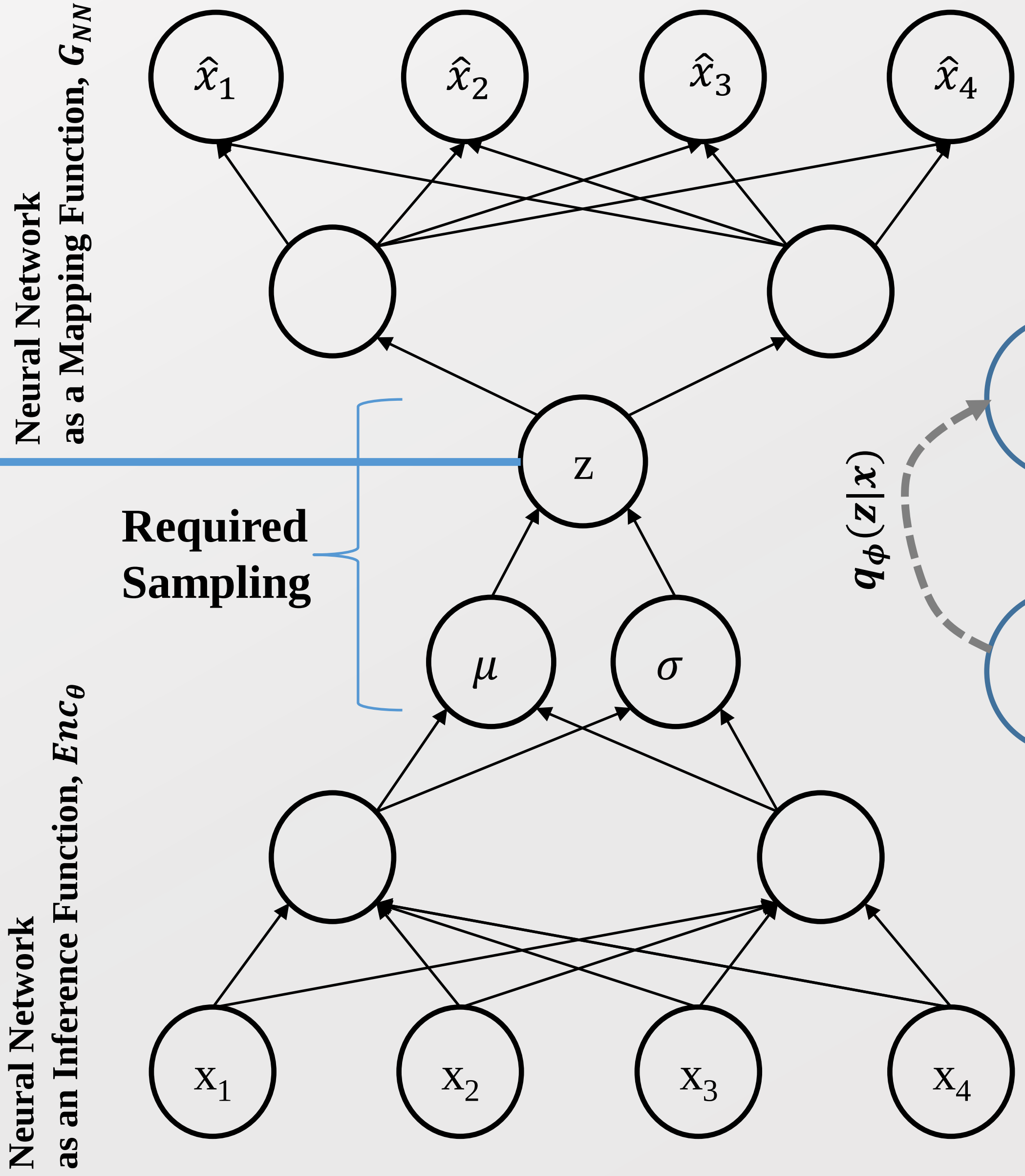
Normalizing Flow

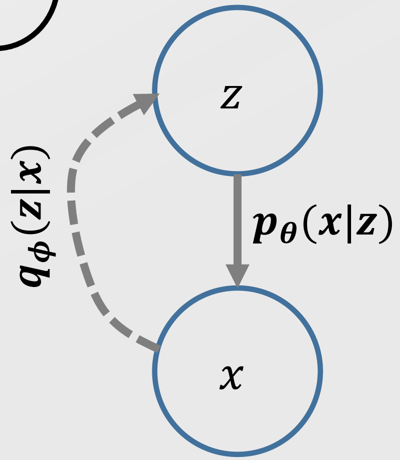
Il-Chul Moon
Department of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

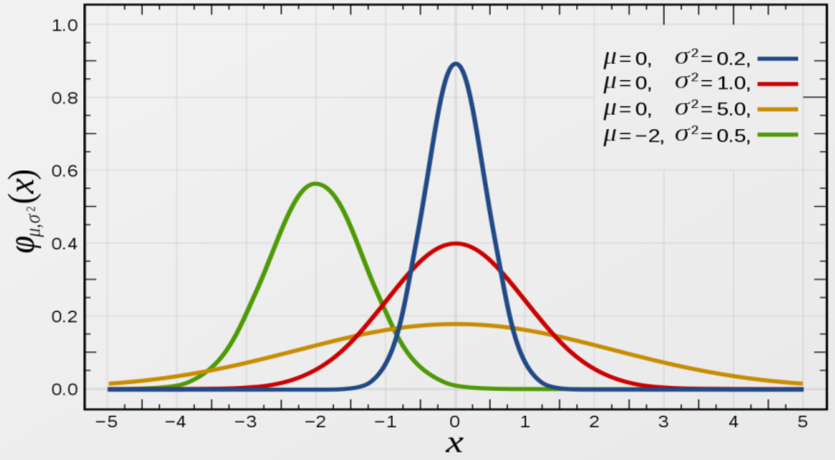
This slide is heavily influenced and adopted from the tutorial by Byeonghu Na, AAILab, KAIST.

Detour: Explicit Deep Generative Model (VAE)

- Generative model + Neural network
  - Neural network is used for the inference of the distribution parameters
- $x_{data} \sim P_{data}, x_{sample} = \hat{x} \sim P_{gen}$
- $z \sim N(Enc_{\mu}(x_{data}), Enc_{\sigma}(x_{data}))$
- Minimizing
  - $Error(x, \hat{x}) = Error(x, G(z))$
  - with some regularizations
    - Prior on Z with $N(0, I)$



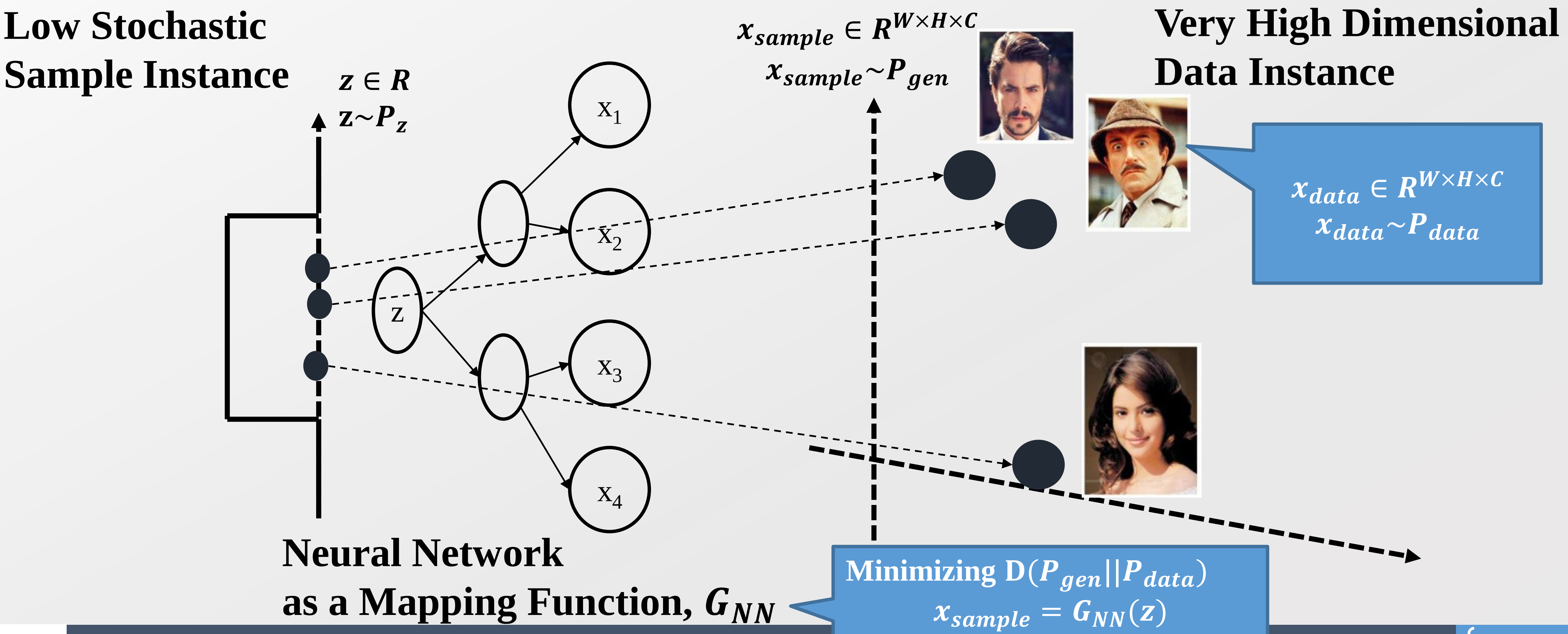




Detour: Implicit Deep Generative Model (GAN)

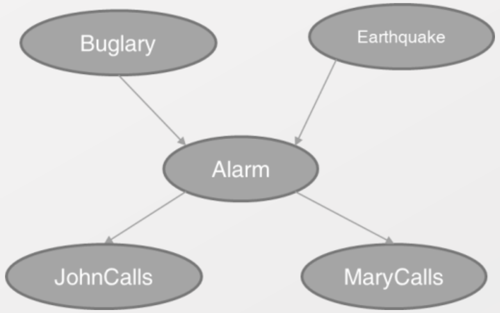
Function Approximation
on
Implicit Probability
Distribution of P_{gen}

- Generative model + Neural network
  - Neural network is used for the specification of the density model

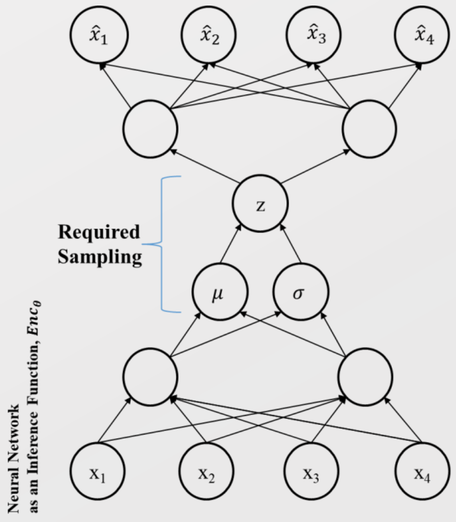


Some Fundamental Questions on Probabilistic Model

- Probabilistic Model, or Generative Models
  - Fundamental asepcts
  - Generation (==Sampling, Simulation…)
    - Able to generate a sample from Bayesian Network, VAE, GAN
  - Inference
    - Able to calculate a likelihood from Bayesian Network and VAE
    - Inable to calculate a likelihood from GAN
      - a.k.a. Likelihood-Free Model, Likelihood-Free Inference
- Distinctions from the types of generative models
  - Given $x_j \sim p_{data}, j = 1,2, \dots, |\mathcal{D}|$
  1) The flexibility of p_θ parameterized by $\theta \in \mathcal{M}$
  2) The property of $d(p_{data}, p_\theta)$
  3) The optimization of $\min_{\theta \in \mathcal{M}} d(p_{data}, p_\theta)$



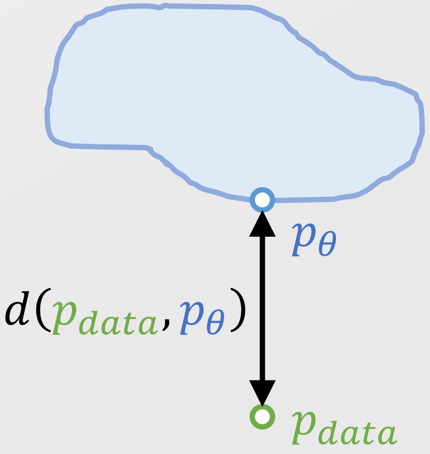
Bayesian Network



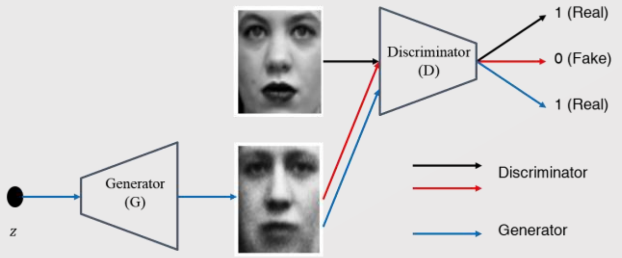
VAE

Model Family

$\theta \in \mathcal{M}$



$\min_{\theta \in \mathcal{M}} d(p_{data}, p_\theta)$



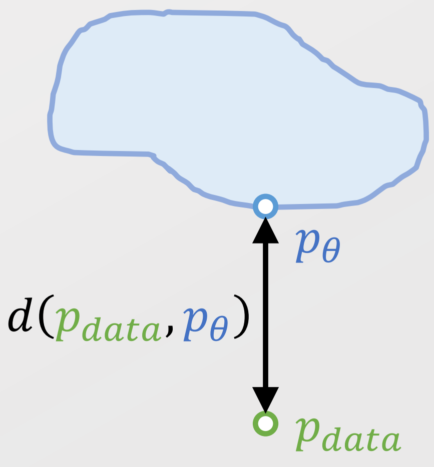
GAN

TRANSFORMATION OF BASIS FROM DATA SPACE TO LATENT SPACE

Approximation Nature of VAE and GAN

- VAE and GAN are all based upon the approximated distance/divergence
  - VAE optimizes the ELBO
    - $\ln P(E) \geq E_{Q(H|E)} \ln P(E|H) - KL(Q(H|E) \parallel P(H))$
    - Optimization on ELBO does not guarantee the optimization of $\ln P(E)$
      - Rather, only guarantees the lower bound optimization, itself
    - Problem of approximation on $\min_{\theta \in \mathcal{M}} d(p_{data}, p_\theta)$
  - GAN optimizes the minimax of the classification error
    - $\min_G \left\{ \max_D \left\{ E_{x \sim p_{data(x)}} [logD(x)] + E_{z \sim p_{z(z)}} [\log(1 - D(G(z)))] \right\} \right\}$
    - Minimization on JS divergence is only feasible when D becomes the optimal Discriminator
      - However, it is very difficult to achive in reality
    - Problem of approximation on $d(p_{data}, p_\theta)$
- Then, the question becomes whether the exact inference is feasible, or not
  - Flow is a family of $\mathcal{M}$ that enables the optimization on the exact inference
    - $\log p_\theta(x) = \log p_z(f_\theta(x)) + \log \left| \frac{\partial f_\theta(x)}{\partial x} \right|$

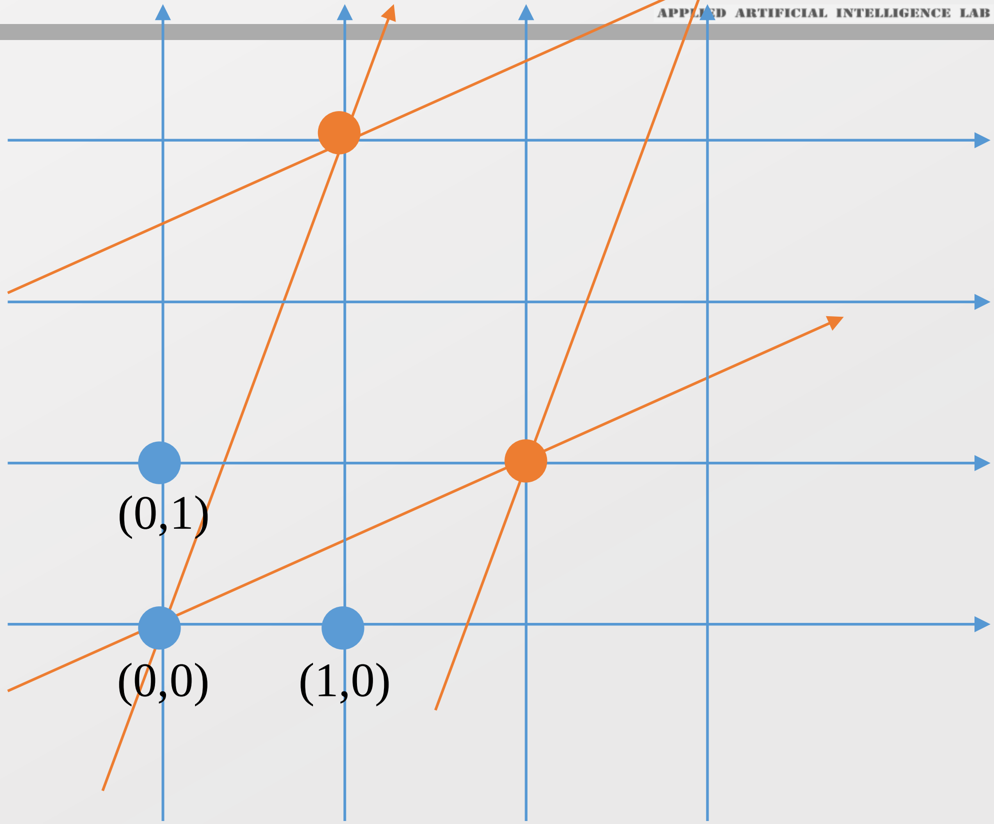
Model Family
$\theta \in \mathcal{M}$



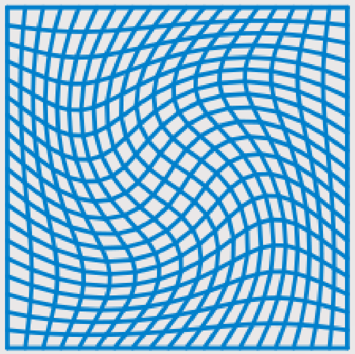
$\min_{\theta \in \mathcal{M}} d(p_{data}, p_\theta)$

Detour: Transformation in Linear Algebra

- Imagine a linear transformation
  - We can transform a basis through matrix multiplication
  - $\begin{bmatrix} x_{t+1}^1 \\ x_{t+1}^2 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} x_t^1 \\ x_t^2 \end{bmatrix} \rightarrow \mathrm{x}_{t+1} = T\mathrm{x}_t$
    - The meaning of the transformation matrix
      - Diagonal : scaling of the basis
      - Off-diagonal : correlation of the basis
      - $Det(T)$: area multiplier after the transformation
        - What-if $Det(T) < 0 \rightarrow$ The basis becomes directed inversely (flipping)
    - Some matrix enables the transformation in the degenerative manner
      - $Det(T) = 0$
        - All points before the transformation are collapsed into a single point of $(0,0)$



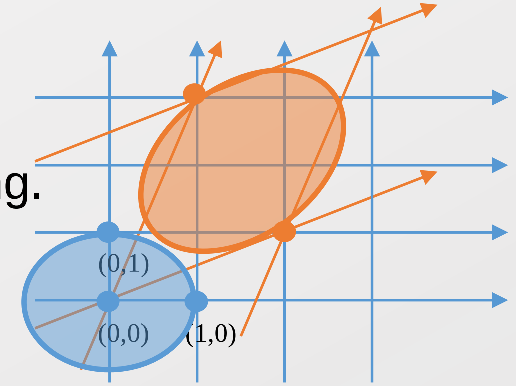
- Linear transformation
  - Can be changed as $\mathrm{x}_4 = (T_4 T_3 T_2 T_1)\mathrm{x}_0 = T_4\left(T_3\left(T_2(T_1\mathrm{x}_0)\right)\right) = T_4\left(T_3(T_2\mathrm{x}_1)\right)$
  - Can be inversed if A is not degenerative as $\mathrm{x}_0 = (T_1^{-1}T_2^{-1}T_3^{-1}T_4^{-1})\mathrm{x}_4$
- Diffeomorphism
  - Given two manifolds of M and N, a differentiable map $f: M \rightarrow N$ is a diffeomorphism
    - if f is a bijective
    - if $f^{-1}: N \rightarrow M$ is differentiable



https://en.wikipedia.org/wiki/Diffeomorphism

Detour: Change of Variable in Calculus

- Change of variable in a single dimension
  - Condition
    - Let f and h be continuous functions from $[a, b]$ to R, and assume h strictly increasing.
    - Let α be the inverse function of h as $y = h(x), x = \alpha(y)$
    - $a \le x \le b, h(a) \le y \le h(b)$
  - Then, $\int_a^b f(x)dx = \int_{h(a)}^{h(b)} f(\alpha(y))d\alpha(y)$
  - In statistics, $p_y(y) = p_x(\alpha(y))\alpha'(y)$
- Change of variable in its multivariate version
  - $\mathrm{x}_{t+1} = T\mathrm{x}_t \rightarrow \Delta A \approx J(\mathrm{x}_t)\Delta \mathrm{x}_{t,1} \dots \Delta \mathrm{x}_{t,m}$
    - $\mathrm{x}_.$: a m-dimensional vector
    - $J(\mathrm{x}_t)$: a Jacobian matrix of T^{-1}
    - A : an area in x_{t+1} space
  - $\iint_{X_t} f(\mathrm{x}_t)d\mathrm{x}_{t,1} \dots d\mathrm{x}_{t,m} = \iint_{X_{t+1}} f(T(\mathrm{x}_t))|\det J(\mathrm{x}_t)|d\mathrm{x}_{t+1,1} \dots d\mathrm{x}_{t+1,m}$
  - In statistics, $p_{\mathrm{x}_t}(\mathrm{x}_t) = p_{\mathrm{x}_{t+1}}(T(\mathrm{x}_t))|\det J(\mathrm{x}_t)| = p_{\mathrm{x}_{t+1}}(T_\theta(\mathrm{x}_t))|\det \frac{\partial T_\theta(\mathrm{x}_t)}{\partial \mathrm{x}_t}|$



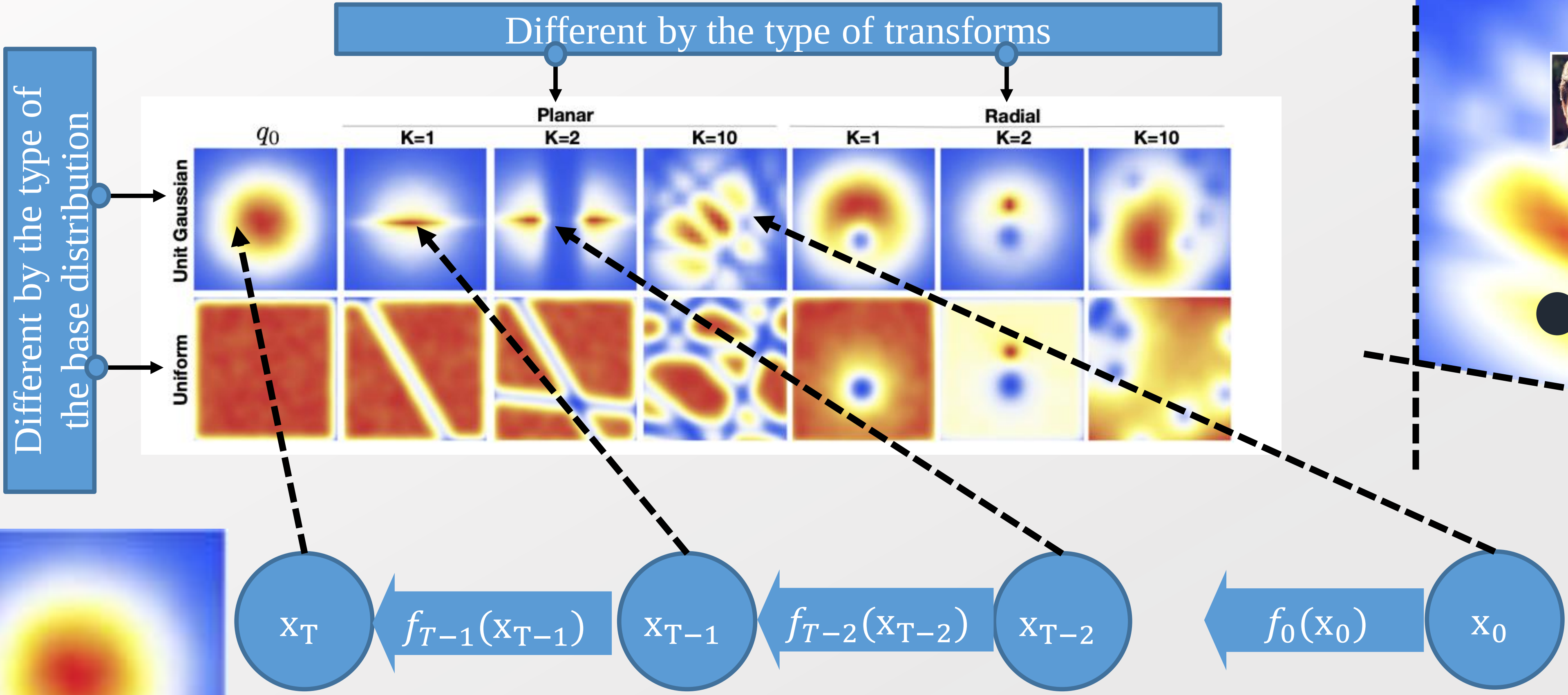
$$Jacobian\ Martrix = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

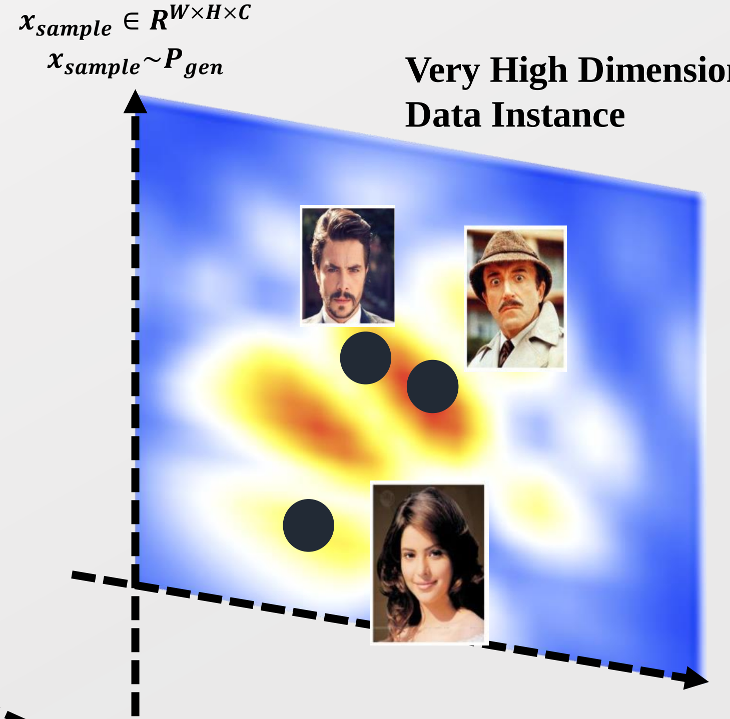
assuming $f: R^n \rightarrow R^m$

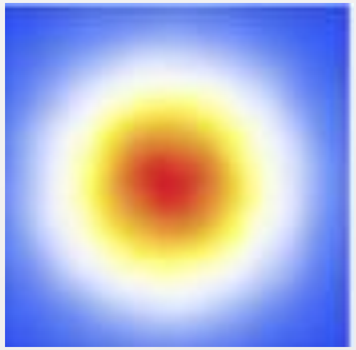
$$Hessian\ Martrix = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_2 x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \frac{\partial^2 f}{\partial x_n x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

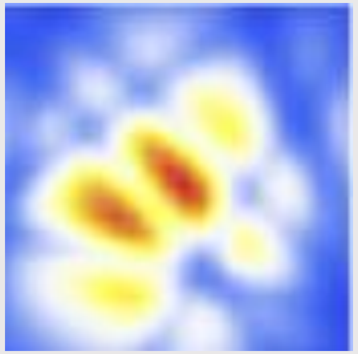
Flow Model

- Generative model + Neural network
  - Neural network is used for the transformation of the distributions





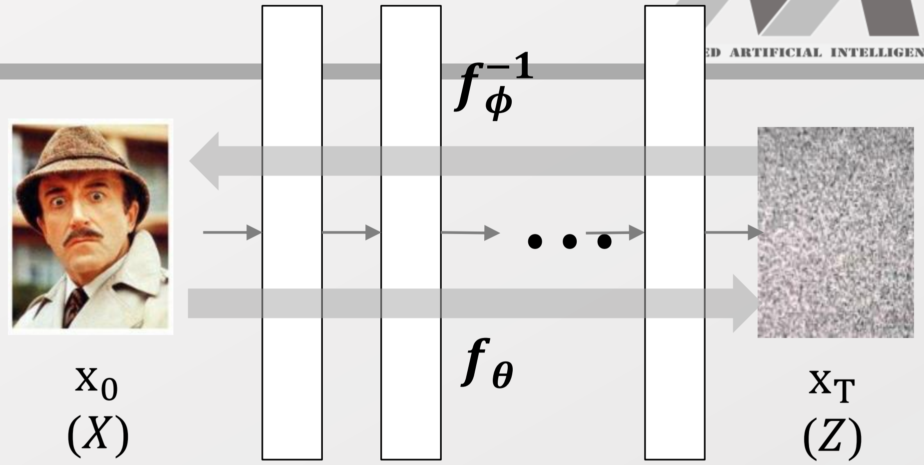




Change of Variables : If $f_\theta : R^M \to R^M$ is a bijective and differentiable function,

$$p_X(\mathrm{x}) = p_Z\big(f_\theta(\mathrm{x})\big)\left|\det \frac{\partial f_\theta(\mathrm{x})}{\partial \mathrm{x}}\right|$$

Flow Model as a Formula



- Change of variable
  - $\mathrm{x}_{t+1} = T\mathrm{x}_t,\ p_{\mathrm{x}_t}(\mathrm{x}_t) = p_{\mathrm{x}_{t+1}}\left(T_\theta\ (\mathrm{x}_t)\right)|\det\frac{\partial T_\theta\ (\mathrm{x}_t)}{\partial \mathrm{x}_t}|$
- Chain of transformations
  - $\mathrm{x}_4 = (T_4T_3T_2T_1)\mathrm{x}_0 = T_4\left(T_3\left(T_2(T_1\mathrm{x}_0)\right)\right) = T_4\left(T_3(T_2\mathrm{x}_1)\right)$
- Flow $f: \mathcal{X} \to \mathcal{Z}$
  - An invertible differentiable function from x to z
  - Usually we denote x as data and z as noise.
- Sampling process is:
  - $z \sim p(z)$
  - $x = f_\theta^{-1}(z)$
- Objective function

$$\mathrm{argm}ax_\theta\ \mathbb{E}_{x\sim p_{data}}[\log p_\theta(x)]$$

  - How can we evaluate the marginal likelihood $p_\theta(x)$? → Using change-of-variables

$$\log p_X(x) = \log p_Z\left(f_\theta(x)\right) + \log\left|\frac{\partial f_\theta(x)}{\partial x}\right|$$

Composition of Flows

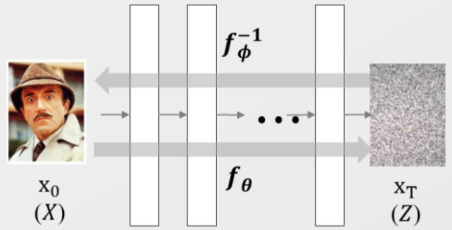
- Each transformation needs to be diffeomorphism
  - Difficult to create highly nonlinear behavior with a simple transformation
- Compose the multiple transformation as a chain
  - $f = f_K \circ f_{K-1} \circ \cdots \circ f_1$
    - Sampling can be $x = f^{-1}(z) = (f_1^{-1} \circ \cdots \circ f_{K-1}^{-1} \circ f_K^{-1})(z)$
  - Density evaluation
    - $p_X(x) = p_Z\big(f_\theta(x)\big)\left|\det\frac{\partial f_\theta(x)}{\partial x}\right|$
    - $\det\frac{\partial f(x)}{\partial x} = \det\prod_{i=1}^{K}\frac{\partial f_i}{\partial f_{i-1}} = \prod_{i=1}^{K}\det\frac{\partial f_i}{\partial f_{i-1}}$
  - Training with log-likelihood

$$\max_\theta\left\{\log p_Z\big(f_\theta(x)\big) + \sum_{i=1}^{K}\log\left|\det\frac{\partial f_i}{\partial f_{i-1}}\right|\right\}$$

- Then, our question becomes what could be the choice of f
  - Following the diffeomorphism
  - Generating a flexible and domain-adapted transformation, i.e. the locality on image

STRUCTURE OF NORMALIZING FLOW

Elementwise Flows



- Elementwise Flows

$$f(x) = f\big((x_1, \dots, x_D)\big) = \big(f_1(x_1), \dots, f_D(x_D)\big) = z$$

  - Each dimension takes only the same dimension value as the transformation input
    - x_D: D-th feature dimension of an input vector of x
    - $z_d = f_d(x_d) = f(x_d; \theta_d)$
    - θ_d : Each dimension's transformation is parameterized per each dimension with the shared structure
    - We will treat z as the transformed x
  - f is invertible if all $f^{(i)}$ be invertible.
    - Analytic Inverse : $f^{-1}\big((z_1, \dots, z_D)\big) = \big(f_1^{-1}(z_1), \dots, f_D^{-1}(z_D)\big)$
  - Jacobian matrix
    - $J_f(x) = \mathrm{diag}\big(f_1'(x_1), \dots, f_D'(x_D)\big)$ ➔ $\det\big(J_f(x)\big) = \prod_{i=1}^{D} f_i'(x_i)$

$$Jacobian\ Martrix = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

- Given the objective function of $\max_\theta \left\{\log p_Z\big(f_\theta(x)\big) + \sum_{i=1}^{K} \log \left|\det \frac{\partial f_i}{\partial f_{i-1}}\right|\right\}$
  - No mixing of variables
    - i.e. images have highly localized correlations → Problem in modeling the density of the joint space
  - But from these, we know that it is okay to use some activation functions or batch normalization in composition of flows.
    - Computation cost of Inverse: $O(D)$
    - Computational cost of determinant: $O(D)$
  - Equivalent to only applying the activation functions without any cross links between inputs and outputs

Linear Flows

- Linear Flows (or Affine Flows)

$$f(x) = f\big((x_1, \dots, x_D)\big) = Ax + b = z$$

$$Jacobian\ Martrix = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

  - f is invertible if A is invertible.
    - Analytic inverse : $f^{-1}(z) = A^{-1}(z-b)$
  - Jacobian matrix
    - $J_f(x) = A$ ➔ $\det\big(J_f(x)\big) = \det A$
- Given the objective function of $\max_\theta \left\{\log p_Z\big(f_\theta(x)\big) + \sum_{i=1}^{K} \log\left|\det \frac{\partial f_i}{\partial f_{i-1}}\right|\right\}$
  - No capability of nonlinear transformation
    - i.e. Let's assume $z \sim N(0, I)$, then $x \sim N(\mu, \Sigma)$: This nature will not change by composing the transformation
    - Why? This is a linear transformation without any activation
      - If we add an activation to this transformation, the inversion and the Jacobian will not be simple
  - High cost of calculation
    - Computation cost of Inverse: $O(D^3)$
    - Computational cost of determinant: $O(D^3)$
  - Equivalent to applying the fully connected neural network layer without activation

KAIST

Detour: Laplace Expansion

- The objective function of flow
  - $\max_{\theta}\left\{\log p_Z\big(f_\theta(x)\big) + \sum_{i=1}^{K}\log\left|\det\frac{\partial f_i}{\partial f_{i-1}}\right|\right\}$
    - The calculation of the determinant of the Jacobian matrix becomes the key
- Laplace expansion is an expression of determinant as the sum of small matrixes
  - Given the $n \times n$ matrix of A
    - Let's say $M_{i,j}$ is the determinant of the minor matrix which removes i and j-th row and column of A
    - Then, the determinant of A becomes

$$\det(A) = \sum_{j=1}^{n}(-1)^{i+j}A_{i,j}M_{i,j}$$

$$Jacobian\ Martrix = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

  - This suggests that the determinant calculation becomes simpler with a well positioned $A_{i,j} = 0$
    - Diagonal matrix and Triangle matrix

$$\det(A) = \prod_{i=1}^{n}A_{i,i}$$

$$Triangular\ Jacobian\ Martrix = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} = 0 & \cdots & \frac{\partial f_1}{\partial x_n} = 0 \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} = 0 \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} = 0 \end{bmatrix}$$

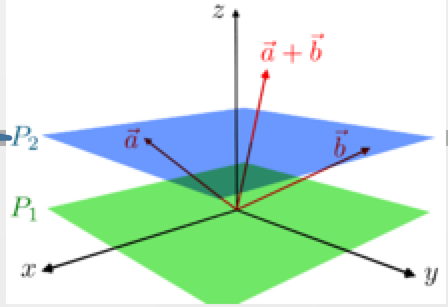
- Computation complexity
  - Laplace expansion takes $O(n!)$
  - LU decomposition takes $O(n^3)$

KAIST

Coupling Flows: Affine

https://en.wikipedia.org/wiki/Affine_space

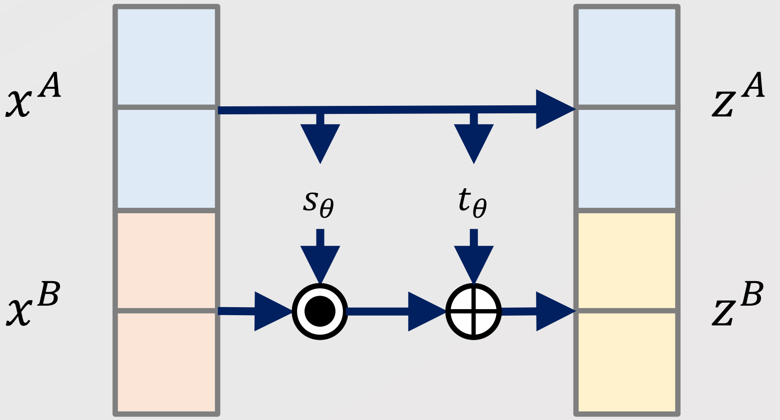
Affine subspace



- Only transform the affine subspace of x
- Affine Coupling Flows
  - Split variables into x^A and x^B → elementwise flows on x^B with the effect of x^A

$$f(x) = f\big((x_1, \dots, x_D)\big) = f([x^A; x^B]) = \big(x^A, x^B \cdot s_\theta(x^A) + t_\theta(x^A)\big) = z$$

    - In other words, $f = (f^A, f^B)$ where $z^A = f^A(x^A) = x^A$ and $z^B = f^B(x^B) = x^B \cdot s_\theta(x^A) + t_\theta(x^A)$
      - $z^A = x^A$: No transformation
      - $z^B = x^B \cdot s_\theta(x^A) + t_\theta(x^A)$
    - Utilizing the elementwise flow in x^B : Good characteristics in nonlinearity and computation complexity
  - f is invertible without significant restrictions of s_θ and t_θ
    - Analytic inverse
      - $x^A = (f^A)^{-1}(z^A) = z^A$
        $x^B = (f^B)^{-1}(z^B) = \{z^B - t_\theta(z^A)\}/s_\theta(z^A)$
  - Jacobian Matrix
    - $J_f(x) = \begin{bmatrix} \frac{\partial f^A}{\partial x^A} & \frac{\partial f^A}{\partial x^B} \\ \frac{\partial f^B}{\partial x^A} & \frac{\partial f^B}{\partial x^B} \end{bmatrix} = \begin{bmatrix} I & 0 \\ \frac{\partial f^B}{\partial x^A} & \mathrm{diag}\big(s(x^A)\big) \end{bmatrix}$ ➔ $\det J_f(x) = \prod_j s(x^A)_j$



Discussion on Affine Coupling Flow

- Affine Coupling Flows
  - Split variables into x^A and x^B → elementwise flows on x^B with the effect of x^A

$$f(x) = f\left((x_1, \dots, x_D)\right) = f([x^A; x^B]) = \left(x^A, x^B \cdot s_\theta(x^A) + t_\theta(x^A)\right) = z$$

- Given the objective function of $\max_\theta \left\{\log p_Z\left(f_\theta(x)\right) + \sum_{i=1}^K \log\left|\det \frac{\partial f_i}{\partial f_{i-1}}\right|\right\}$
  - Some capability of nonlinear transformation
    - $s_\theta(x^A)$ and $t_\theta(x^A)$: the source of the nonlinear nature
    - i.e. Let's assume $z \sim N(0, I)$, then $x \sim N(\mu, \Sigma)$: This nature will not change by composing the transformation
    - Why? This is a linear transformation without any activation
      - If we add an activation to this transformation, the inversion and the Jacobian will not be simple
  - Low cost of calculation
    - Computation cost of Inverse: $O(D)$
    - Computational cost of determinant: $O(D)$
  - Equivalent to applying the some selective links on neural network layer with some activations
  - Moreover, through the compositions of transformations, the split can be changed
    - Split variables into x^A and x^B at $f_1 \neq$ Split variables into x^A and x^B at f_2

Generalized Coupling Flow

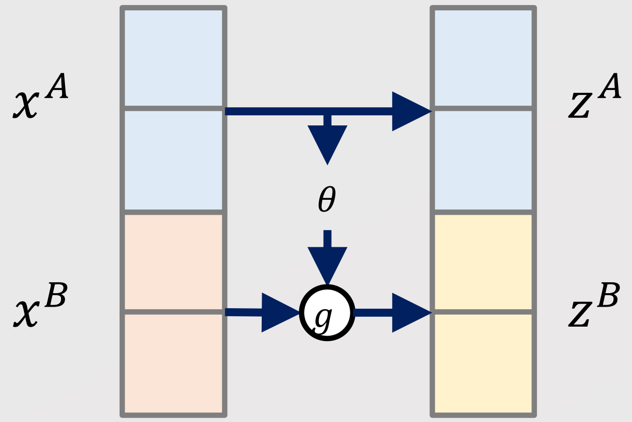
- Affine Coupling Flows
  - Split variables into x^A and x^B → elementwise flows on x^B with the effect of x^A

$$f(x) = f\big((x_1, \dots, x_D)\big) = f([x^A; x^B]) = \big(x^A, x^B \cdot s_\theta(x^A) + t_\theta(x^A)\big) = z$$

- Generalized Coupling Flows : general approach to construct non-linear flows

$$f(x) = f([x^A; x^B]) = \Big(x^A, g\big(x^B; \theta(x^A)\big)\Big) = z$$

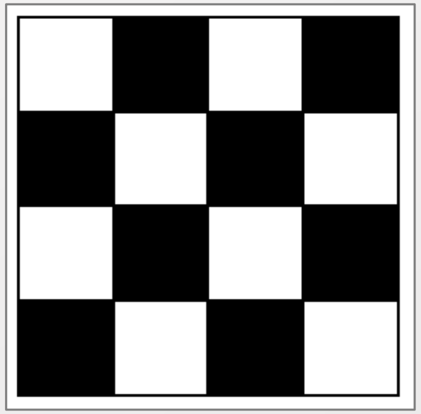
  - i.e. $f = (f^A, f^B)$ where $f^A(x^A) = x^A$ (identity), $f^B(x^B) = g\big(x^B; \theta(x^A)\big)$
    - $\theta(x^A)$: the parameterization dynamically determined by x^A
  - f is invertible if g is invertible. (not θ!)
    - Analytic inverse
      - $f^{-1}(z) = \big(z^A, g^{-1}\big(z^B; \theta(z^A)\big)\big)$
  - Jacobian Matrix
    - $J_f(x) = \begin{bmatrix} \frac{\partial f^A}{\partial x^A} & \frac{\partial f^A}{\partial x^B} \\ \frac{\partial f^B}{\partial x^A} & \frac{\partial f^B}{\partial x^B} \end{bmatrix} = \begin{bmatrix} I & 0 \\ \frac{\partial g}{\partial x^A} & J_g(x^B) \end{bmatrix}$ → $\det J_f(x) = \det J_g(x^B) = \prod_j J_g(x^B)_j$



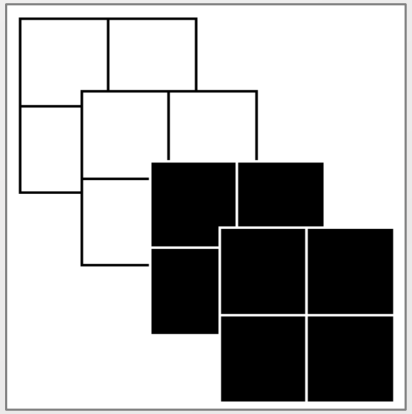
- g can be any function to be applied in the elementwise manner
  - Linear function, Spline function, and inverse CDF….

Diverse Suggestions on Variable Split

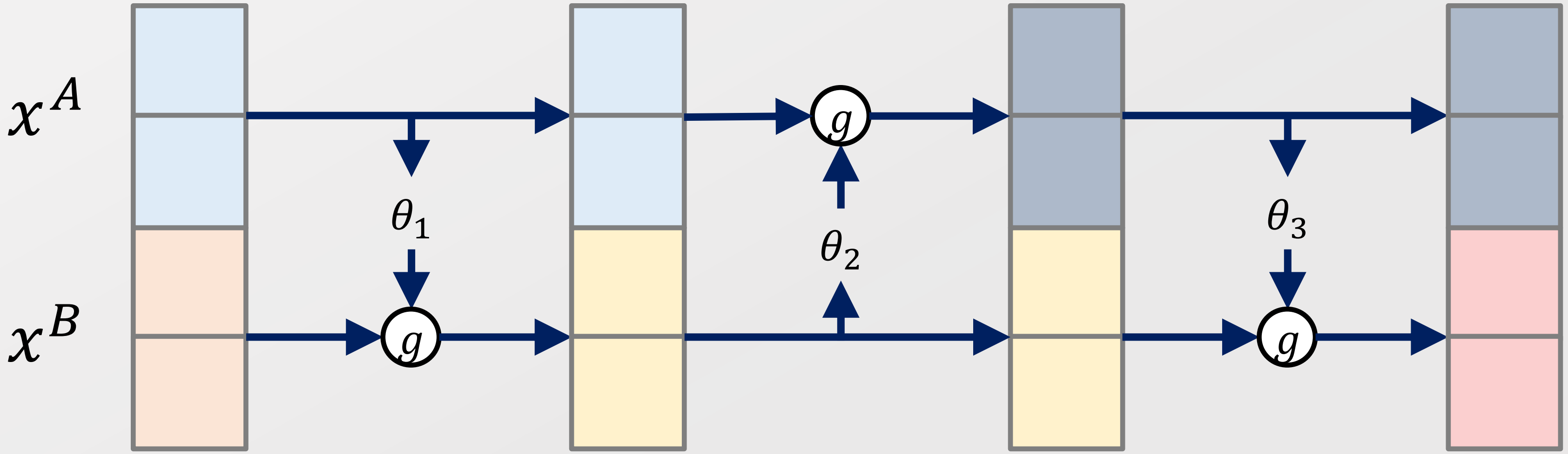
- Coupling flow
  - Split variables into x^A and x^B → elementwise flows on x^B with the effect of x^A
  - Setting x^A and x^B depends upon the domain of application
    - Images? Sequences?....
  - Split needs to change over the composition of the transformation



Spatial Checkerboard Pattern



Channel-wise Masking

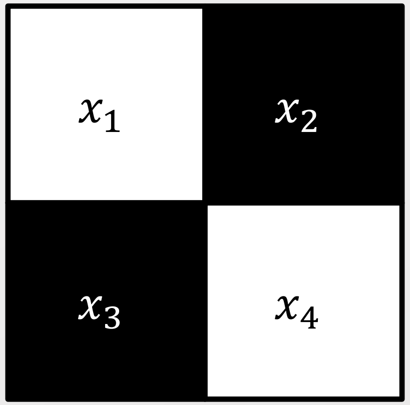


Interpretation of Coupling Flow from Bayesian Perspective

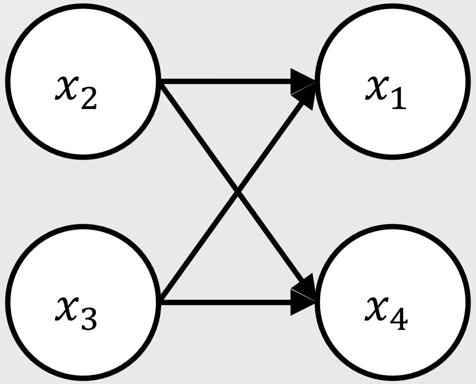
- Coupling Flows (a.k.a. RealNVP) : general approach to construct non-linear flows

$$f(x) = f([x^A; x^B]) = \left(x^A, g\left(x^B; \theta(x^A)\right)\right) = z$$

- Interpretation from the probabilistic modeling perspective
  - Flow of each variable becomes the Bayesian network links
    - $f(x_2) = x_2$
    - $f(x_3) = x_3$
    - $f(x_1) = x_1 s_1(x_2, x_3) + t_1(x_2, x_3)$
    - $f(x_4) = x_4 s_4(x_2, x_3) + t_4(x_2, x_3)$
  - Is same as
    - $f(x_2) = x_2$
    - $f(x_3) = x_3$
    - $f(x_1) = x_1 s_1\left(\mathrm{parent}(x_1)\right) + t_1\left(\mathrm{parent}(x_1)\right)$
    - $f(x_4) = x_4 s_4\left(\mathrm{parent}(x_4)\right) + t_4\left(\mathrm{parent}(x_4)\right)$
  - This creates a DAG, which is the key characteristics of Bayesian network
    - However, very typical and stereotype
    - Then, is it possible to create a more interesting structure by Bayesian network?
      - with some inspiration from the old graphical models?



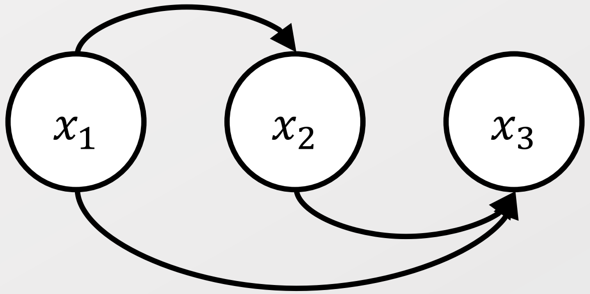
Variable Split



Bayesian Network

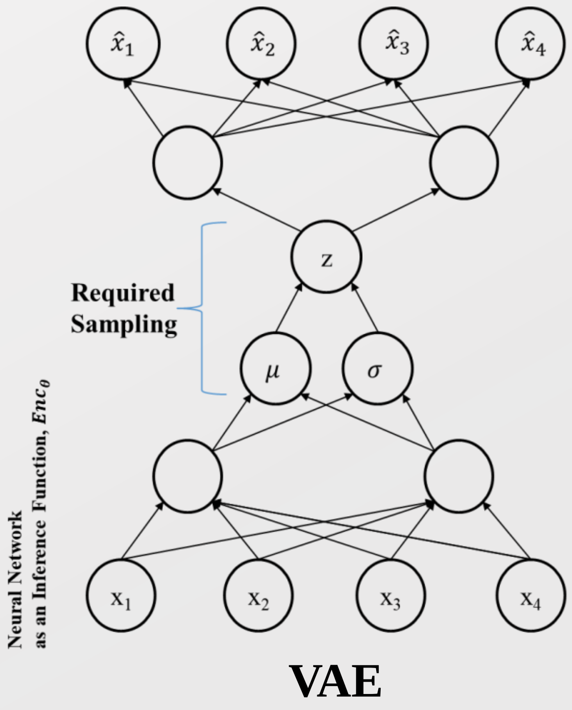
Autoregressive Flow

- Autoregressive Bayesian network structure
  - Density model
    - $x_1 \sim p_\theta(x_1)$
    - $x_2 \sim p_\theta(x_2|x_1)$
    - $x_3 \sim p_\theta(x_3|x_1, x_2)$
  - Generation process : sampling x given z : $p(x|z)$
    - $x_1 = f_\theta^{-1}(z_1)$
    - $x_2 = f_\theta^{-1}(z_2; x_1)$
    - $x_3 = f_\theta^{-1}(z_3; x_1, x_2)$
  - Inference process : learning the distribution of z given x : $q(z|x)$
    - $z_1 = f_\theta(x_1)$
    - $z_2 = f_\theta(x_2; x_1)$
    - $z_3 = f_\theta(x_3; x_1, x_2)$
- Jacobian matrix : triangular matrix because of independence between z_t and $x_{(t+1):T}$
- Computation complexity
  - Let's consider the order
    - Inference process : all inputs are given : Can be parallel → Fast
    - Geneation process : some inputs are given : Needs to be sequential → Slow



Inverse Autoregressive Flow and ELBO Interpretation

- Autoregressive flow
  - Generation process : sampling x given z : $p(x|z)$
    - $x_1 = f_\theta^{-1}(z_1), x_2 = f_\theta^{-1}(z_2; x_1), x_3 = f_\theta^{-1}(z_3; x_1, x_2)$
  - Inference process : learning the distribution of z given x : $q(z|x)$
    - $z_1 = f_\theta(x_1), z_2 = f_\theta(x_2; x_1), z_3 = f_\theta(x_3; x_1, x_2)$
- Inverse autoregressive flow
  - Generative process (Fast)
    - $x_1 = f_\theta^{-1}(z_1), x_2 = f_\theta^{-1}(z_2; z_1), x_3 = f_\theta^{-1}(z_3; z_1, z_2)$
  - Inference process (Slow)
    - $z_1 = f_\theta(x_1), z_2 = f_\theta(x_2; z_1), z_3 = f_\theta(x_3; z_1, z_2)$



- Interpretation by the latent inference algorithm, such as variational inference
  - $\log p(x) \geq \mathbb{E}_{z\sim q(z|x)}[\log p(x|z)] - D_{KL}[q(z|x)||p(z)] = \mathbb{E}_{z\sim q(z|x)}[\log p(x|z)] - \mathbb{E}_{z\sim q(z|x)}\left[\frac{q(z|x)}{p(z)}\right]$ $\approx E_Z\left[\log p(x|z) - \left(\frac{q(z|x)}{p(z)}\right)\right]$ where $z\sim q(z|x)$
    - Finding $z\sim q(z|x)$ → Flexible Posterior by $z_1 = f_\theta(x_1), z_2 = f_\theta(x_2; z_1), z_3 = f_\theta(x_3; z_1, z_2)$
  - In VAE, Need the generation on $z\sim q(z|x)$ to calculate the expectation in the Monte-carlo manner

KAIST

Diversification of Coupling Flow Function

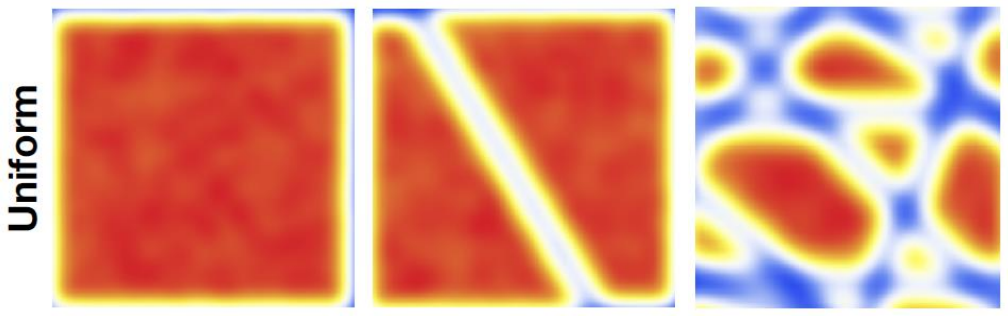
- What could be the structure of the transformation function?
  - By following the Bayesian network interpretation
    - $x_i = f_\theta\big(z_i; \mathrm{parent}(x_i)\big)$
- Desired/Required nature of f_θ
  - Need to be applied to a single dimension by creating influence from the other many
    - Invertible
    - Nonlinear
  - Variations
    - NICE, RealNVP, IAF-VAE
      - Using affine transformation (common)
    - Glow
      - Using invertible 1x1 convolution
    - Flow++
      - Using continuous mixture CDFs
    - Spline Flow
      - Using splines

Diverse Choices of Transformation

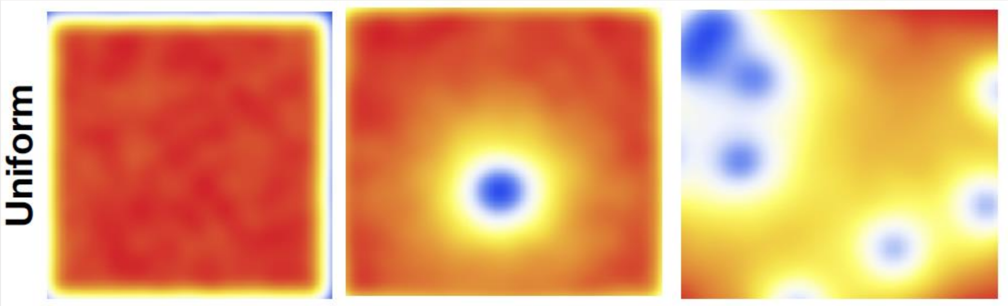
	Name	$f(x)$	$(\boldsymbol{f^{-1}}, \lvert \boldsymbol{Df} \rvert)$ Complexity
Finite Flow	Elementwise Flow	$(f(x_1), \ldots, f(x_D))$	
	Linear Flow	$Ax + b$	$O(D^3), \mathrm{O}(\mathrm{D}^3)$
	Planar Flow	$x + uh(w^T x + b)$	$-, O(D)$
	Radial Flow	$x + \frac{\beta}{\alpha + \lVert x - x_0 \rVert}(x - x_0)$	$-, O(D)$
	Coupling Flow	$(h(x^A; \Theta(x^B)), x^B)$	Invertible iff h is invertible / Depends on h, but efficient
	Autoregressive Flow	$(h(x_t; \Theta_t(x_{1:t-1})))_t$	Inverse cost high / determinant elementwise product
	Spline Flow	Spline	Inverse requires Newton method / determinant closed form
	Residual Flow	$(x^A + F(x^B), x^B + G(f(x)^A))$	Invertible easy / Jacobian difficult
Infinitesimal Flow	Langevin Flow	$\frac{\partial}{\partial t} q_t(\mathrm{x}) = -\sum_i \frac{\partial}{\partial \mathrm{x}_i}[F_i(\mathrm{x}, t) q_t] + \frac{1}{2} \sum \frac{\partial}{\partial \mathrm{x}_i \partial \mathrm{x}_j}[D_{ij}(\mathrm{x}, t) q_t]$	
	Hamiltonian Flow	$\mathcal{H}(\mathrm{x}, \omega) = -\mathcal{L}(\mathrm{x}) - \frac{1}{2} \omega^T M \omega$	

Flow Models with Diverse Transformation as f_θ

- Planar Flow
  - $f_k(z_{k-1}) = z_{k-1} + u_k h_k(w_k^T z_{k-1} + b_k)$
    - z_{k-1} : Linear component
    - $u_k h_k(w_k^T z_{k-1} + b_k)$: Nonlinear component
  - Determinant of Jacobian
    - $\left|\frac{\partial f_k}{\partial z_{k-1}}\right| = \left|I + u_k\left(h_k'\left(w_k^T z_{k-1} + b_k\right)w_k\right)^T\right| = \left|1 + u_k^T h_k'\left(w_k^T z_{k-1} + b_k\right)w_k\right|$
  - Objective function
    - $\log q_K(z_K) = \log q_0(z_0) - \sum_{k=1}^K \log\left|1 + u_k^T \psi_k(z_{k-1})\right|$, where $\psi_k(z_{k-1}) = h_k'\left(w_k^T z_{k-1} + b_k\right)w_k$



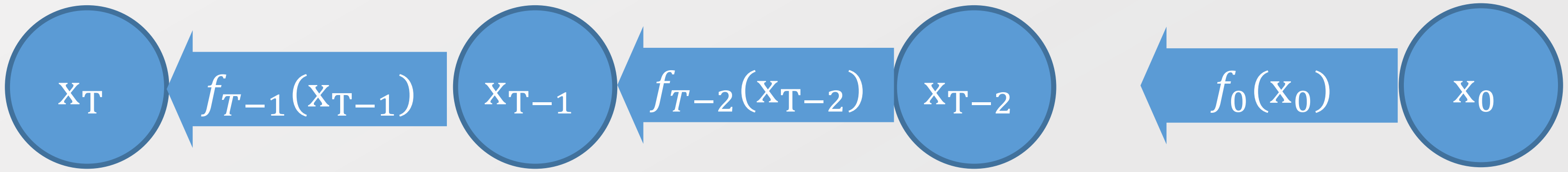
- Radial Flow
  - $f_k(z_{k-1}) = z_{k-1} + \frac{\beta}{\alpha + \|z_{k-1} - z_{k-1}^*\|}(z_{k-1} - z_{k-1}^*)$
  - Determinant of Jacobian
    - $\left|\frac{\partial f_k}{\partial z_{k-1}}\right| = \left[1 + \frac{\beta}{\alpha + \|z_{k-1} - z_{k-1}^*\|}\right]^{D-1}\left[1 + \frac{\beta}{\alpha + \|z_{k-1} - z_{k-1}^*\|} - \frac{\|z_{k-1} - z_{k-1}^*\|}{\left(\alpha + \|z_{k-1} - z_{k-1}^*\|\right)^2}\right]$
  - Objective function
    - $\log q_K(z_K) = \log q_0(z_0) - \sum_{k=1}^K (D-1)\log\left(1 + \frac{\beta}{\alpha + \|z_{k-1} - z_{k-1}^*\|}\right) - \sum_{k=1}^K \log\left(1 + \frac{\beta}{\alpha + \|z_{k-1} - z_{k-1}^*\|} - \frac{\|z_{k-1} - z_{k-1}^*\|}{\left(\alpha + \|z_{k-1} - z_{k-1}^*\|\right)^2}\right)$

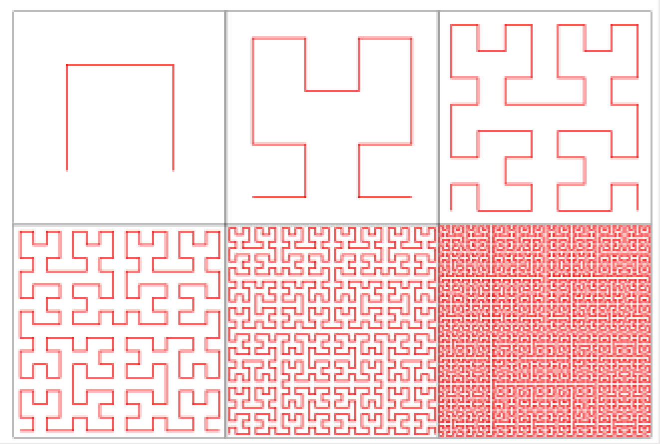


COMBINATION
WITH OTHER GENERATIVE MODELS

Dimension Restriction of Flow

- Rank-Nullity Theorem
  - [Rank-Nullity Theorem] Let $T: V \rightarrow W$ be a linear transformation, where V and W are the finite-dimensional vector spaces. Then, $Rank(T) + Nullity(T) = \dim V$, where $Rank(T) = \dim\big(T(V)\big)$ and $Nullity(T) = \dim\big(Ker(T)\big)$.
- Simple interpretations
  - [Theorem] A function $f: \mathbb{R}^n \rightarrow \mathbb{R}^\mathrm{m}$ with $n > m$ can be bijective.
  - [Theorem] A continuous function $f: \mathbb{R}^n \rightarrow \mathbb{R}^\mathrm{m}$ with $n > m$ cannot be bijective.
- Requirements of the flow transformation
  - Bijective function
  - Differentiable function
  - Hence, the flow transformation needs to be the matching dimension between the data and the latent variables





Hilbert Curve

(https://math.stackexchange.com/questions/557801/why-is-this-not-a-space-filling-curve)

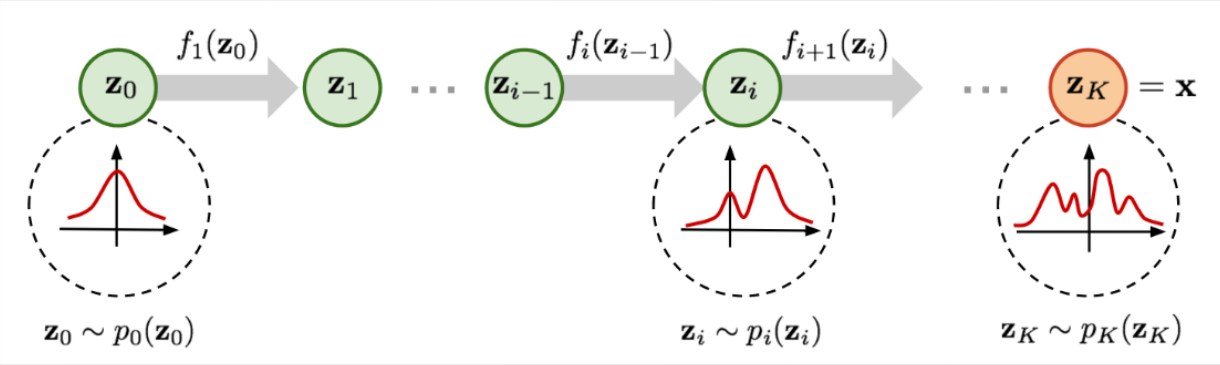
Learning in Implicit Models

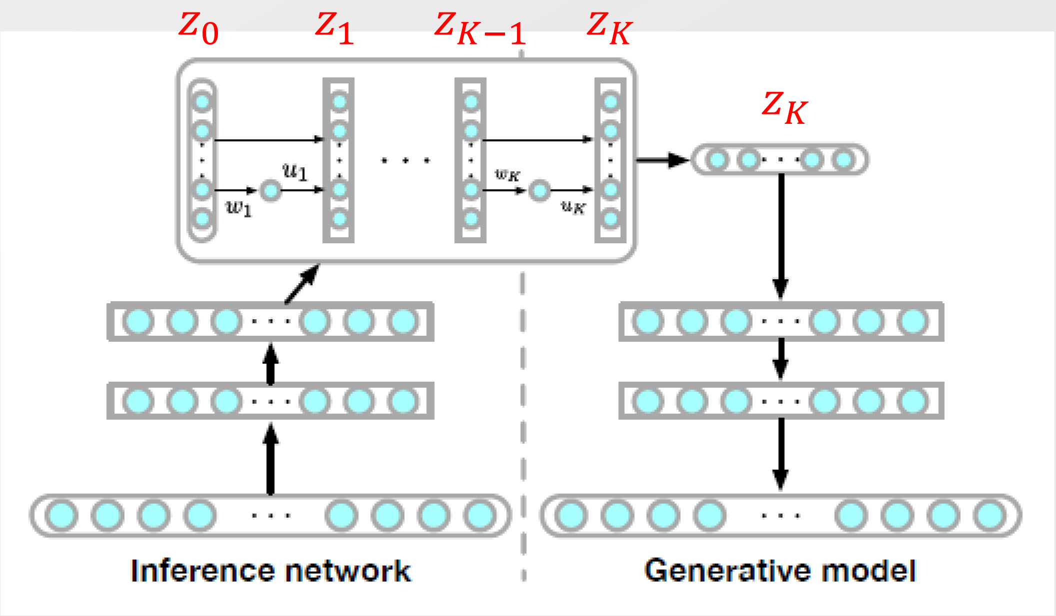
- $\mathbb{E}_{q_\phi(h|e)}\left[\ln \frac{q_\phi(h)}{p(h)}\right]$ is the expected density ratio
  - The density ratio can be computed by building a classifier to distinguish observed data from that generated by the model.
    - $p(h)$ is used for a set of n samples $E_p = \{e_1^{(p)}, \dots, e_n^{(p)}\}$
    - $q_\phi(h)$ is used for a set of n samples $E_q = \{e_1^{(q)}, \dots, e_n^{(q)}\}$
      - Through the ancestral sampling of $q_\phi(h|e)$ by randomly selecting e
  - A random variable y that assigns a label $y = 1$ to all samples in E_p and $y = 0$ to all samples in E_q.
    - $p(h) = p(h|y = 0)$ and $q_\phi(h) = p(h|y = 1)$
    - $p^*(h|y) \equiv \begin{cases} q_\phi(h), y = 1 \\ p(h), y = 0 \end{cases}$
- By applying Bayes' rule, we can compute the ratio $r(\boldsymbol{h}) = \ln \frac{q_\phi(h)}{p(h)}$ as:
  - $\frac{q_\phi(h)}{p(h)} = \frac{p^*(h|y=1)}{p^*(h|y=0)} = \frac{\frac{p^*(h,y=1)}{p^*(y=1)}}{\frac{p^*(h,y=0)}{p^*(y=0)}} = \frac{\frac{p^*(y=1|h)p^*(h)}{p^*(y=1)}}{\frac{p^*(y=0|h)p^*(h)}{p^*(y=0)}} = \frac{p^*(y=1|h)}{p^*(y=0|h)} \cdot \frac{\pi}{1-\pi} = \frac{p^*(y=1|\boldsymbol{h})}{p^*(y=0|\boldsymbol{h})} = \frac{D(h)}{1-D(h)}$
    - $D(h) = p^*(y = 1|h)$
    - Which indicates density ratio estimation = class probability estimation.
  - The problem is reduced to computing the probability $p(y = 1|h)$
    - Discriminative modeling can be applied, as a discriminator

Discriminator in VAE
for
Optimal Prior Learning

Variational Autoencoder + Flow

- Merge of two generative models
  - Variational autoencoder
    - requires a flexible latent space model
    - provides the information extraction through data compresions
  - Flow
    - restricted to the continuous and bijective transformation, so no dimensionality reduction
    - provides the flexible modeling on the probability space
- What-if we model the latent space by flow
  - If $z_0 \sim q_0(z_0)$ and
    - $z_K = f_K \circ \cdots \circ f_2 \circ f_1(z_0)$,





ELBO of VAE + Flow

- Original ELBO in VAE
  - $\mathcal{L} = \mathbb{E}_{q_\phi(H|E)}[\log p_\theta(E|H)] - D_{KL}\left(q_\phi(H|E)||p_\theta(H)\right)$
- $L_{ELBO}(\theta,\phi) = \int q_\phi(z_K|x)\log p_\theta(x|z_K)\,dz_K + \int q_\phi(z_K|x)\log\frac{p(z_K)}{q_\phi(z_K|x)}dz_K$

$$= E_{z_K\sim q_\phi(z_K|x)}[\log p_\theta(x|z_K)] + E_{z_K\sim q_\phi(z_K|x)}\left[\log\frac{p(z_K)}{q_\phi(z_K|x)}\right]$$
$$= E_{z_0\sim q_\phi(z_0|x)}[\log p_\theta(x|f_K\circ\cdots\circ f_1(z_0))] + E_{z_0\sim q_\phi(z_0|x)}\left[\log\frac{p(z_K)}{q_\phi(f_K\circ\cdots\circ f_1(z_0)|x)}\right]$$
$$= E_{z_0\sim q_\phi(z_0|x)}[\log p_\theta(x|f_K\circ\cdots\circ f_1(z_0))] + E_{z_0\sim q_\phi(z_0|x)}[\log p(z_K)] - E_{z_0\sim q_\phi(z_0|x)}[\log q_\phi(f_K\circ\cdots\circ f_1(z_0)|x)]$$

  - Law of the unconscious statistician (LOTUS)
    - $E_{z_K\sim q_K(z_K)}[\log h(z_K)] = E_{z_0\sim q_0(z_0|x)}[\log h(f_K\circ\cdots\circ f_1(z_0))]$
  - Flow
    - $\log q_\phi(z_K) = \log q_\phi(z_0) - \sum_{k=1}^{K}\log\det\left|D_{z_k}f_k\right|$
- Eventually, ELBO becomes
  - $L_{ELBO}(\theta,\phi) = E_{z_0\sim q_\phi(z_0|x)}[\log p_\phi(x|f_K\circ\cdots\circ f_1(z_0))] + E_{z_0\sim q_\phi(z_0|x)}[\log p(z_K)]$

$$-E_{z_0\sim q_\phi(z_0|x)}[\log q_\phi(z_0|x)] + \sum_{k=1}^{K} E_{z_0\sim q_\phi(z_0|x)}[\log\det|D_{z_k}f_k|]$$

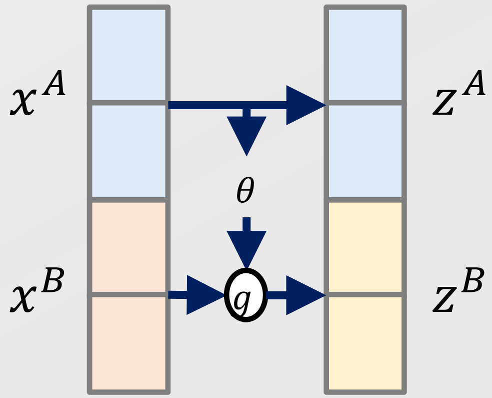
CONTINUOUS TRANSFORMATION ON FLOW

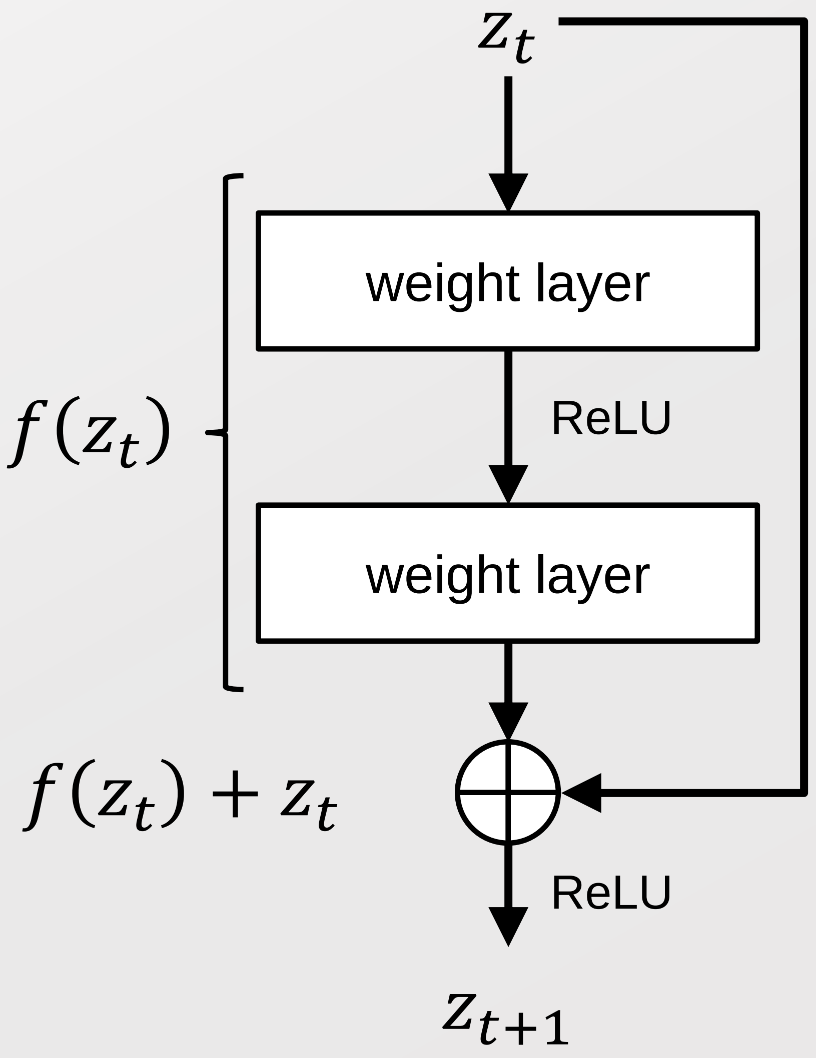
Detour: Residual Network

- Generalized Coupling Flows : general approach to construct non-linear flows

$$f(x) = f([x^A; x^B]) = \left(x^A, g\left(x^B; \theta(x^A)\right)\right) = z$$

  - $f = (f^A, f^B)$
    - where $f^A(x^A) = x^A$ (identity), $f^B(x^B) = g\left(x^B; \theta(x^A)\right)$
- Residual Network Structure
  - $F(z_t) = W_2\sigma(W_1 z_t)$
  - $z_{t+1} = z_t + f(z_t; \{W_i\})$
- Similarity
  - Existence of the identity operation
    - ResNet
      - All dimensions are processed
    - Flow
      - Some dimensions are processed
      - The other dimensions are kept





He et al., *Deep Residual Learning for Image Recognition*, CVPR 2016

APPLIED ARTIFICIAL INTELLIGENCE LAB

- Invertible Residual Networks
  - Generative Modeling with i-ResNets
    - Let $F: \mathcal{X} \to \mathcal{Z}$ be a transformation function. Then, equation of flow models:

$$\log p_x(x) = \log p_z(z) + \log \left| \det \frac{\partial F}{\partial x} \right|$$

    - From the residual connection ($F = I + f$),

$$\log \left| \det \frac{\partial F}{\partial x} \right| = tr\left(\log\left(I + \frac{\partial f}{\partial x} \right) \right) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{tr\left(\left(\frac{\partial f}{\partial x} \right)^k \right)}{k}$$

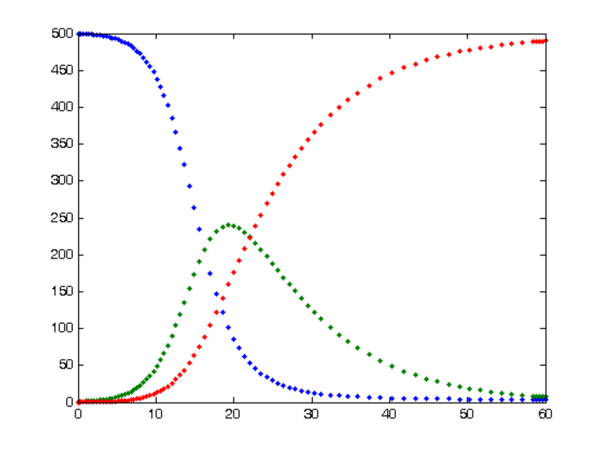
    - Stochastic Approximation of log-determinant
      - 1) Hutchinson's trace estimator ➔ $tr(J_f) = \mathbb{E}_{p(v)}[v^T J_f v]$ with $\mathbb{E}[v] = 0, Cov(v) = I$
      - 2) Infinite sum ➔ truncated at index n

$$\log p_x(x) = \log p_z(z) + \mathbb{E}_v \left[\sum_{k=1}^{n} \frac{(-1)^{k+1}}{k} v^T J_f(x)^k v \right]$$

      - By 2), the estimator is biased. (but the paper said 5-10 iterations are enough)

Detour: Ordinary Differential Equation

- Ordinary Differential Equation
  - Differential equation with a set of functions on an independent variable and the derivatives of the functions
  - $F\left(x, y, y', \dots, y^{(n)}\right) = 0$
    - y: depenent variable
    - x: independent variable
    - $y = f(x)$: unknown function of x
- System of Differential Equations
  - Coupled differential equations
    - $\boldsymbol{y} : \boldsymbol{y}(\boldsymbol{x}) = [y_1(x), y_2(x), \dots, y_m(x)]$
    - $\begin{pmatrix} f_1\left(x, y, y', \dots, y^{(n)}\right) \\ f_2\left(x, y, y', \dots, y^{(n)}\right) \\ \dots \\ f_m\left(x, y, y', \dots, y^{(n)}\right) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 0 \end{pmatrix}$
  - Example : SIR Model
    - $\frac{dS}{dt} = -\beta IS, \frac{dI}{dt} = \beta IS - \nu \mathrm{I}, \frac{dR}{dt} = \nu \mathrm{I}$
      - $\boldsymbol{y} = [S, I, R], x = t$
  - How to calculate $\boldsymbol{y}$ for an arbitrary x

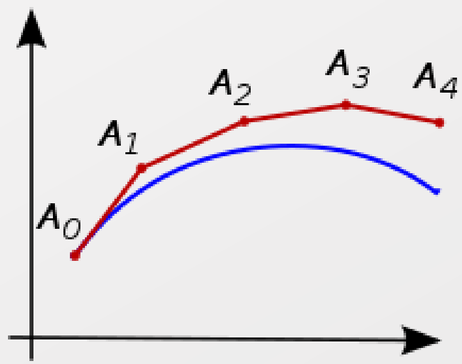


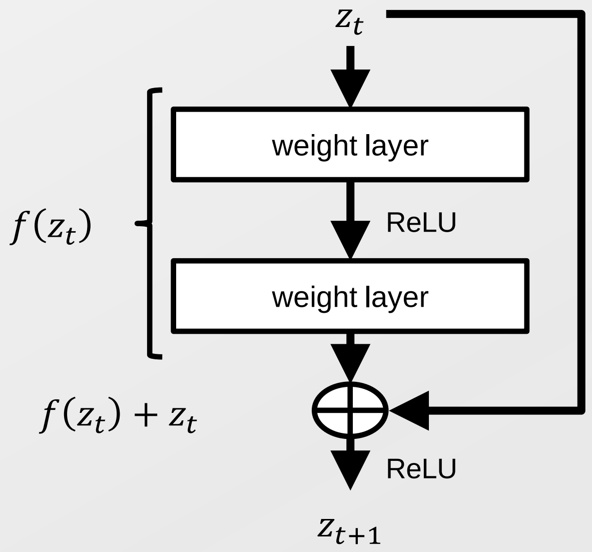
https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology

Detour: Euler Method

- Ordinary Differential Equation
  - $$\begin{pmatrix} f_1\left(t, y, y', \dots, y^{(n)}\right) \\ f_2\left(t, y, y', \dots, y^{(n)}\right) \\ \dots \\ f_m\left(t, y, y', \dots, y^{(n)}\right) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 0 \end{pmatrix}$$
  - How to calculate $\boldsymbol{y}$ for an arbitrary t
- Given an initial point and an ODE
  - $y(t_0) = y_0$
  - $y'(t) = f\big(t, y(t)\big)$
  - Let's assume a Talyor expansion on y at t_0, and let's assume a very small step of h
    - $y(t_0 + h) = y(t_0) + hy'(t_0) + \frac{1}{2}h^2 y''(t_0) + O(h^3)$
    - $y(t_0 + h) \approx y(t_0) + hf\big(t, y(t)\big)$
- Euler method
  - $y_{n+1} = y_n + hf(t_n, y_n)$
    - Assuming given $f(t_0, y_0)$
  - can be expanded to Runge-Kutta method
    - Dynamic setting of h is feasible

https://en.wikipedia.org/wiki/Euler_method



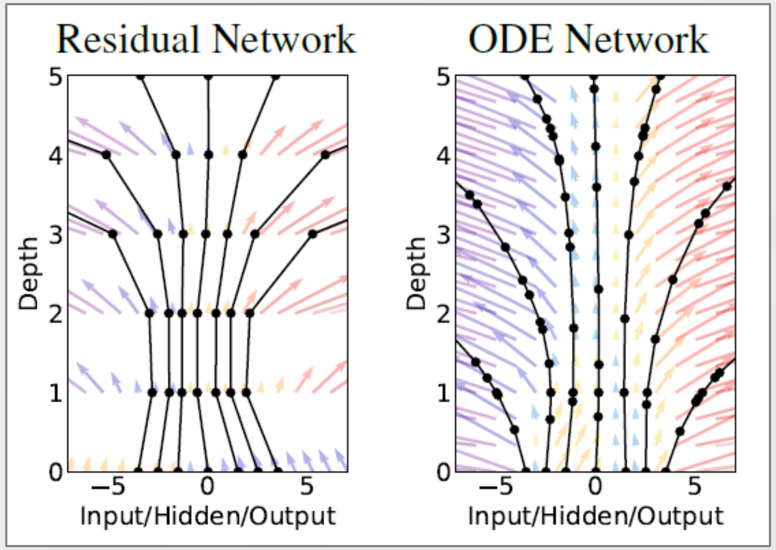


$$F(z_t) = W_2\sigma(W_1 z_t)$$
$$z_{t+1} = z_t + f(z_t; \{W_i\})$$
$$\rightarrow$$
$$z_{t+1} = z_t + \Delta t \frac{f(z_t; \theta)}{\Delta t} = z_t + \Delta t g(z_t, t; \theta)$$

Exact Value Output vs. Diffential Value Output from NN

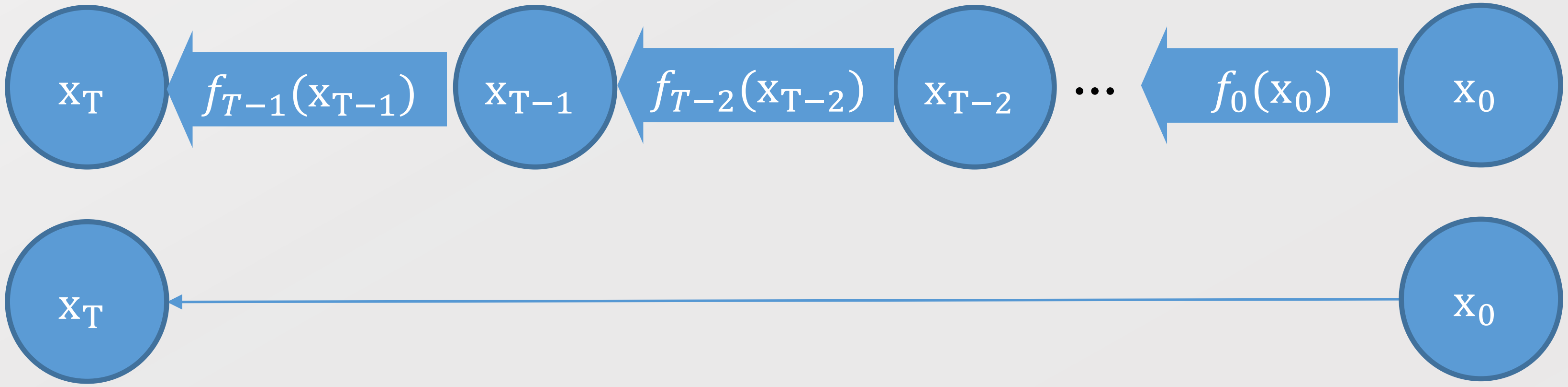
- When we consider t as the layer number
  - Initially, t becomes the integer index of layer
  - However, is it feasible to see t as a continuous value?
- Usual neural network
  - $z_{t+1} = f_{NN_t}(z_t; \theta)$
    - z_{t+1} is the actual output given z_t
- Differential value integration with ODE solver
  - $\frac{dz_t}{dt} = g_{NN}(z(t), t; \theta)$
  - $z(t_1) = ODESolver(z(t_0), g_{NN}, t_0, t_1, \theta) = z(t_0) + (t_1 - t_0) g_{NN}(z(t_0), t_0; \theta)$
- Now, we can calculate z_t at t with precision values
- Then, how to view the flow model with the discrete transformation?

R. Chen et al., *Neural Ordinary Differential Equations*, NeurIPS 2018



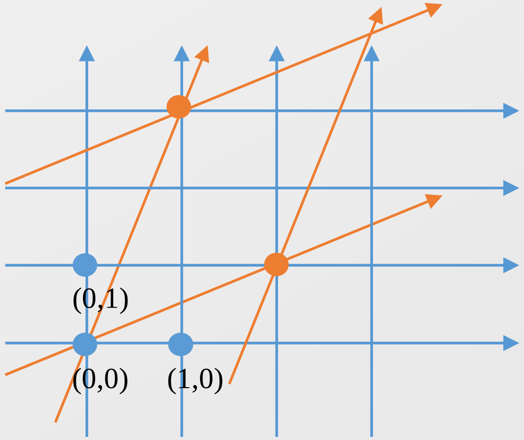
Discrete Transformation:

Continuous Transformation:



Background of Continuous Normalizing Flow

- Compose the multiple transformation as a chain
  - $f = f_K \circ f_{K-1} \circ \cdots \circ f_1$
  - Training with log-likelihood
    - $\max_\theta \left\{ \log p_Z\left(f_\theta(x)\right) + \sum_{i=1}^{K} \log \left| \det \frac{\partial f_i}{\partial f_{i-1}} \right| \right\}$
  - Role of $\det \frac{\partial f_i}{\partial f_{i-1}}$: Estimating the change of probability after the transformation
- Instantaneous Change of Variables
  - Assumption
    - Let $z(t)$ be a finite continuous random variable with $p(z(t))$ dependent on time
    - Let $\frac{\boldsymbol{dz}}{\boldsymbol{dt}} = \boldsymbol{g}(\boldsymbol{z}(\boldsymbol{t}), \boldsymbol{t})$ be a differential equation describing a continuous-in-time transformation of $z(t)$
    - Assuming that f is uniformly Lipschitz continuous in z and continuous in t
  - Then,
    - The change in log probability also follows a differential equation, $\frac{\boldsymbol{\partial \log p(z(t))}}{\boldsymbol{\partial t}} = -\boldsymbol{tr}\left(\frac{\boldsymbol{dg}}{\boldsymbol{dz(t)}}\right)$
      - $\boldsymbol{g}$ does not have to be bijective, but $\boldsymbol{g}$ should be Lipschitz continuous
- Continuous transformation in probability
  - $z(t_1) - z(t_0) = \int_{t_0}^{t_1} g(z(t), t)dt$, where $z(t_1) = x$
  - $\log p\left(z(t_1)\right) - \log p\left(z(t_0)\right) = \int_{t_0}^{t_1} -\boldsymbol{tr}\left(\frac{\boldsymbol{dg}}{\boldsymbol{dz(t)}}\right) dt$, where $\log p(z(t_1)) = \log p(x)$
    - Log Determininant of Jacobian matrix → Trace of Jacobian matrix
    - Bijective constraint → Lipschitz constraint



FFJORD

- (Discrete) Normalizing Flow
  - $\log p_\theta(x) = \log p(z_0) - \sum_{k=1}^{K} \log \det \left|\frac{\partial f_k}{\partial z_k}\right|$
  - Complexity
    - $O(D^3)$ to calculate the log determinant

$$\begin{pmatrix} \frac{\partial f_\theta^1}{\partial x_1} & \cdots & \frac{\partial f_\theta^1}{\partial x_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_\theta^D}{\partial x_1} & \cdots & \frac{\partial f_\theta^D}{\partial x_D} \end{pmatrix}$$

- Continuous Normalizing Flow
  - $\log p_\theta(x) = \log p(z_0) - \int_{t_0}^{t_1} \boldsymbol{tr}\left(\frac{\boldsymbol{dg}}{\boldsymbol{dz(t)}}\right) \boldsymbol{dt}$
  - Complexity
    - $O(D^2)$ to calculate the trace
- FFJORD: Continuous-time Flow with Hutchinson's trace estimator
  - $\log p_\theta(x) = \log p(z_0) - \mathbb{E}_{\boldsymbol{p(\epsilon)}}\left[\int_{t_0}^{t_1} \boldsymbol{v^T}\left(\frac{\boldsymbol{dg}}{\boldsymbol{dz(t)}}\right)\boldsymbol{vdt}\right]$
  - Complexity
    - $O(D)$ to calculate the trace estimator

$$tr(A) = \mathbb{E}_{p(v)}[v^T A v]$$
Where $\mathbb{E}[v] = 0, Cov(v) = I$