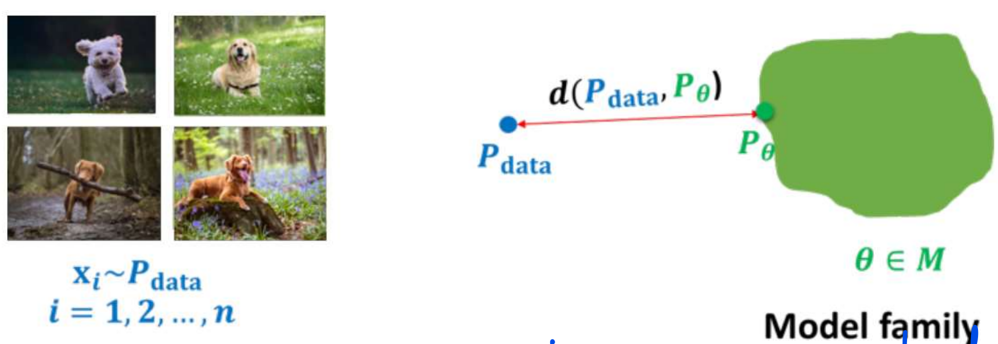
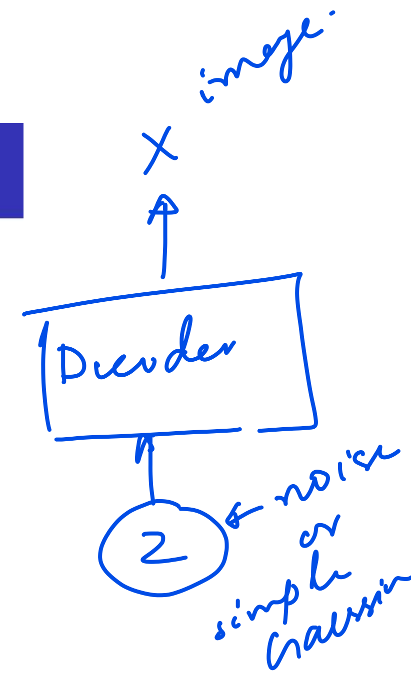
Normalizing flows

Recap of likelihood-based learning so far:





- Model families: Graphical model to express joint distribution.
  - Autoregressive Models: $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i|\mathbf{x}_{<i})$ ✗
  - Variational Autoencoders: $p_\theta(\mathbf{x}) = \int_{\mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$
- Autoregressive models provide tractable likelihoods but no direct mechanism for learning features : actual dependency is best captured in a latent space.
- Variational autoencoders can learn feature representations (via latent variables $\mathbf{z}$) but have intractable marginal likelihoods
- **Key question**: Can we design a latent variable model with tractable likelihoods? Yes!

Simple Prior to Complex Data Distributions

- Desirable properties of any model distribution:
  - Analytic density.
  - Easy-to-sample + Latent space of features.
- Many simple distributions satisfy the above properties e.g., Gaussian, uniform distributions, or a Bayesian network.
- Unfortunately, data distributions could be much more complex (multi-modal)
- **Key idea**: Map simple distributions (easy to sample and evaluate densities) to complex distributions (learned via data) using **change of variables**.

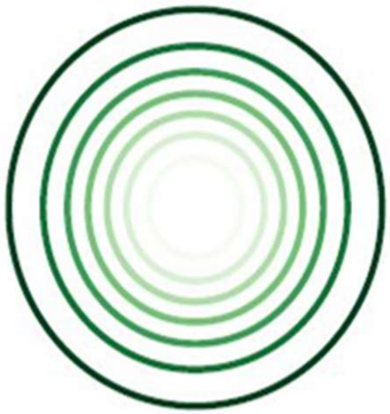
Normalizing Flows for Probabilistic Modeling and Inference

George Papamakarios* GPAPAMAK@GOOGLE.COM
Eric Nalisnick* ENALISNICK@GOOGLE.COM
Danilo Jimenez Rezende DANILOR@GOOGLE.COM
Shakir Mohamed SHAKIR@GOOGLE.COM
Balaji Lakshminarayanan BALAJILN@GOOGLE.COM
DeepMind

Abstract

Normalizing flows provide a general mechanism for defining expressive probability distributions, only requiring the specification of a (usually simple) base distribution and a series of bijective transformations. There has been much recent work on normalizing flows, ranging from improving their expressive power to expanding their application. We believe the field has now matured and is in need of a unified perspective. In this review, we attempt to provide such a perspective by describing flows through the lens of probabilistic modeling and inference. We place special emphasis on the fundamental principles of flow design, and discuss foundational topics such as expressive power and computational trade-offs. We also broaden the conceptual framing of flows by relating them to more general probability transformations. Lastly, we summarize the use of flows for tasks such as generative modeling, approximate inference, and supervised learning.

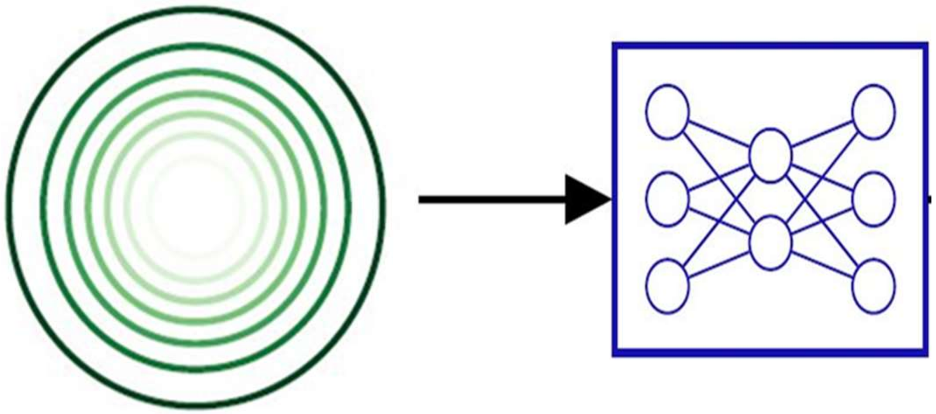
Basics of Normalizing Flows



$$\mathbf{u} \sim p(\mathbf{u}) \sim \mathcal{N}(0, I)$$

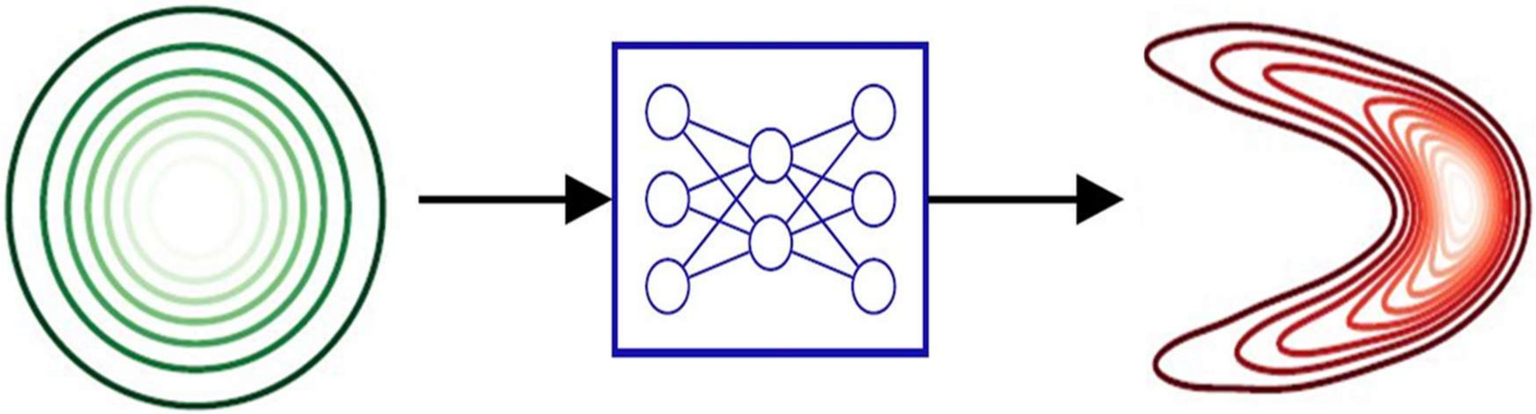
latent space $\mathbf{z}$

Basics of Normalizing Flows



$\mathbf{u} \sim p(\mathbf{u})$ $\qquad T(\mathbf{u})$

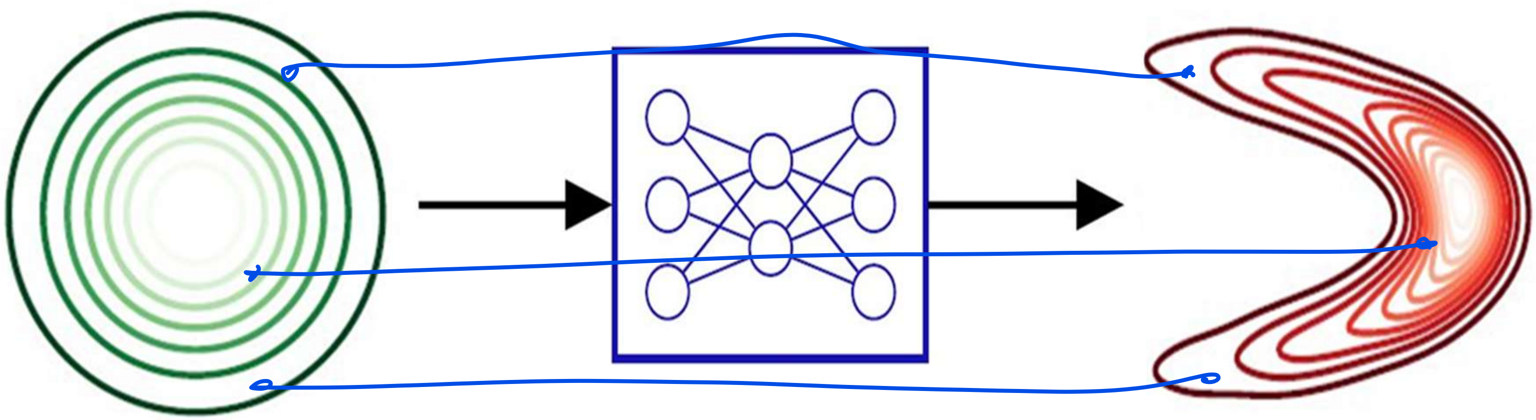
Basics of Normalizing Flows



$\mathbf{u} \sim p(\mathbf{u})$ $\qquad$ $T(\mathbf{u})$ $\qquad$ $\mathbf{x} = T(\mathbf{u})$

Like the decoder of VAEs

Basics of Normalizing Flows



bijection.

$$\mathbf{u} \sim p(\mathbf{u}) \qquad T(\mathbf{u}) \qquad \mathbf{x} = T(\mathbf{u})$$

$\tilde{u} = T^{-1}(x)$ *T* must be invertible $\tilde{x}$

$p(\tilde{x}) = \times \ p(\tilde{u})$

Calculating Density p(x) using change of variables

$x = T(u)$ $x \in \mathbb{R}$; $u \in \mathbb{R}$ X U

What is $p(x)$?

Consider 1-d data $x \in \mathbb{R}$

$$F_X(x) = P(X \leq x)$$

CDF of U

$$= P(U \leq T^{-1}(x)) = F_U(T^{-1}(x))$$

chain rule of differentiation

$$p_X(x) = \frac{\partial}{\partial x} F_X(x) = \frac{\partial}{\partial x} F_U(T^{-1}(x)) = \frac{\partial}{\partial u} F_U(u) \frac{\partial T^{-1}(x)}{\partial x}$$

$$= p_U(u) \frac{\partial T^{-1}(x)}{\partial x}$$

$$\boxed{\frac{\partial T^{-1}(x)}{\partial x}} = \left(\frac{\partial x}{\partial T^{-1}(x)}\right)^{-1} = \boxed{\left(\frac{\partial T(u)}{\partial u}\right)^{-1}}$$

Change of variable formula for 1-D variables.

$$\log p_x(x) = \log p_U(u)\left[\frac{\partial T(u)}{\partial u}\right]^{-1},$$

$$= \log p_u(u) - \log \frac{\partial T(u)}{\partial u}$$

Jacobians

$x \in \mathbb{R}^D$, $u \in \mathbb{R}^D$ T is invertible.

$$x_1 = T_1(u) = T_1(u_1, \ldots u_D)$$

$$x_2 = T_2(u) = T_2(u_1, \ldots u_D)$$

$$\vdots$$

$$x_D = T_n(u) = T_n(u_1 \ldots u_D)$$

Jacobian derivative

$$J_T = \begin{bmatrix} \frac{\partial T_1}{\partial u_1} & \cdots & \frac{\partial T_1}{\partial u_D} \\ \vdots & & \\ \partial T_n / \partial u_1 & & \partial T_D / \partial u_n \end{bmatrix}$$

$$J_{T^{-1}}(x) = \begin{bmatrix} \frac{\partial (T^{-1})_1}{\partial x_1} & \cdots & \frac{(\partial T^{-1})_1}{\partial x_n} \\ \frac{\partial (T^{-1})_n}{\partial x_1} & & \frac{(\partial T^{-1})_n}{\partial x_n} \end{bmatrix}$$

$$J_T(u) = \left[J_{T^{-1}}(x) \right]^{-1} \text{ where } x = T(u)$$

Change-of-Variables Formula

$$\log p_{\mathrm{x}}(\mathbf{x}) = \log p_{\mathrm{u}}(\mathbf{u}) - \log |\det J_T(\mathbf{u})|$$

Jacobian matrix of *T*

$$J_T(\mathbf{u}) = \begin{bmatrix} \frac{\partial T_1}{\partial \mathrm{u}_1} & \cdots & \frac{\partial T_1}{\partial \mathrm{u}_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial T_D}{\partial \mathrm{u}_1} & \cdots & \frac{\partial T_D}{\partial \mathrm{u}_D} \end{bmatrix}$$

Change-of-Variables Formula

$$\log p_{\mathrm{X}}(\mathbf{x}) = \log p_{\mathrm{u}}(\mathbf{u}) - \log |\det J_T(\mathbf{u})|$$

Jacobian matrix of *T*

In practice, two requirements on *T*:

1. Invertible
2. Easy-to-compute determinant of Jacobian

Flows Support Two Core Operations

Sampling:

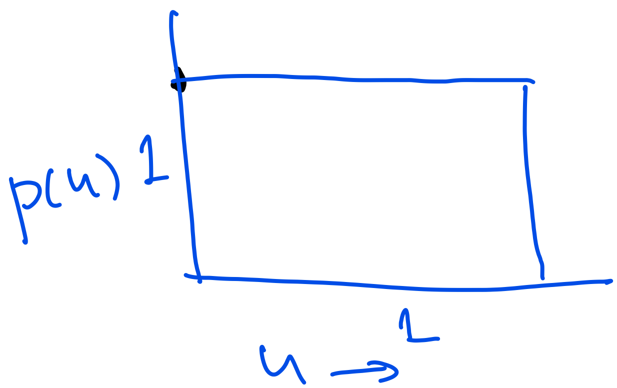
$$\hat{\mathbf{u}} \sim p(\mathbf{u}), \quad \hat{\mathbf{x}} = T(\hat{\mathbf{u}})$$

Density Evaluation:

$$\log p_{\mathrm{x}}(\mathbf{x}) \;=\; \log p_{\mathrm{u}}(T^{-1}(\mathbf{x})) \;+\; \log|\det J_{T^{-1}}(\mathbf{x})|$$

Example Uniform distribution

$u \sim U(0, 1)$



$$T^{-1}(x) = \frac{x}{4}$$

$$x = 4u = T(u)$$

$$p(x) = p(u) \frac{\partial T^{-1}(x)}{\partial x}$$

$$1 \cdot \frac{\partial (x/4)}{\partial x} = \frac{1}{4}$$

$$p(x) = \frac{1}{4}.$$

$$x \sim U(0, 4) \cdot \quad p(x) = \frac{1}{4}$$

Normalizing flows - simplest case - linear transform in 1D

Normalizing Flows allow for defining complex densities by transforming simple one by invertible mappings i.e. bijections.

Let's build a simple model for the simple example considered in the previous slide.

- we start from the simplest univariate gaussian distribution

$$u \sim \mathcal{N}(0,1) \qquad p(u) = \frac{e^{-\frac{u^2}{2}}}{\sqrt{2\pi}}$$

- we define a affine transformation (a forward pass)

$$x = \mu + \cancel{\varepsilon}^{u}\sigma \qquad x = \mu + u\sigma = T(u)$$

- and inverse (with constraint sigma > 0):

$$\varepsilon = (x-\mu)/\sigma \qquad u = T^{-1}(x) = \frac{(x-\mu)}{\sigma}$$

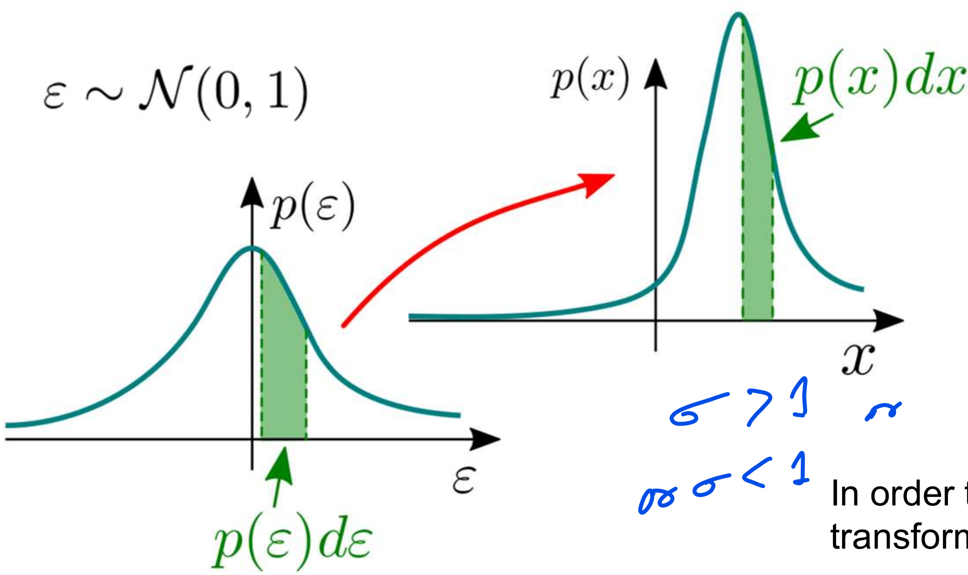
- we know $\cancel{p(\varepsilon)}$ $p(u)$ and we want to find $p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

$$p_x(x) = p_u(T^{-1}(x)) \frac{\partial T^{-1}(x)}{\partial x}$$

$$= \frac{e^{-\frac{(x-\mu)^2}{2\cdot\sigma^2}}}{\sqrt{2\pi}} \left[\frac{\partial}{\partial x} \frac{(x-\mu)}{\sigma}\right] =$$

Normalizing flows - transforming scalars

The volume change, but density must be preserved:



The necessary condition for this is:

$$p(\varepsilon)d\varepsilon = p(x)dx$$

The transformed density is then:

$$x = \mu + \varepsilon\sigma$$

change of volume

$$p(x) = p(\varepsilon)\left|\frac{dx}{d\varepsilon}\right|^{-1} = \frac{p(\varepsilon)}{\sigma}$$

In order to estimate the density at x we have to apply inverse transform:

$$\varepsilon = (x - \mu)/\sigma$$

Which will result in:

$$p_{\text{model}}(x;\theta) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

$D = 2$

$T_1(\vec{u}) = T_1(u_1, u_2)$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 2u_1 + 3u_2 \\ u_1 - u_2 \end{bmatrix}}_{T(u)} \qquad T^{-1}(\cancel{x}) = \begin{bmatrix} (x_1 + 3x_2)/5 \\ (x_1 - 2x_2)/5 \end{bmatrix}$$

$$= \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$p(u_1, u_2) \sim \mathcal{N}(0, I)$$

$$= \frac{e^{-\frac{u_1^2 - u_2^2}{2}}}{\sqrt{2\pi}}$$

$p(x_1, x_2)$?

Gaussian Copulas

Straightforward to Fit via Divergence of Choice

Example: *Maximum Likelihood Estimation*

$$D_{\mathrm{KL}}\left[p^*(\mathbf{x}) \;||\; p(\mathbf{x})\right]$$

Data generating process

Flow model

Straightforward to Fit via Divergence of Choice

$$\frac{1}{N}\sum_{i=1}^{N} \log p(x^i)$$

Example: *Maximum Likelihood Estimation*

$$D_{\mathrm{KL}}\left[p^*(\mathbf{x}) \mid\mid p(\mathbf{x})\right] = -\mathbb{E}_{p^*}\left[\log p(\mathbf{x})\right] + \text{const.}$$

Flow model

Data generating process

Straightforward to Fit via Divergence of Choice

Example: *Maximum Likelihood Estimation*

$$D_{\mathrm{KL}}\left[p^*(\mathbf{x}) \;||\; p(\mathbf{x})\right] = -\mathbb{E}_{p^*}\left[\log p(\mathbf{x})\right] + \text{const.}$$

$$= -\mathbb{E}_{p^*}\left[\log p_{\mathrm{u}}(T^{-1}(\mathbf{x})) + \log\left|\det J_{T^{-1}}(\mathbf{x})\right|\right] + \text{const.}$$

Universal Representation

Can any distribution be represented as a flow?

*modest conditions apply

Universal Representation

Can any distribution be represented as a flow?

Yes! The intuition is given by inverse transform sampling...

$$u \sim \text{Uniform}(0, 1)$$

$$x = \boxed{\text{CDF}^{-1}}(u)$$

$T(u)$

Don't know this in practice.

*modest conditions apply

Key Properties for Tractability

Key Properties for Tractability

#1 Compositionality

#2 Link Between Chain Rule and Triangular Jacobians

#3 Free to Choose Directionality

Key Properties for Tractability

#1 Compositionality

#2 Link Between Chain Rule and Triangular Jacobians

#3 Free to Choose Directionality

#1 Compositionality

Transformations can be composed without violating invertibility:

$$T = T_K \circ \ldots \circ T_1$$

#1 Compositionality

Transformations can be composed without violating invertibility:

$$T = T_K \circ \ldots \circ T_1$$

The det-Jacobian decomposes locally over sub-flows:

$$\log |J_T(\mathbf{u})| = \log \prod_k \left| J_{T_k}(\mathbf{u}) \right| = \sum_k \log \left| J_{T_k}(\mathbf{u}) \right|$$

determinant

#1 Compositionality

Transformations can be composed without violating invertibility:

$$T = T_K \circ \ldots \circ T_1$$

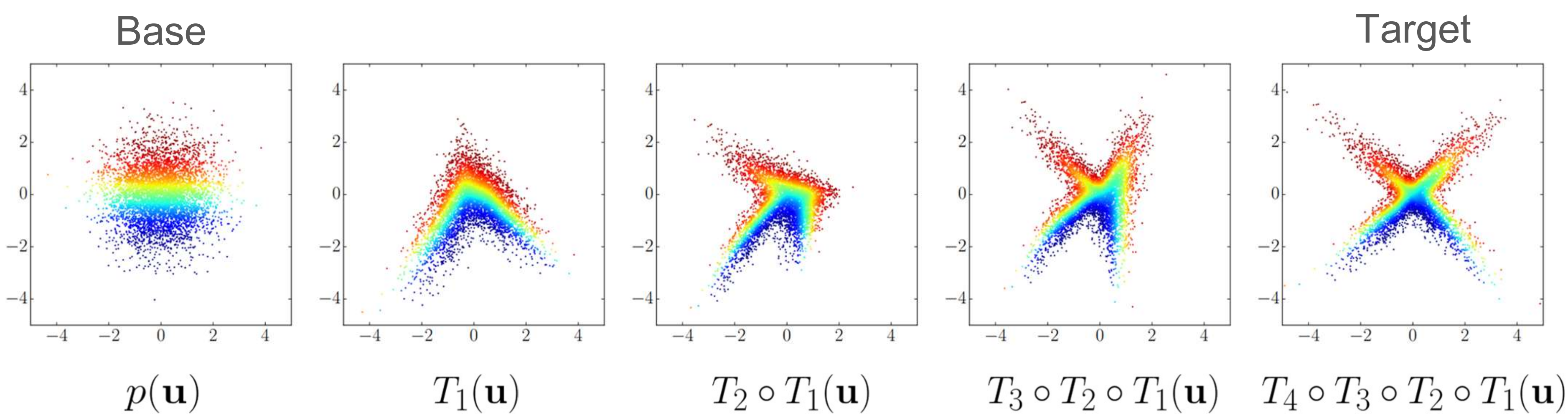
The det-Jacobian decomposes locally over sub-flows:

$$\log |J_T(\mathbf{u})| = \log \prod_k \left| J_{T_k}(\mathbf{u}) \right| = \sum_k \log \left| J_{T_k}(\mathbf{u}) \right|$$

Pay only *O*(*K*) cost for composition!

#1 Compositionality

Linear cost allows expressive transforms to be defined by composing many simple sub-flows...



Key Properties for Tractability

#1 Compositionality

#2 Link Between Chain Rule and Triangular Jacobians

#3 Free to Choose Directionality