  - $\mathcal{M}$-flows better on StyleGAN
  - Slightly worse than AF on the CelebA-HQ dataset (possible reason: suboptimal choice of n)

- FID: Fréchet inception distance

Model	FID scores			Log posterior
	$n = 2$	$n = 64$	CelebA	$n = 64$
AF	58.3 $\pm$ 1.5	24.0 $\pm$ 0.0	**33.6** $\pm$ 0.2	0.17 $\pm$ 1.18
PIE	139.5 $\pm$ 5.0	32.2 $\pm$ 0.8	75.7 $\pm$ 5.1	$-$6.40 $\pm$ 1.54
$\mathcal{M}$-flow	43.9 $\pm$ 0.2	**20.8** $\pm$ 0.5	37.4 $\pm$ 0.2	**2.67** $\pm$ 0.27
$\mathcal{M}_e$-flow	**43.5** $\pm$ 0.2	23.7 $\pm$ 0.2	35.8 $\pm$ 0.4	1.81 $\pm$ 0.70

Figure: Image Manifolds: Results

Image manifolds III

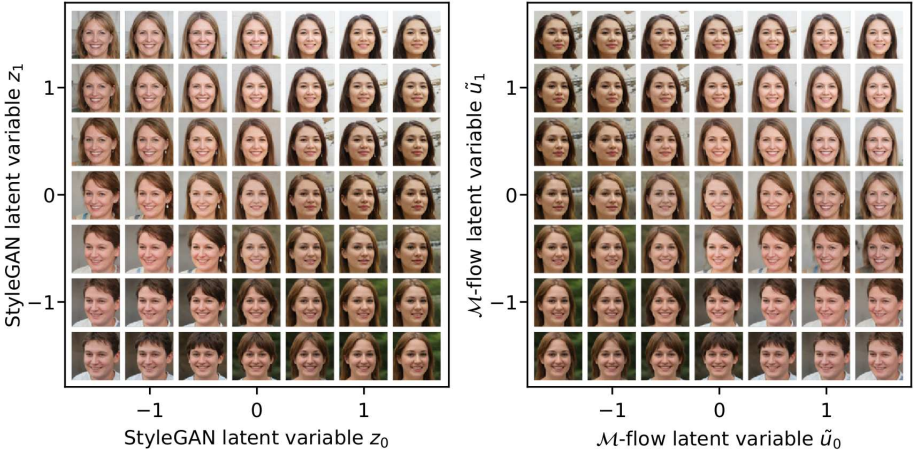


Figure: StyleGAN: Latent Representation

Image manifolds IV

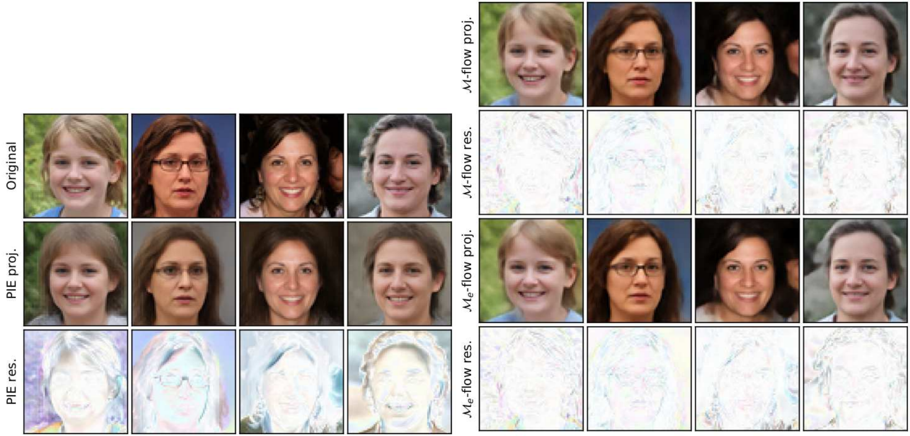


Figure: StyleGAN: Projection onto manifold (n=64)

Section 5: Critic

Critic

- Good
  - Well structured
  - Motivation through comparison of different models
  - Novel model to learn both the manifold and the density
- Improvements
  - Experiments
    - More real data (mostly synthetic or simulation data was used)
    - Comparison to GAN
  - In depth analysis of $\mathcal{M}$-Flow versus $\mathcal{M}_e$-Flow performance
  - Deeper analysis of computational complexity

Discussion

Appendix

Section 7: Introduction

Generative adversarial networks[5]

- Generator
  - Prior distribution on input noise variables $p_z(z)$
  - Mapping from z to the data space X modeled by a differential function $G(z;\theta_g)$ (modeled by a MLP with parameters θ_g)
- Discriminator
  - $D(x;\theta_d)$ modeled by a MLP that outputs a single output the probability that x came from real data rather than from p_G the generator
- Goal learn the distribution of the generator $p_G(x)$
- Training
  - train D to maximize the probability of discriminating the correct label
  - train simultaneously G to minimize $\log(1 - D(G(z))$

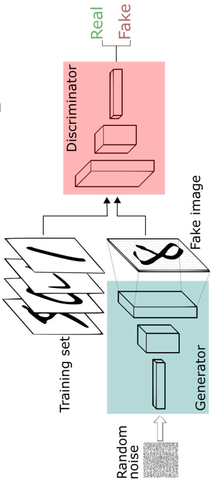


Figure: GAN[4]

[4]TDS: Generative Adversarial Networks — Explained

[5]Generative Adversarial Nets, Ian J. Goodfellow et al.

Variational Autoencoders[6][7]

- Dataset $X = \{x^{(i)}\}_{i=1}^{N}$ of N i.i.d samples
- Assumption: Data is generated by some random process
  1. $z^{(i)}$ is generated from prior $p_{\theta^*}(z)$
  2. $x^{(i)}$ is generated from $p_{\theta^*}(x \mid z)$
- θ^* and the values of the latent variables $z^{(i)}$ are unknown
- Goal: compute approximation $q_\phi(z \mid x)$ of the intractable true posterior $p_\theta(z \mid x) = \frac{p_\theta(z)p_\theta(x|z)}{p_\theta(x)}$
- Training is done by optimizing the ELBO

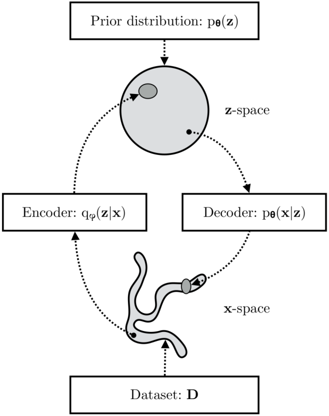


Figure: VAE

[6] Auto-Encoding Variational Bayes, Diederik P. Kingma, Max Welling

[7] An Introduction to Variational Autoencoders, Diederik P. Kingma, Max Welling

Ambient flow (AF) I

$$\text{AF:} \qquad x \xleftarrow{\;f\;} (u,v) \sim p_{uv}$$

- Euclidean normalizing flow $f : U \times V \mapsto X$
  - is a diffeomorphism in the ambient/surrounding space
  - can be modeled as a neural network
- Tractable base density $p_{u,v}(u,v)$
- The density of the data is than given by

$$p_x(x) = p_{u,v}(f^{-1}(x)) \left|\det J_f(f^{-1}(x))\right|^{-1}$$

- Generative mode:
  - sample $(u,v) \sim p_{u,v}$
  - apply the transformation $x = f(u,v)$, which leads to samples $x \sim p_x(x)$

Ambient flow (AF) II

- No explicit alignment of the latent variables u, v
  $\longrightarrow$ no notion of the data manifold

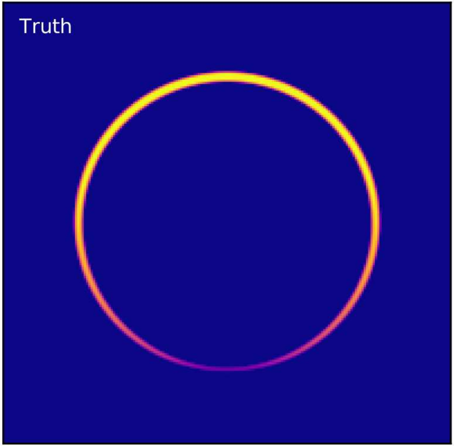


Figure: Gaussian on a circle - Truth

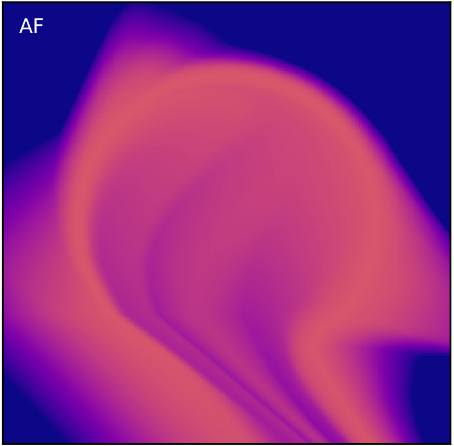


Figure: Gaussian on a circle - AF

Flow on a prescribed manifold (FOM) I

$$\text{FOM:} \quad x \xleftarrow{g^*} u \xleftarrow{h} \tilde{u} \sim p_{\tilde{u}}$$

- Given a priori: Chart $g^* : U \mapsto \mathcal{M}^* \subset X$ for the data manifold
- Than the density (only defined) over $\mathcal{M}^*$ is

$$p_{\mathcal{M}^*}(x) = p_u(g^{*-1}(x)) \left| \det \left[J_g^T(g^{*-1}(x)) J_g(g^{*-1}(x)) \right] \right|^{-\frac{1}{2}},$$

where J_g is the Jacobian a $n \times d$ matrix of g^*

- $p_u(u)$ is modeled with a normalizing flow (diffeomorphism) h
- h maps from a set of latent variables $\tilde{u} \sim p_{\tilde{u}}(\tilde{u})$ to u and is learnable

Flow on a prescribed manifold (FOM) II

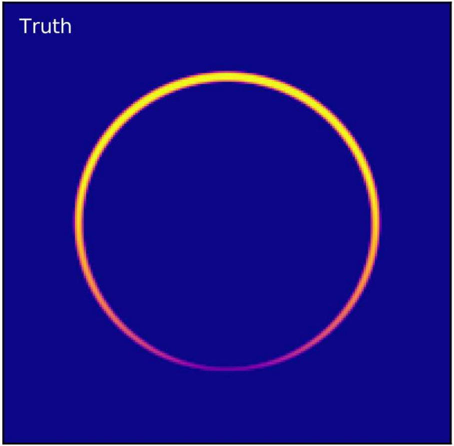


Figure: Gaussian on a circle - Truth

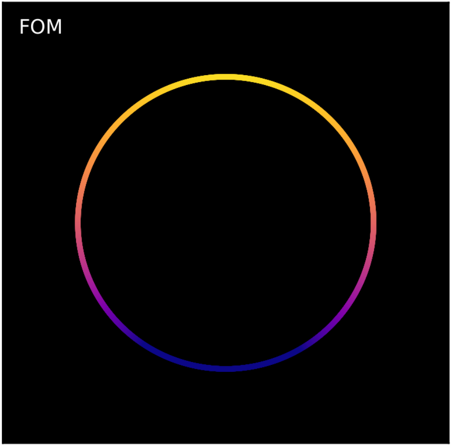


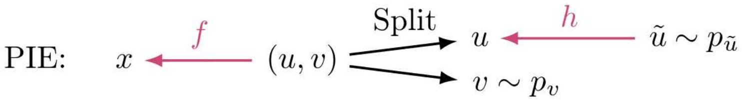
Figure: Gaussian on a circle - FOM

Generative adversarial networks (GAN)

$$\text{GAN:} \quad x \xleftarrow{\;g\;} u \sim p_u$$

- Learn a mapping from the n-dimensional latent space U to the data space: $g : U \mapsto M \subset X$
- g might not be invertible nor injective
- Hence g is not a chart and the transformation not a manifold
- Pro: g is not restricted, which increases the expressivenes of the neural network
- Con: Model density is intractable

Pseudo-invertible encoder (PIE) I



- Splits the latent variables u, v of an ambient flow into two vectors sampled from different base densities
- The distribution of u models the coordinates on the manifold. h is again a n-dim. eucledian flow that maps to the latent variable $\tilde{u}$.
- v should represent the off-dimension-direction of the manifold in the latent space. $p_v(v)$ is the base density that sharply peaks around 0.
- In practice with this bias an alignment of u with the data manifold can be achieved.
- Sampling: $u \sim p_u(u)$ and apply $x = f(u, 0)$

Pseudo-invertible encoder (PIE) II

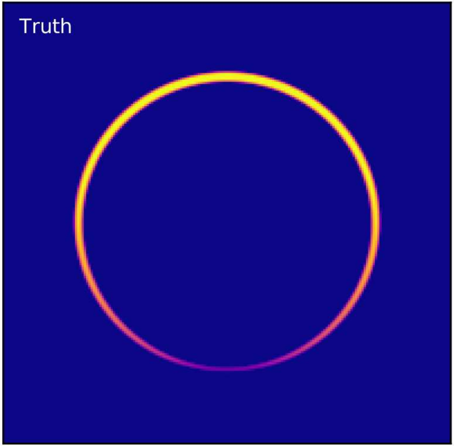


Figure: Gaussian on a circle - Truth

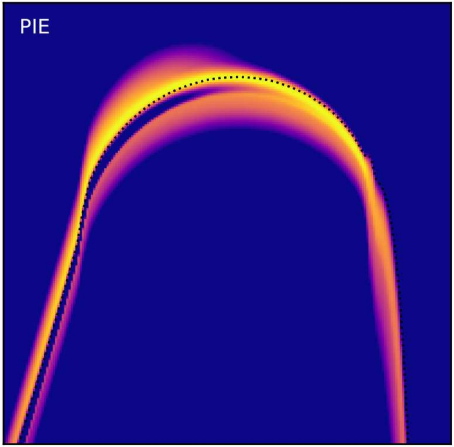


Figure: Gaussian on a circle - PIE

Model comparison overview

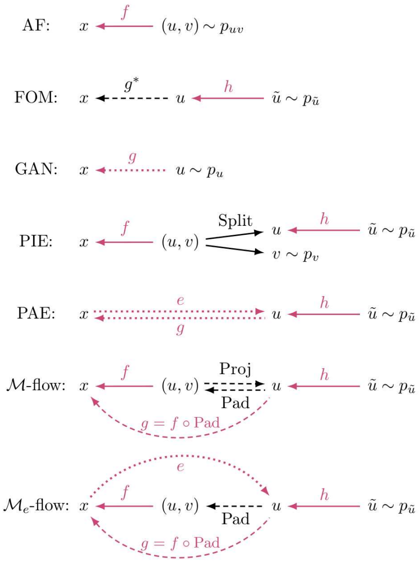


Figure: Model overview

Model	Manifold	Chart	Tractable density	Restr. to $\mathcal{M}$
AF	no manifold	×	✓	×
FOM	prescribed	✓	✓	✓
GAN	learned	×	×	✓
VAE	learned	×	only ELBO	(×)
PIE	learned	✓	✓	(×)
PAE	learned	×	×	✓
$\mathcal{M}$-flow	learned	✓	✓	✓
$\mathcal{M}_e$-flow	learned	✓	✓	✓

Figure: Model comparison

Naive maximum likelihood training

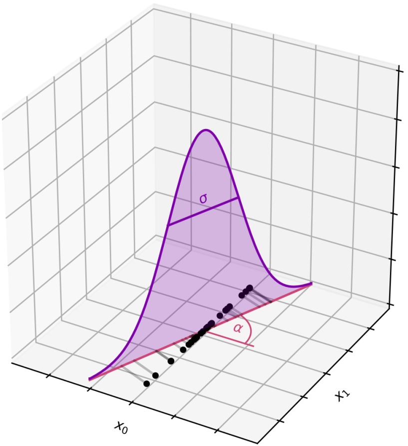


Figure: Setup

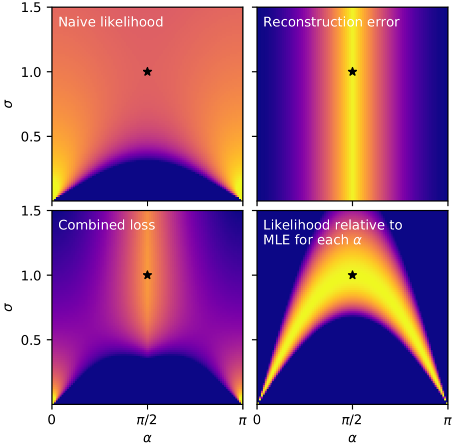


Figure: Loss function

M/D-Phase training

1. M: manifold phase
   - Update the parameters of f
   - f defines the manifold $\mathcal{M}$ and the chart g
   - We minimize the squared reconstruction error

     $$L^{M/D}_{manifolod}[g] = \frac{1}{b} \sum_x ||x - g(g^{-1}(x))||_2^2$$

     where b is the batch size.
   - For the $\mathcal{M}_e$ model we also update the parameters of the encoder e
2. D: density phase
   - Update only the parameters of h by maximum likelihood
   - The manifold stays fixed, because h only affects the density p_u
   - We minimize the negative log likelihood

     $$L^{M/D}_{density}[h] = -\frac{1}{b} \sum_x \log p_u(g^{-1}(x))$$

Experiments general architecture

- Comparision of: $\mathcal{M}$, $\mathcal{M}_e$-flows, AF, PIE, PAE
- Based on rational-quadratic neural spline flows
- For tabular datasets
  - model transformations f an h by alternating coupling layers (20-35 layers)
- For image datasets
  - f is based on a multi-scale architecture (20-28 layers) across four levels interspersed with actnorm and 1x1 convolution layers
  - for $\mathcal{M}$-flows, $\mathcal{M}_e$-flows and PIE: additionally for a subset of the channels two invertible transformations are applied before the projection to the manifold coordinates u
  - h is the same as for the tabular datasets

StyleGAN Samples

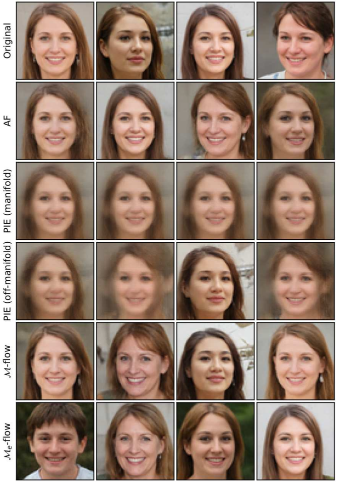


Figure: StyleGAN: Generated Samples (n = 2)

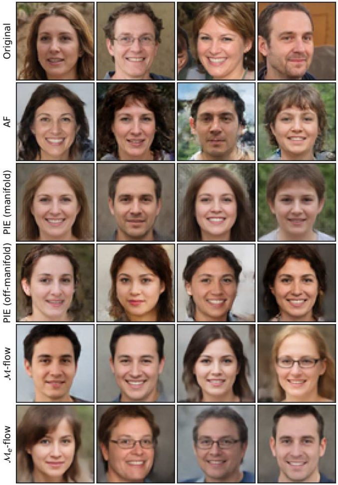


Figure: StyleGAN: Generated Samples (n = 64)

CelebA-HQ Results

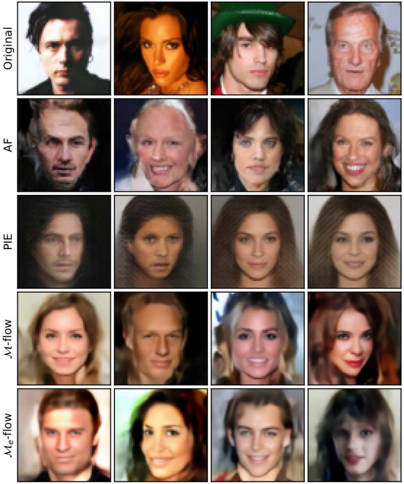


Figure: CelebA-HQ

Model	FID score	Reconstruction error
AF	**33.6** ± 0.2	–
PIE	75.7 ± 5.1	6970 ± 97
$\mathcal{M}$-flow	37.4 ± 0.2	**830** ± 5
$\mathcal{M}_e$-flow	35.8 ± 0.4	991 ± 4

Figure: CelebA-HQ: Results

Lorenz attractor

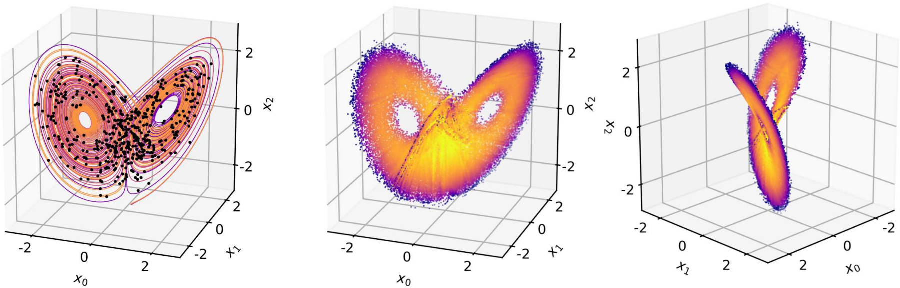


Figure: Lorenz Attractor

- $x \in \mathbb{R}^3$ changes with time based on three diffrential equations
- With certain initial parameter choices the solution tends towards the Lorenz attractor
- Generated 100 trajectories with different initial conditions (from t $= 0$ to t $= 1000$)

Particle physics I

- Proton-proton collision at the Large Hadron Collider
- Observations $x \in \mathbb{R}^{40}$
- $p(x \mid \theta)$ is conditional on 3 constants of nature θ
- By the laws of particle physics the data must be restricted by a 14-dimensional manifold
- The goal is to infer the posterior over θ given observations x_i
- Trained on 10^6 samples generated by a simulator

Particle physics II

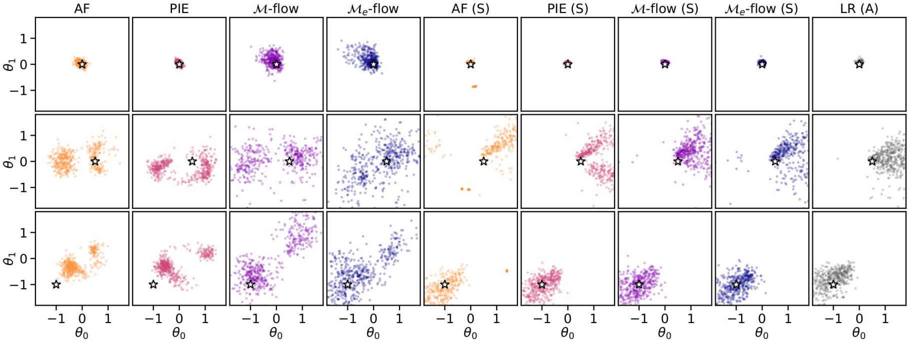


Figure: Posterior samples from an MCMC

Particle physics III

Model (algorithm)	Sample closure	Mean reconstruction error	Log posterior
AF	**0.0019** ± 0.0001	–	-3.94 ± 0.87
PIE (original)	0.0023 ± 0.0001	2.054 ± 0.076	-4.68 ± 1.56
PIE (unconditional manifold)	0.0022 ± 0.0001	1.681 ± 0.136	-1.82 ± 0.18
PAE	0.0073 ± 0.0001	0.052 ± 0.001	–
$\mathcal{M}$-flow	0.0045 ± 0.0004	**0.012** ± 0.001	-1.71 ± 0.30
$\mathcal{M}_e$-flow	0.0046 ± 0.0002	0.029 ± 0.001	-1.44 ± 0.34
AF (SCANDAL)	0.0565 ± 0.0059	–	-0.40 ± 0.09
PIE (original, SCANDAL)	0.1293 ± 0.0218	3.090 ± 0.052	0.03 ± 0.17
PIE (uncond. manifold, SCANDAL)	0.1019 ± 0.0104	1.751 ± 0.064	**0.23** ± 0.05
PAE (SCANDAL)	0.0323 ± 0.0010	0.053 ± 0.001	–
$\mathcal{M}$-flow (SCANDAL)	0.0371 ± 0.0030	**0.011** ± 0.001	0.11 ± 0.04
$\mathcal{M}_e$-flow (SCANDAL)	0.0291 ± 0.0010	0.030 ± 0.002	**0.14** ± 0.09
Likelihood ratio estimator (ALICES)	–	–	0.05 ± 0.05

Figure: Results

Section 11: Change of variable

Change of variable[8]

- $M \subset \mathbb{R}^m$ where M is a n-manifold and $n < m$
- $\vec{x} = \phi(\vec{u}) : \mathbb{R}^n \mapsto \mathbb{R}^m$ maps the embedded manifold to its intrinsic Euclidean space
- $d\vec{x} = \sqrt{\det J_\phi^T J_\phi} d\vec{u}$ describes the change of volumes
- Formula to compute the density over M

$$\int_{M \subset \mathbb{R}^m} f(\vec{x}) d\vec{x} = \int_{\mathbb{R}^n} (f \circ \phi)(\vec{u}) \sqrt{\det J_\phi^T J_\phi} d\vec{u}$$

- Then the density becomes

$$p(\vec{u}) = (f \circ \phi)(\vec{u}) \sqrt{\det J_\phi^T J_\phi(\vec{u})} = f(\vec{x}) \sqrt{\det J_\phi^T J_\phi(\phi^{-1}(\vec{x}))}$$

[8] Normalizing Flows on Riemannian Manifolds, Mevlana C. Gemici et al.