

Deep Generative Models

Lecture 6

Roman Isachenko

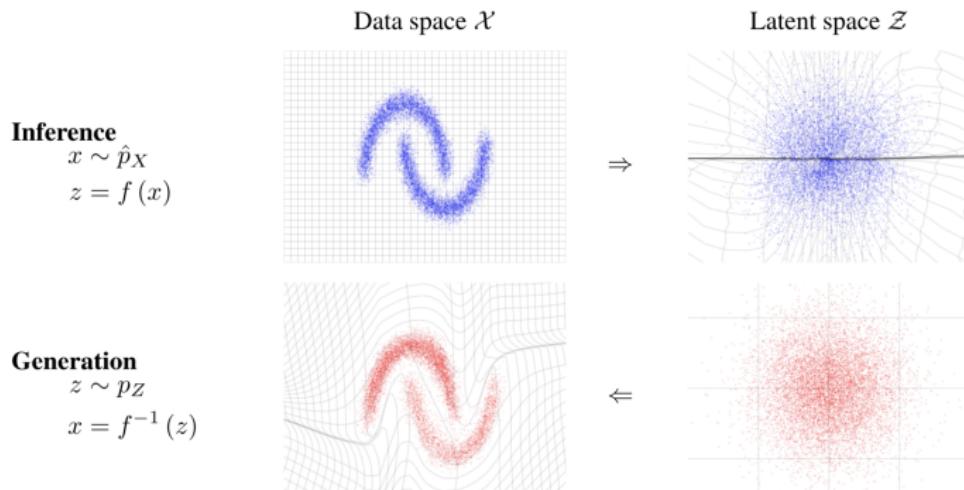


Autumn, 2022

Recap of previous lecture

Definition

Normalizing flow is a *differentiable, invertible* mapping from data \mathbf{x} to the noise \mathbf{z} .



Log likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f_K \circ \dots \circ f_1(\mathbf{x})) + \sum_{k=1}^K \log |\det(\mathbf{J}_{f_k})|,$$

Recap of previous lecture

Forward KL for flow model

$$\log p(\mathbf{x}|\theta) = \log p(f(\mathbf{x}, \theta)) + \log |\det(\mathbf{J}_f)|$$

Reverse KL for flow model

$$KL(p||\pi) = \mathbb{E}_{p(\mathbf{z})} [\log p(\mathbf{z}) - \log |\det(\mathbf{J}_g)| - \log \pi(g(\mathbf{z}, \theta))]$$

Flow KL duality

$$\arg \min_{\theta} KL(\pi(\mathbf{x})||p(\mathbf{x}|\theta)) = \arg \min_{\theta} KL(p(\mathbf{z}|\theta)||p(\mathbf{z})).$$

- ▶ $p(\mathbf{z})$ is a base distribution; $\pi(\mathbf{x})$ is a data distribution;
- ▶ $\mathbf{z} \sim p(\mathbf{z})$, $\mathbf{x} = g(\mathbf{z}, \theta)$, $\mathbf{x} \sim p(\mathbf{x}|\theta)$;
- ▶ $\mathbf{x} \sim \pi(\mathbf{x})$, $\mathbf{z} = f(\mathbf{x}, \theta)$, $\mathbf{z} \sim p(\mathbf{z}|\theta)$;

Recap of previous lecture

Flow log-likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(f(\mathbf{x}, \boldsymbol{\theta})) + \log |\det(\mathbf{J}_f)|$$

The main challenge is a determinant of the Jacobian.

Linear flows

$$\mathbf{z} = f(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}, \quad \boldsymbol{\theta} = \mathbf{W}, \quad \mathbf{J}_f = \mathbf{W}^T$$

- ▶ LU-decomposition

$$\mathbf{W} = \mathbf{P}\mathbf{L}\mathbf{U}.$$

- ▶ QR-decomposition

$$\mathbf{W} = \mathbf{Q}\mathbf{R}.$$

Decomposition should be done only once in the beginning. Next, we fit decomposed matrices ($\mathbf{P}/\mathbf{L}/\mathbf{U}$ or \mathbf{Q}/\mathbf{R}).

Kingma D. P., Dhariwal P. Glow: Generative Flow with Invertible 1x1 Convolutions, 2018

Hoogeboom E., et al. Emerging convolutions for generative normalizing flows, 2019

Outline

1. Gaussian autoregressive flows
2. Inverse gaussian autoregressive flows
3. Autoregressive flows
4. RealNVP: coupling layer

Outline

1. Gaussian autoregressive flows
2. Inverse gaussian autoregressive flows
3. Autoregressive flows
4. RealNVP: coupling layer

Gaussian autoregressive model

Consider an autoregressive model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^m p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}), \quad p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}) = \mathcal{N}(\mu_j(\mathbf{x}_{1:j-1}), \sigma_j^2(\mathbf{x}_{1:j-1})).$$

Sampling: reparametrization trick

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}), \quad z_j \sim \mathcal{N}(0, 1).$$

Inverse transform

$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

- ▶ We have an **invertible** and **differentiable** transformation from $p(\mathbf{z})$ to $p(\mathbf{x}|\boldsymbol{\theta})$.
- ▶ It is an autoregressive (AR) flow with the base distribution $p(\mathbf{z}) = \mathcal{N}(0, 1)!$
- ▶ How to derive a linear flow from this AR model?

Gaussian autoregressive flow

$$\mathbf{x} = g(\mathbf{z}, \theta) \quad \Rightarrow \quad x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \theta) \quad \Rightarrow \quad z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

Generation function $g(\mathbf{z}, \theta)$ is **sequential**.

Inference function $f(\mathbf{x}, \theta)$ is **not sequential**.

Forward KL for NF model

$$\log p(\mathbf{x}|\theta) = \log p(f(\mathbf{x}, \theta)) + \log \left| \det \left(\frac{\partial f(\mathbf{x}, \theta)}{\partial \mathbf{x}} \right) \right|$$

- ▶ We need to be able to compute $f(\mathbf{x}, \theta)$ and its Jacobian.
- ▶ We need to be able to compute the density $p(\mathbf{z})$.
- ▶ We don't need to think about computing the function $g(\mathbf{z}, \theta) = f^{-1}(\mathbf{z}, \theta)$ until we want to sample from the flow.

Outline

1. Gaussian autoregressive flows
2. Inverse gaussian autoregressive flows
3. Autoregressive flows
4. RealNVP: coupling layer

Inverse gaussian autoregressive flow (IAF)

Let us use the following reparametrization: $\tilde{\sigma} = \frac{1}{\sigma}$; $\tilde{\mu} = -\frac{\mu}{\sigma}$.

Gaussian autoregressive flow

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}) = (z_j - \tilde{\mu}_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{x}_{1:j-1})}$$

$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})} = \tilde{\sigma}_j(\mathbf{x}_{1:j-1}) \cdot x_j + \tilde{\mu}_j(\mathbf{x}_{1:j-1}).$$

Let just swap \mathbf{z} and \mathbf{x} .

Inverse gaussian autoregressive flow

$$\mathbf{x} = g(\mathbf{z}, \theta) \quad \Rightarrow \quad x_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1})$$

$$\mathbf{z} = f(\mathbf{x}, \theta) \quad \Rightarrow \quad z_j = (x_j - \tilde{\mu}_j(\mathbf{z}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{z}_{1:j-1})}.$$

Inverse gaussian autoregressive flow (IAF)

Gaussian autoregressive flow: $f(\mathbf{x}, \theta)$

$$x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}).$$

Inverse transform: $g(\mathbf{z}, \theta)$

$$z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})};$$

$$z_j = \tilde{\sigma}_j(\mathbf{x}_{1:j-1}) \cdot x_j + \tilde{\mu}_j(\mathbf{x}_{1:j-1}).$$

Inverse gaussian autoregressive flow:
 $f(\mathbf{x}, \theta)$

$$x_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1}).$$

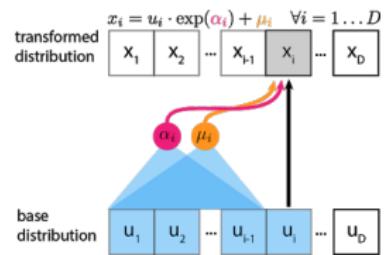
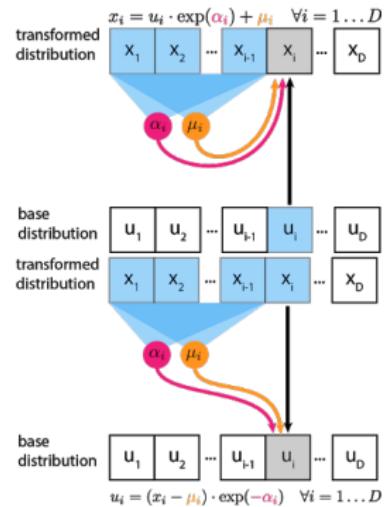


image credit: <https://blog.evjang.com/2018/01/nf2.html>

Gaussian autoregressive NF

Forward and inverse transforms in gaussian AR NF

$$\mathbf{x} = g(\mathbf{z}, \theta) \Rightarrow x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \theta) \Rightarrow z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

- ▶ Sampling is sequential.
- ▶ Density estimation is parallel.

Forward and inverse transforms in inverse gaussian AR NF

$$\mathbf{x} = g(\mathbf{z}, \theta) \Rightarrow x_j = \tilde{\sigma}_j(\mathbf{z}_{1:j-1}) \cdot z_j + \tilde{\mu}_j(\mathbf{z}_{1:j-1}).$$

$$\mathbf{z} = f(\mathbf{x}, \theta) \Rightarrow z_j = (x_j - \tilde{\mu}_j(\mathbf{z}_{1:j-1})) \cdot \frac{1}{\tilde{\sigma}_j(\mathbf{z}_{1:j-1})}.$$

- ▶ Sampling is parallel.
- ▶ Density estimation is sequential.

Outline

1. Gaussian autoregressive flows
2. Inverse gaussian autoregressive flows
3. Autoregressive flows
4. RealNVP: coupling layer

Autoregressive flows

$$x_j = \tau(z_j, c(\mathbf{z}_{1:j-1})) \Leftrightarrow z_j = \tau^{-1}(x_j, c(\mathbf{z}_{1:j-1}))$$

- ▶ $\tau(\cdot, \cdot)$ – coupling law (invertible by first argument, differentiable).
- ▶ $c(\cdot)$ – coupling function (do not need to be invertible, could be neural network).

Coupling law $\tau(\cdot, \cdot)$

- ▶ $\tau(x, c) = x + c$ – additive;
- ▶ $\tau(x, c) = x \odot c_1 + c_2$ – affine.

What is the Jacobian for the additive/affine coupling law?

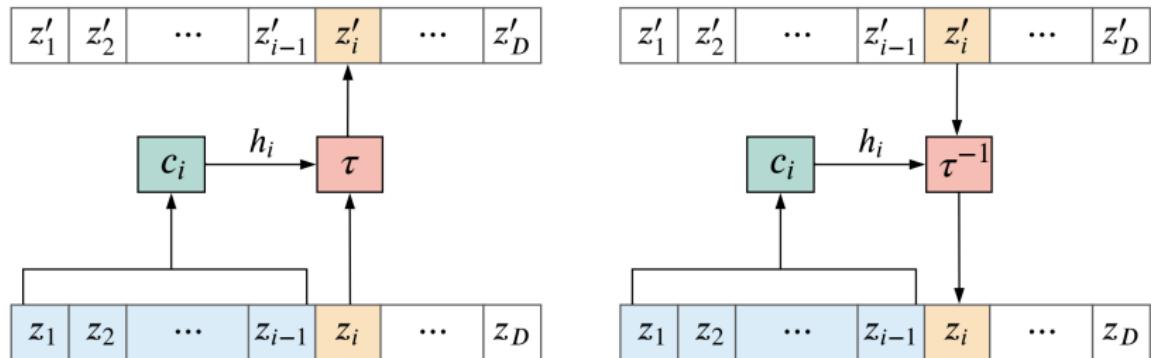
Jacobian

$$\det \left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}} \right) = \prod_{j=1}^m \frac{\partial x_j}{\partial z_j} = \prod_{j=1}^m \frac{\partial \tau(z_j, c(\mathbf{z}_{1:j-1}))}{\partial z_j}$$

Autoregressive flows

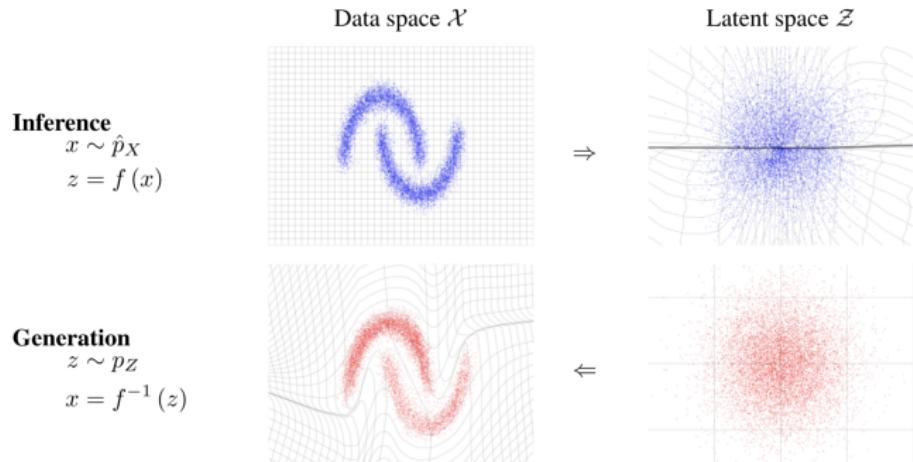
Forward and inverse transforms

$$x_j = \tau(z_j, c(\mathbf{z}_{1:j-1})) \Leftrightarrow z_j = \tau^{-1}(x_j, c(\mathbf{z}_{1:j-1}))$$



- ▶ Forward transform is **not sequential**.
- ▶ Inverse transform is **sequential**.

Autoregressive flows



- ▶ AF performs parallel inference that is useful for density estimation tasks (forward KL or MLE).
- ▶ IAF performs parallel generation that is useful for optimization of reverse KL.

Outline

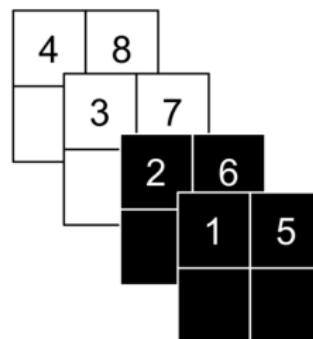
1. Gaussian autoregressive flows
2. Inverse gaussian autoregressive flows
3. Autoregressive flows
4. RealNVP: coupling layer

RealNVP

Coupling layer

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \tau(\mathbf{x}_{d:m}, c(\mathbf{x}_{1:d})); \end{cases} \Leftrightarrow \begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \tau^{-1}(\mathbf{z}_{d:m}, c(\mathbf{z}_{1:d})). \end{cases}$$

Image partitioning



Checkerboard ordering uses masking, channelwise ordering uses splitting.

Affine coupling law

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \mathbf{x}_{d:m} \odot c_1(\mathbf{x}_{1:d}, \theta) + c_2(\mathbf{x}_{1:d}, \theta). \end{cases}$$

$$\begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = (\mathbf{z}_{d:m} - c_2(\mathbf{z}_{1:d}, \theta)) \cdot \frac{1}{c_1(\mathbf{z}_{1:d}, \theta)}. \end{cases}$$

Jacobian

$$\det \left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \right) = \det \begin{pmatrix} \mathbf{I}_d & 0_{d \times m-d} \\ \frac{\partial \mathbf{z}_{d:m}}{\partial \mathbf{x}_{1:d}} & \frac{\partial \mathbf{z}_{d:m}}{\partial \mathbf{x}_{d:m}} \end{pmatrix} = \prod_{j=1}^{m-d} c_1(\mathbf{x}_{1:d}, \theta)_j.$$

Non-Volume Preserving (the determinant of Jacobian $\neq 1$).

AF vs IAF vs RealNVP

MADE/AF

$$\mathbf{x} = \sigma(\mathbf{z}) \odot \mathbf{z} + \boldsymbol{\mu}(\mathbf{x}).$$

Estimating the density $p(\mathbf{x}|\theta)$ - 1 pass, sampling - m passes.

IAF

$$\mathbf{x} = \tilde{\sigma}(\mathbf{z}) \odot \mathbf{z} + \tilde{\boldsymbol{\mu}}(\mathbf{z}).$$

Estimating the density $p(\mathbf{x}|\theta)$ - m passes, sampling - 1 pass.

RealINVP

$$\begin{cases} \mathbf{x}_{1:d} &= \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} &= \mathbf{z}_{d:m} \odot c_1(\mathbf{z}_{1:d}, \theta) + c_2(\mathbf{z}_{1:d}, \theta). \end{cases}$$

Estimating the density $p(\mathbf{x}|\theta)$ - 1 pass, sampling - 1 pass.

AF vs IAF vs RealINVP

RealINVP

$$\begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \mathbf{z}_{d:m} \odot c_1(\mathbf{z}_{1:d}, \boldsymbol{\theta}) + c_2(\mathbf{z}_{1:d}, \boldsymbol{\theta}). \end{cases}$$

- ▶ Calculating the density $p(\mathbf{x}|\boldsymbol{\theta})$ - 1 pass.
- ▶ Sampling - 1 pass.

RealINVP is a special case of AF and IAF:

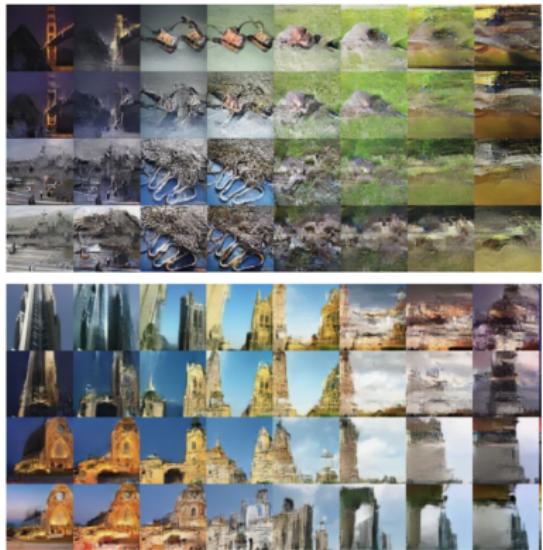
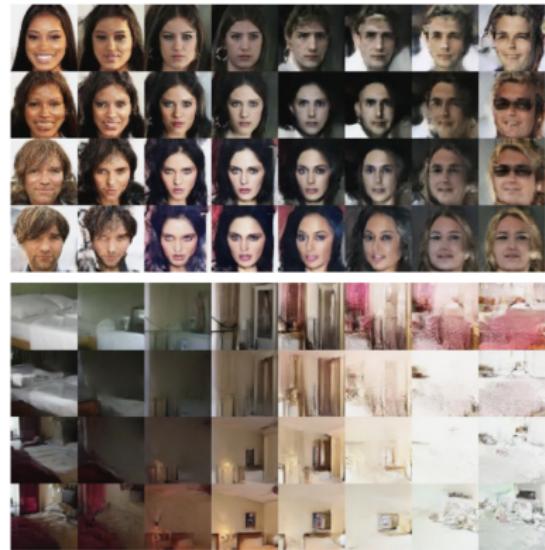
AF

$$\begin{cases} \mu_j = 0, \sigma_j = 1, j = 1, \dots, d; \\ \mu_j, \sigma_j - \text{functions of } \mathbf{x}_{1:d}, j = d + 1, \dots, m. \end{cases}$$

IAF

$$\begin{cases} \tilde{\mu}_j = 0, \tilde{\sigma}_j = 1, j = 1, \dots, d; \\ \tilde{\mu}_j, \tilde{\sigma}_j - \text{functions of } \mathbf{z}_{1:d}, j = d + 1, \dots, m. \end{cases}$$

RealNVP samples



Linear flows

RealNVP

$$\begin{cases} \mathbf{z}_{1:d} = \mathbf{x}_{1:d}; \\ \mathbf{z}_{d:m} = \tau(\mathbf{x}_{d:m}, c(\mathbf{x}_{1:d})); \end{cases} \Leftrightarrow \begin{cases} \mathbf{x}_{1:d} = \mathbf{z}_{1:d}; \\ \mathbf{x}_{d:m} = \tau^{-1}(\mathbf{z}_{d:m}, c(\mathbf{z}_{1:d})). \end{cases}$$

- ▶ First step is a **split** operator which decouples a variable into 2 subparts: \mathbf{x}_1 and \mathbf{x}_2 (usually channel-wise).
- ▶ We should **permute** components between different layers.

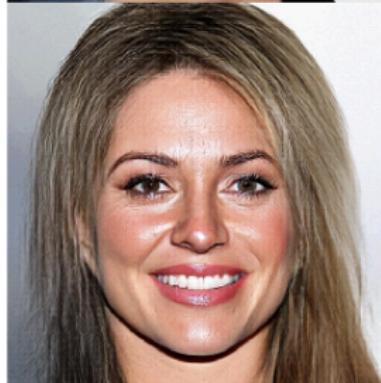
$$\mathbf{z} = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}$$

In general, we need $O(m^3)$ to invert matrix.

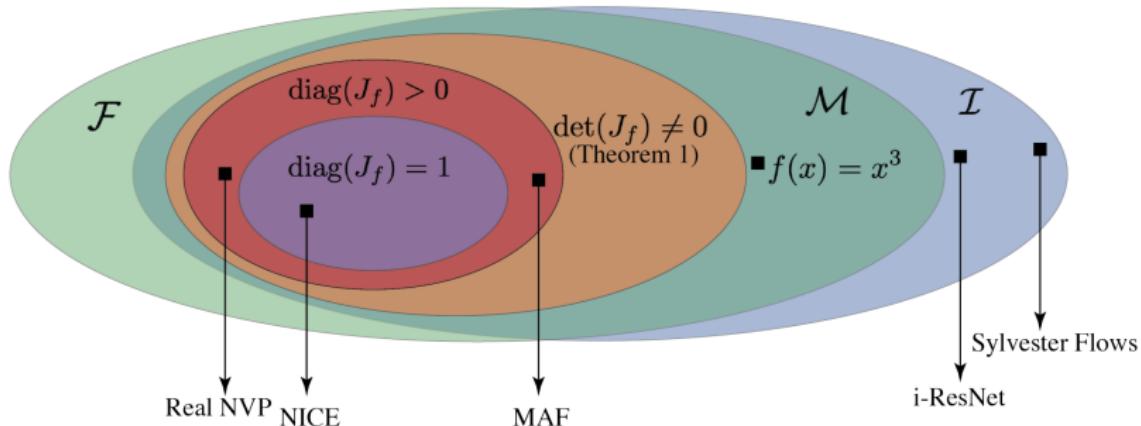
Invertibility

- ▶ Diagonal matrix $O(m)$.
- ▶ Triangular matrix $O(m^2)$.
- ▶ It is impossible to parametrize all invertible matrices.

Glow samples



Venn diagram for Normalizing flows



- ▶ \mathcal{I} – invertible functions.
- ▶ \mathcal{F} – continuously differentiable functions whose Jacobian is lower triangular.
- ▶ \mathcal{M} – invertible functions from \mathcal{F} .

Summary

- ▶ Gaussian autoregressive flow is an autoregressive model with triangular Jacobian.
- ▶ Inverse gaussian autoregressive flow generate new samples fast, but the inference is slow.
- ▶ AF/IAF is a special case of autoregressive flows.
- ▶ The RealNVP is an effective type of flow (special case of AR flows) that uses coupling layer.