Normalising flows and continuously indexed flows for machine learning

Rob Cornish

Department of Statistics, University of Oxford

December 10, 2021

Outline

1. Normalising flows overview
2. Variational inference
3. Density estimation
4. Practical implementations
5. Limitations
6. Continuously-indexed flows

Overview

Motivation

It is often important to parameterise an expressive families of densities

Key tasks:

- Variational inference: find $\operatorname{argmin}_\phi \mathrm{KL}(q_\phi \parallel p(\cdot \mid \overline{X}))$
- Density estimation: determine p_{data} from $X_1, \ldots, X_n \overset{\text{i.i.d.}}{\sim} p_{\mathrm{data}}$

Normalising flows use neural networks to parameterise families of diffeomorphisms, which induce densities via the change-of-variables formula

Key idea

Parameterise a family of diffeomorphisms f_ψ and choose a fixed noise distribution p_Z

Define model p_ψ to be the W marginal of the following generative process:

$$Z \sim p_Z \qquad W := f_\psi(Z)$$

This gives a procedure for sampling from p_ψ

Can also compute densities via change-of-variables:

$$p_\psi(w) = p_Z(f_\psi^{-1}(w)) \left| \det \mathrm{D} f_\psi^{-1}(w) \right|$$

where $\mathrm{D} f_\psi^{-1}(w)$ denotes the Jacobian of f_ψ^{-1} evaluated at w.

Variational inference

Overview

Assume a Bayesian model $p_{\overline{X},Y}$ with prior p_Y and likelihood $p_{\overline{X}|Y}$, e.g.

$$p_{\overline{X},Y}(\overline{x}, y) = p_Y(y) \prod_{i=1}^{n} p_{X|Y}(x_i \mid y)$$

Observe data $\overline{X} = (X_1, \ldots, X_n)$, and seek posterior $p_{Y|\overline{X}}(\cdot \mid \overline{X})$

Variational inference: (non-amortised)

1. Posit a family of approximate posteriors q_ϕ on Y-space
2. Approximate the true posterior via

$$\operatorname{argmin}_\phi \mathrm{KL}(q_\phi \,\|\, p_{Y|\overline{X}}(\cdot \mid \overline{X}))$$

Mean-field approximation

For high-quality inference, expressiveness of q_ϕ is key; otherwise $\min_\phi \mathrm{KL}(q_\phi \parallel p_{Y|\overline{X}}(\cdot \mid \overline{X}))$ will be large (for complex posteriors)

One approach is mean-field:

$$q_\phi(y) = \prod_{i=1}^{\dim(Y)} q_i(y_i; \phi),$$

where e.g. $\phi = (\overline{\mu}, \overline{\sigma})$ and $q_i(y_i; \phi) = \mathrm{Normal}(y_i; \mu_i, \sigma_i)$

Can rewrite as

$$q_\phi(y) = \mathrm{Normal}(y; \overline{\mu}, \overline{\sigma}\, I),$$

so unimodal and axis-aligned, i.e. fairly limited expressiveness

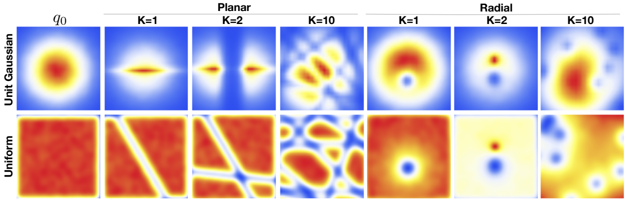
Normalising flows for variational inference

Key idea of Rezende and Mohamed [2015]: use normalising flows to parameterise a more expressive approximate posterior q_ϕ

In particular, take q_ϕ to be distribution of Y, where

$$Z \sim p_Z \qquad Y := f_\phi(Z)$$

with f_ϕ a diffeomorphism



The evidence lower bound (ELBO)

Can compute

$$\begin{aligned}
\mathrm{KL}(q_\phi \parallel p_{Y|\overline{X}}(\cdot \mid \overline{X})) &= -\int q_\phi(y) \log \frac{p_{Y|\overline{X}}(y \mid \overline{X})}{q_\phi(y)}\, dy \\
&= -\int q_\phi(y) \left(\log \frac{p_{\overline{X},Y}(\overline{X}, y)}{q_\phi(y)} - \log p_{\overline{X}}(\overline{X}) \right) dy \\
&= -\int q_\phi(y) \log \frac{p_{\overline{X},Y}(\overline{X}, y)}{q_\phi(y)}\, dy + \log p_{\overline{X}}(\overline{X}),
\end{aligned}$$

so that

$$\mathrm{argmin}_\phi\, \mathrm{KL}(q_\phi \parallel p_{Y|\overline{X}}(\cdot \mid \overline{X})) = \mathrm{argmax}_\phi \underbrace{\int q_\phi(y) \log \frac{p_{\overline{X},Y}(\overline{X}, y)}{q_\phi(y)}\, dy}_{=:\mathrm{ELBO}(\phi)}$$

Optimising the ELBO

A general strategy for optimising the ELBO is stochastic gradient ascent

For normalising flows, since q_ϕ is the pushforward of p_Z by f_ϕ,

$$\begin{aligned} \mathrm{ELBO}(\phi) &:= \int q_\phi(y) \log \frac{p_{\overline{X},Y}(\overline{X}, y)}{q_\phi(y)} \, dy \\ &= \int p_Z(z) \log \frac{p_{\overline{X},Y}(\overline{X}, f_\phi(z))}{q_\phi(f_\phi(z))} \, dz \end{aligned}$$

By differentiating under the integral sign,

$$\nabla_\phi \, \mathrm{ELBO}(\phi) = \int p_Z(z) \, \nabla_\phi \log \frac{p_{\overline{X},Y}(\overline{X}, f_\phi(z))}{q_\phi(f_\phi(z))} \, dz,$$

so that if $Z \sim p_Z$, then $\nabla_\phi \log \frac{p_{\overline{X},Y}(\overline{X}, f_\phi(Z))}{q_\phi(f_\phi(Z))}$ is an unbiased estimate of $\nabla_\phi \, \mathrm{ELBO}(\phi)$ suitable for optimisation

Summary

When using normalising flows for variational inference:

1. Choose p_Z and parameterise f_ϕ
2. Sample $Z \sim p_Z$ and compute $\nabla_\phi \log \frac{p_{\overline{X},Y}(\overline{X}, f_\phi(Z))}{q_\phi(f_\phi(Z))}$
3. Update ϕ via stochastic gradient ascent

In practice:

- Use neural network for f_ϕ (can for p_Z also if reparameterisable)
- Obtain ϕ gradient via autodiff
- Must be able to sample efficiently from q_ϕ (i.e. compute $f_\phi(z)$)
- Only need to be able to compute efficiently

$$q_\phi(f_\phi(Z)) = p_Z(f_\phi^{-1}(f_\phi(Z))) \left|\det \mathrm{D} f_\phi^{-1}(f_\phi(Z))\right| = p_Z(Z) \left|\det \mathrm{D} f_\phi(Z)\right|^{-1}$$

  (or an unbiased estimate of its log), i.e. not $f_\phi^{-1}(y)$ given only y
- Can amortise this procedure

Density estimation

Overview

Aim: determine p_{data} from samples $X_1, \ldots, X_n \overset{\text{i.i.d.}}{\sim} p_{\mathrm{data}}$

Applications:

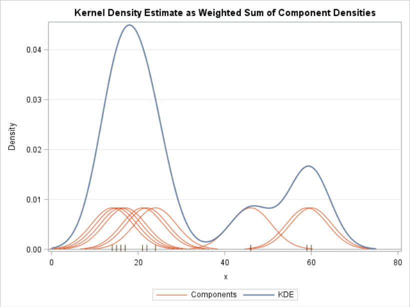
- Out-of-distribution detection
- Synthetic data generation

Illustrative approach

Kernel density estimation: approximate density of p_{data} by

$$p(x) := \frac{1}{n}\sum_{i=1}^{n} k(x - X_i),$$

where k is e.g. a scaled Gaussian



Source: `blogs.sas.com`

Motivation

Curse of dimensionality $\Rightarrow$ different strategies needed in high dimensions

Neural networks have had great success with high-dimensional data e.g. in classification problems

How can we leverage this expressiveness for density estimation?

High-level strategy

Alternatively, consider projecting p_{data} onto a model family, i.e. estimate is

$$\operatorname{argmin}_{\theta \in \Theta} \mathrm{KL}(p_{\mathrm{data}} \parallel p_\theta)$$

where p_θ is a parametrised density

As with variational inference, expressiveness of p_θ is clearly important

Normalising flows for density estimation

A popular idea is to use normalising flows to define p_θ, i.e. p_θ is the X marginal of the following generative process:

$$Z \sim p_Z \qquad X := f_\theta(Z),$$

where f_θ is a parameterised diffeomorphism

This gives a procedure for sampling from p_θ

Can also compute densities via change-of-variables:

$$p_\theta(x) = p_Z(f_\theta^{-1}(x)) \left| \det \mathrm{D} f_\theta^{-1}(x) \right|$$

where $\mathrm{D} f_\theta^{-1}(x)$ denotes the Jacobian of f_θ^{-1} evaluated at x

A nice feature is that this is often exactly tractable by construction

Gradient estimation

By differentiating under the integral sign

$$\nabla_\theta \, \mathrm{KL}(p_{\mathrm{data}} \parallel p_\theta) = -\nabla_\theta \int p_{\mathrm{data}}(x) \log \frac{p_\theta(x)}{p_{\mathrm{data}}(x)} \, dx$$
$$= -\int p_{\mathrm{data}}(x) \, \nabla_\theta \log p_\theta(x) \, dx,$$

so if $X \sim p_{\mathrm{data}}$, then $-\nabla_\theta \log p_\theta(X)$ is an unbiased gradient estimate

This allows finding $\mathrm{argmin}_\theta \, \mathrm{KL}(p_{\mathrm{data}} \parallel p_\theta)$ by stochastic gradient descent

Summary

When using normalising flows for density estimation:

1. Choose p_Z and parameterise f_θ
2. Obtain $X \sim p_{\mathrm{data}}$ and compute $-\nabla_\theta \log p_\theta(X)$
3. Update θ via stochastic gradient descent

In practice:

- Use neural network for f_θ
- Obtain θ gradient via autodiff
- Must be able to compute efficiently

$$p_\theta(x) = p_Z(f_\theta^{-1}(x)) \left|\det \mathrm{D} f_\theta^{-1}(x)\right|$$

(or an unbiased estimate of $\nabla_\theta \log p_\theta(x)$)

- Don't need to be able to sample from p_θ

Some flow architectures

Key considerations

Want to parameterise a family of diffeomorphisms f_ψ

Key requirements:

- f_ψ must be invertible (in practice this may be implicit)
- Tractable log Jacobian (or tractable unbiased estimate)

Can compose flows to obtain greater complexity

RealNVP

For $w \in \mathbb{R}^D$ and $1 \leq d < D$, Dinh et al. [2017] defines

$$f_\psi(w) = \begin{bmatrix} w_{1:d} \\ \exp(s(w_{1:d};\psi)) \odot w_{d+1:D} + t(w_{1:d};\psi) \end{bmatrix}$$

where $s, t : \mathbb{R}^d \to \mathbb{R}^{D-d}$ are unconstrained neural networks

Clear that this is invertible

Jacobian matrix is lower-triangular, so determinant is tractable

Inverse Autoregressive Flow / Masked Autoregressive Flow

For $w \in \mathbb{R}^D$, define

$$f_i(w; \psi) = \exp(s_i(w_{1:i-1}; \psi)) \odot w_i + t_i(w_{1:i-1}; \psi),$$

where $s_i, t_i : \mathbb{R}^{i-1} \to \mathbb{R}$ are unconstrained neural networks, and set

$$f_\psi(w) := \begin{bmatrix} f_1(w; \psi) \\ \vdots \\ f_D(w; \psi) \end{bmatrix}$$

Again invertible and triangular Jacobian matrix

For efficiency (in one direction), MADE [Germain et al., 2015] provides a way to parameterise an autoregressive neural network

Used by Kingma et al. [2016] (for VI) and Papamakarios et al. [2017] (for density estimation)

Residual Flows

Triangular Jacobians seem to reduce expressiveness

Alternative strategy [Behrmann et al., 2019, Chen et al., 2020]: if $g_\psi : \mathbb{R}^D \to \mathbb{R}^D$ has $\operatorname{Lip} g_\psi < 1$, then we get a diffeomorphism

$$f_\psi(w) := w + g_\psi(w)$$

Can ensure a neural network g_ψ is Lipschitz via spectral normalisation

Can estimate Jacobians unbiasedly by expanding as a matrix power series, and using debiasing techniques plus the Skilling-Hutchinson trace estimator

Can be inverted numerically by Banach Fixed Point theorem

Limitations of normalising flows

Intuition

Diffeomorphisms preserve topological properties of their input space, e.g.

- Number of connected components
- Number of "holes"
- How the space is "knotted"

Intuitively suggests that if $X = f(Z)$ then the supports of X and Z will share the same topological properties

Precise result

Theorem (Cornish et al. [2020])

If $\operatorname{supp} p_Z$ *is not homeomorphic to* $\operatorname{supp} p_{\mathrm{data}}$*, then a sequence of diffeomorphisms* f_n *can yield* $f_n(Z) \to p_{\mathrm{data}}$ *in distribution only if*

$$\max\{\mathrm{Lip}\, f_n, \mathrm{Lip}\, f_n^{-1}\} \to \infty$$

Convergence in distribution straightforwardly implies a version in terms of being "approximately not homeomorphic"

General consequence: numerical noninvertibility (observed by Behrmann et al. [2020])

Consequences for Residual Flows

The following densities were learned using a Gaussian prior with a 10-layer Residual Flow [Behrmann et al., 2019, Chen et al., 2020]

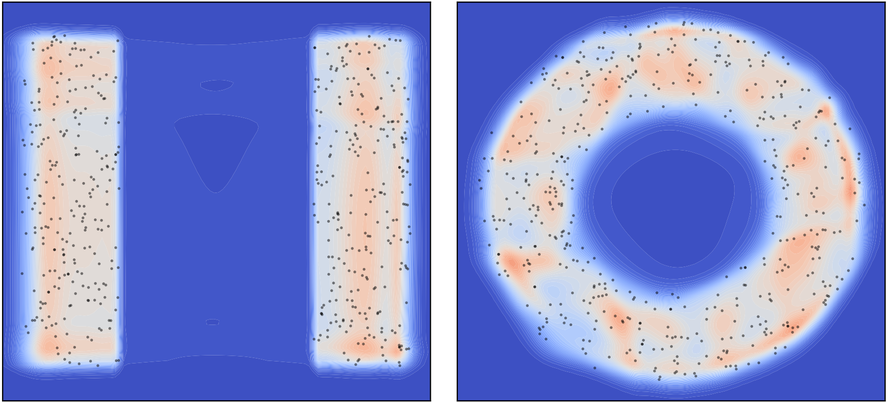


Figure: Darker regions indicate lower density. Data shown in black.

Continuously-Indexed Normalizing Flows (for density estimation)

Continuously-Indexed Flow Layer

To gain expressiveness over baseline flows, continuously-indexed flows (CIFs) now model the data as the X-marginal of

$$Z \sim p_Z, \qquad U \mid Z \sim p_{U|Z}(\cdot \mid Z), \qquad X := F(Z; U)$$

where

- U is a continuous index variable
- $p_{U|Z}$ is a (parametrized) conditional distribution
- $F(\cdot; u)$ is a diffeomorphism for every u

Any existing normalizing flow f can be used to construct F, e.g. via

$$F(z; u) := f\left(e^{s(u)} \odot z + t(u)\right)$$

for neural networks s, t outputting values in Z-space

Multi-Layer CIFs

An L-layer CIF is obtained by stacking the single-layer model:

$$
\begin{aligned}
& && Z_0 \sim p_{Z_0}, \\
U_1 &\sim p_{U_1|Z_0}(\cdot \mid Z_0), && Z_1 := F_1(Z_0; U_1), \\
& && \cdots \\
U_L &\sim P_{U_L|Z_{L-1}}(\cdot \mid Z_{L-1}), && X := F_L(Z_{L-1}; U_L)
\end{aligned}
$$

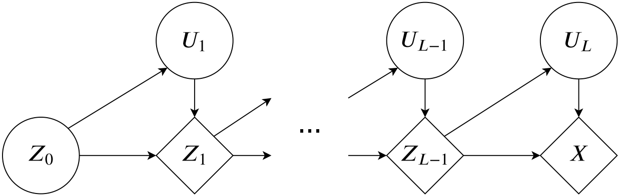


Figure: Graphical multi-layer CIF generative model

Training and inference

The marginal p_X is intractable, but the *joint* $p_{X,U_{1:L}}$ has a closed-form, e.g. for a single layer

$$p_{X,U}(x,u) = p_Z(F^{-1}(x;u))p_{U|Z}(u \mid F^{-1}(x;u))|\det \mathrm{D}F^{-1}(x;u)|$$

Given an inference model $q_{U_{1:L}|X}$, we can use the *ELBO* for training:

$$\mathcal{L}(x) := \mathbb{E}_{u_{1:L} \sim q_{U_{1:L}|X}(\cdot|x)}\left[\log \frac{p_{X,U_{1:L}}(x,u_{1:L})}{q_{U_{1:L}|X}(u_{1:L} \mid x)}\right] \leq \log p_X(x)$$

At test time, we can estimate $\log p_X(x)$ to arbitrary precision using an m-sample *IWAE* estimate with $m \gg 1$

Inference model

To obtain an efficient inference model $q_{U_{1:L}|X}$, we exploit the *conditional independence structure* of $p_{U_{1:L}|X}$ from the forward model:

$$
\begin{aligned}
& Z_L := X, \\
U_L \sim q_{U_L|Z_L}(\cdot \mid Z_L), \qquad & Z_{L-1} := F_L^{-1}(Z_L; U_L), \\
& \cdots \\
U_1 \sim q_{U_1|Z_1}(\cdot \mid Z_1), \qquad & Z_0 := F_1^{-1}(Z_1; U_1)
\end{aligned}
$$

In other words,

$$q_{U_{1:L}|X}(u_{1:L} \mid x) := \prod_{\ell=1}^{L} q_{U_\ell|Z_\ell}(u_\ell \mid z_\ell)$$

This induces a natural weight-sharing scheme between the forward and inverse models, since the same F_ℓ are used in both cases

Benefits over Standard Flows

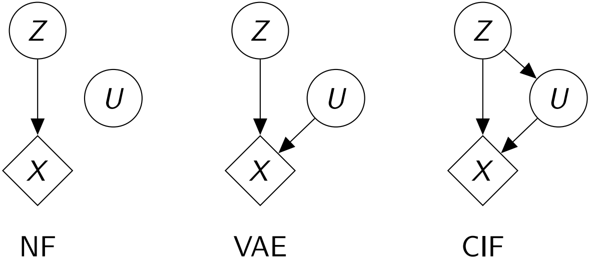
Intuitively, the additional flexibility afforded by $p_{U|Z}$ allows a CIF to "clean up" mass that would be misplaced by a single bijection

Proposition: Under mild conditions on the target and F, there exists $p_{U|Z}$ such that the model p_X has the same support as the target $p_X^\star$

Proposition: If $F(z; \cdot)$ is surjective for each z, there exists $p_{U|Z}$ such that p_X matches $p_X^\star$ exactly

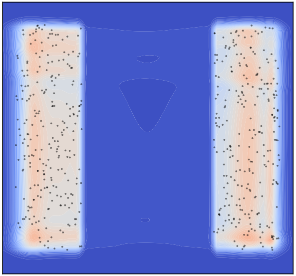
Comparison with Related Models

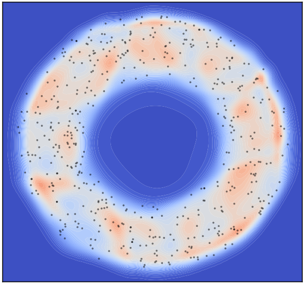
CIFs may be understood as a hybrid between standard normalizing flow and VAE density models:

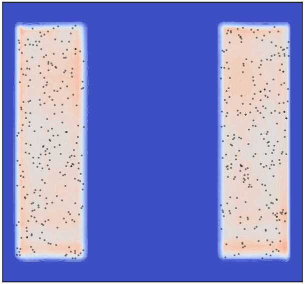


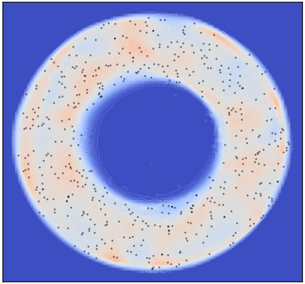
In all cases, $X = F(Z; U)$ for some family of bijections F

2D ResFlow Experiments









2D Masked Autoregressive Flow (MAF) Experiments

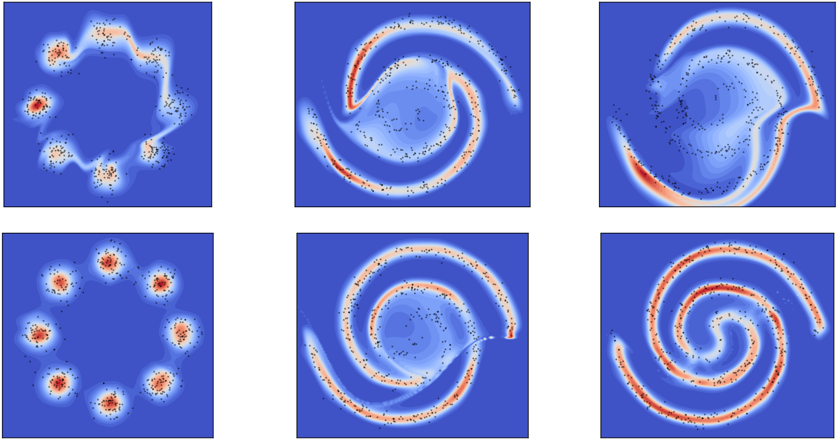


Figure: Density models learned by a 20-layer MAF (above) and a 5-layer CIF-MAF (below) for 2-D target distributions. The far right column uses a higher-capacity model for each method.

High-Dimensional ResFlow Experiments

Table: Test set bits per dimension. Lower is better.

	MNIST	CIFAR-10
ResFlow (small)	1.074	3.474
ResFlow (big)	1.018	3.422
CIF-ResFlow	**0.922**	**3.334**

NB: These ResFlows were smaller than those from Chen et al. [2019]

We obtained similar improvements on several other problems and flow models

Thank you!

Figure: Joint work with Anthony Caterini, George Deligiannidis, Arnaud Doucet, and Dino Sejdinovic

References I

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR*, 2017.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889. PMLR, 2015.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2335–2344, 2017.

References II

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks, 2019.

Ricky T. Q. Chen, Jens Behrmann, David Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling, 2020.

Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows. In *International Conference on Machine Learning*, pages 2133–2143, 2020.

Jens Behrmann, Paul Vicol, Kuan-Chieh Wang, Roger Grosse, and Jörn-Henrik Jacobsen. Understanding and mitigating exploding inverses in invertible neural networks. *arXiv preprint arXiv:2006.09347*, 2020.

Ricky T. Q. Chen, Jens Behrmann, David K Duvenaud, and Joern-Henrik Jacobsen. Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems*, volume 32, 2019.