Thursday, April 15, 2021 10:17 AM

Lecture 23 Density Estimation & Normalizing Flows

last times: GAN & VAE learn density implicitly
(only access to samples, not $p(x)$)

Some tasks require $p(x)$

- likelihood of an observation (anomaly detection)
- computing expectation values, means, variances

Density Estimation

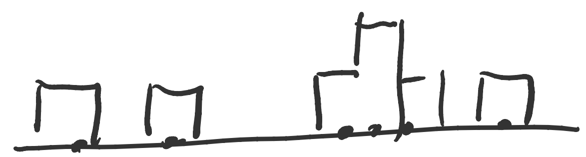
objective: learn the underlying pdf from which a set of iid samples was drawn

Friday, April 16, 2021 3:28 PM

pdf : probability density function $p(x) \geq 0$ $\int p(x)\,dx = 1$

iid independent, identically distributed

1st try: histogram

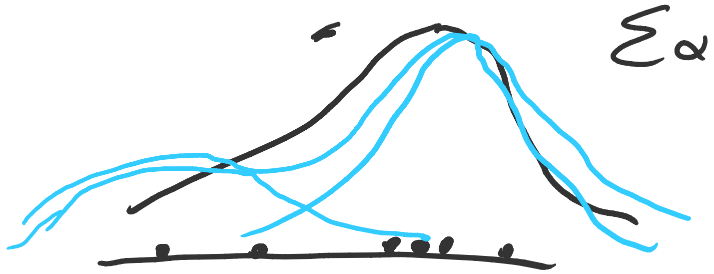


parametric & non-parametric DE

- fit parameters of known dist.
- GMM - Gaussian Mixture Models

$p(x) = \sum_i \alpha_i N(\mu_i, \sigma_i)$

$\sum \alpha_i = 1$

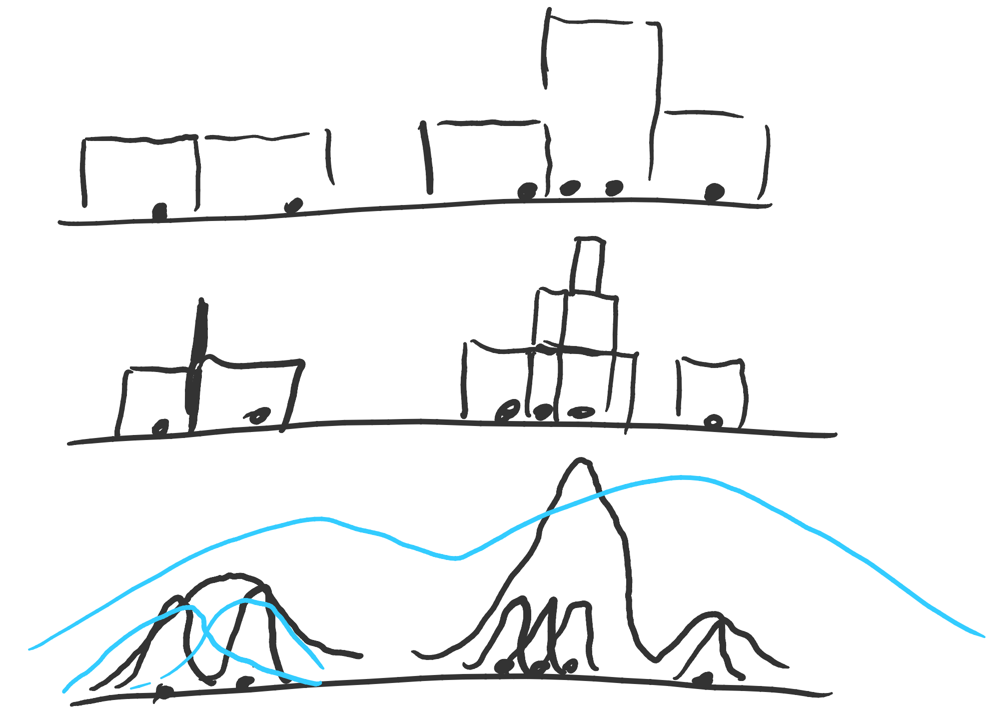


+ fast
- assumptions on data
- inflexible

Kernel Density Estimation - KDE

$p(x) = \frac{1}{nh} \sum_i^n K\left(\frac{x - x_i}{h}\right)$

sum all data points



+ no parameters (easy "learning") h... bandwidth
+ flexible
- numerically expensive (store full dataset)

Friday, April 16, 2021 3:42 PM

NN as universal functions offer new possibilities → use as coordinate transformations

base distribution
- Gaussian
- uniform

⇒ easy z

$\xrightarrow{\text{bijector } f}$ / $\leftarrow$

"target" dist.
$p(x)$
data
x

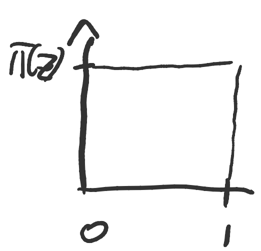
coordinate transform $\vec{x} = f(\vec{z})$

$$\boxed{p(x) = \pi(z) \left| \det \frac{df}{dz} \right|^{-1}}$$

→ need Jacobian determinant

→ need inverse $\vec{z} = f^{-1}(\vec{x})$

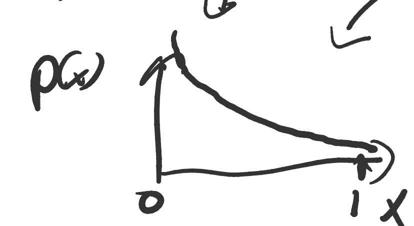
ex:

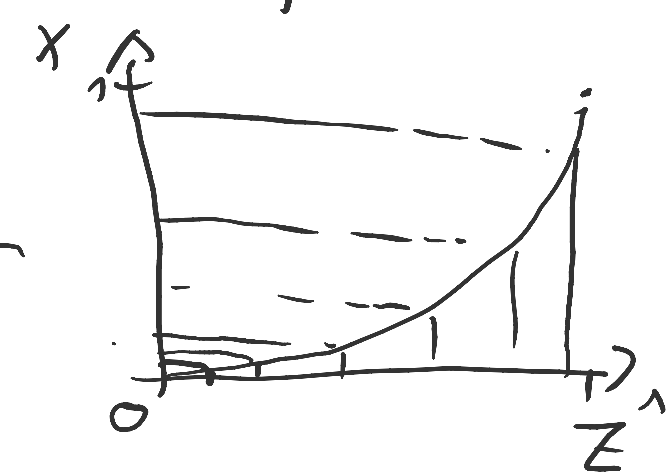


uniform

$x = f(z) = z^2$

$p(x) = 1 \cdot |2z|^{-1} = \frac{1}{2\sqrt{x}}$





bijector: work both ways

$$\pi(z) \underset{\text{inference}}{\overset{\text{sampling}}{\rightleftarrows}} p(x)$$

Friday, April 16, 2021 3:51 PM

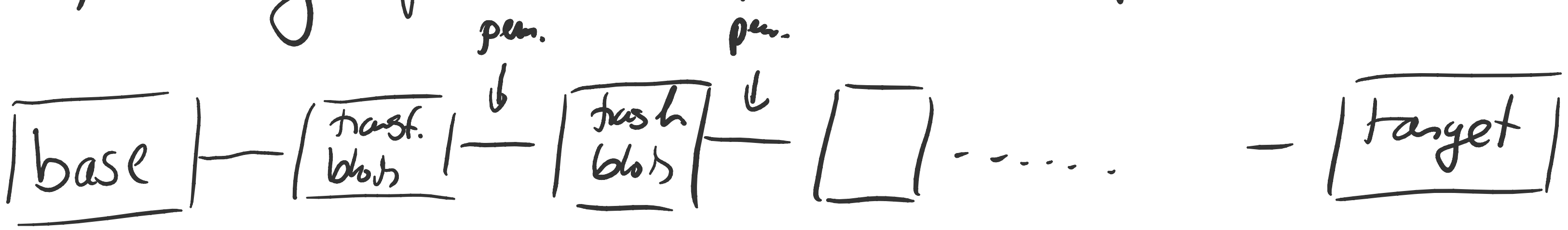
what about using $f(z) = NN$?

↳ does not work :

- inverse ?
- Jacobian $\mathcal{O}(n^3)$

Normalizing Flows avoid these problems

by learning parameters of a series of invertible transformations w/ tractable Jacobian



(chain of bijectors is bijector)

$$X = C_K \circ C_{K-1} \circ \dots \circ C_1(\vec{z}_0) \qquad \vec{z}_0 \sim \pi_0(\vec{z}_0)$$

$$p(x) = \pi_0(\vec{z}_0) \cdot \prod_{k=1}^{K} \left| \det \frac{\partial C_k}{\partial z_{k-1}} \right|^{-1}$$

1505.05770

since we have $p(x)$, we can just train by $\text{Loss} = -\log p(x)$

Friday, April 16, 2021 4:00 PM

inside each block: assume the transformation factorizes

$$\vec{C}(\vec{z}, \vec{\mu}) = \left(C_1(z_1, \vec{\mu}_1), \; C_2(z_2, \vec{\mu}_2), \; \ldots, \; C_n(z_n, \vec{\mu}_n) \right)^T$$

$\vec{\mu}_i$ still depend on z_j $(i \neq j)$ to capture all correlations

(+ external conditions)

there are 2 main architectures to tame Jacobians

① autoregressive models

assume $\vec{\mu}_1 = \text{const}$; $\vec{\mu}_2(z_1), \ldots, \vec{\mu}_i(z_1, z_2, z_3, \ldots, z_{i-1}), \ldots$

$x_i = C_i(z_{1:i})$ $\quad J = (\ \ )$

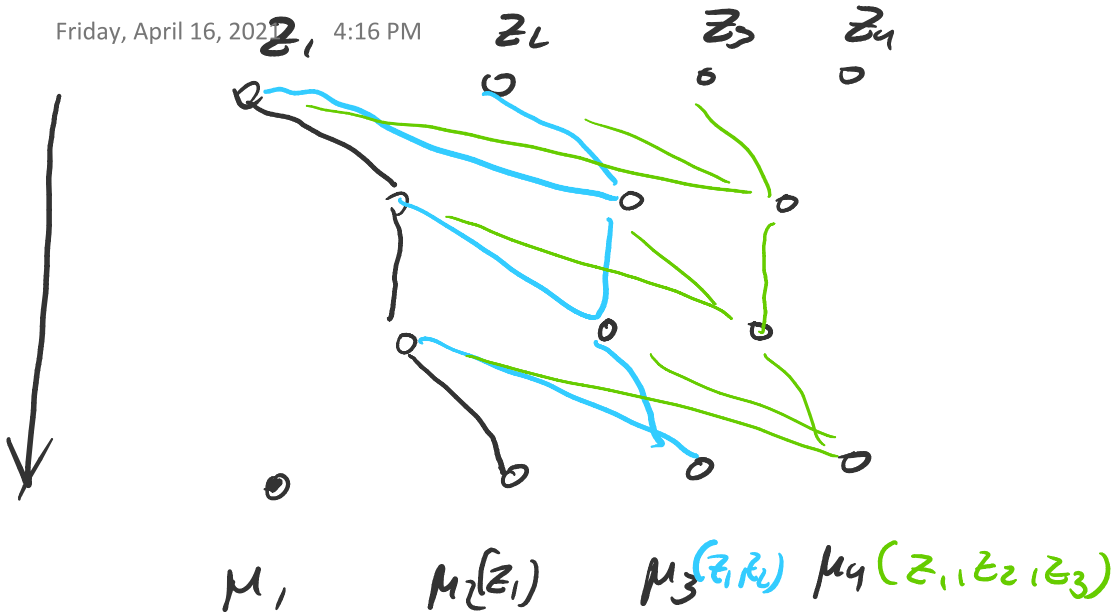
$p(x_1), \; p(x_2 | x_1), \; p(x_i | x_1, x_2, \ldots x_{i-1})$

$\det J = \prod_{i=1}^{n} J_{ii}$

$$p(x) \approx \prod_{i=1}^{n} p(x_i | x_{1:i-1})$$

NN realization: MADE Masked Autoencoder for Density Estimation 1502.03509

Friday, April 16, 2021 4:16 PM



forward pass : single pass: all μs

inverse pass : loop n-times to get all z's

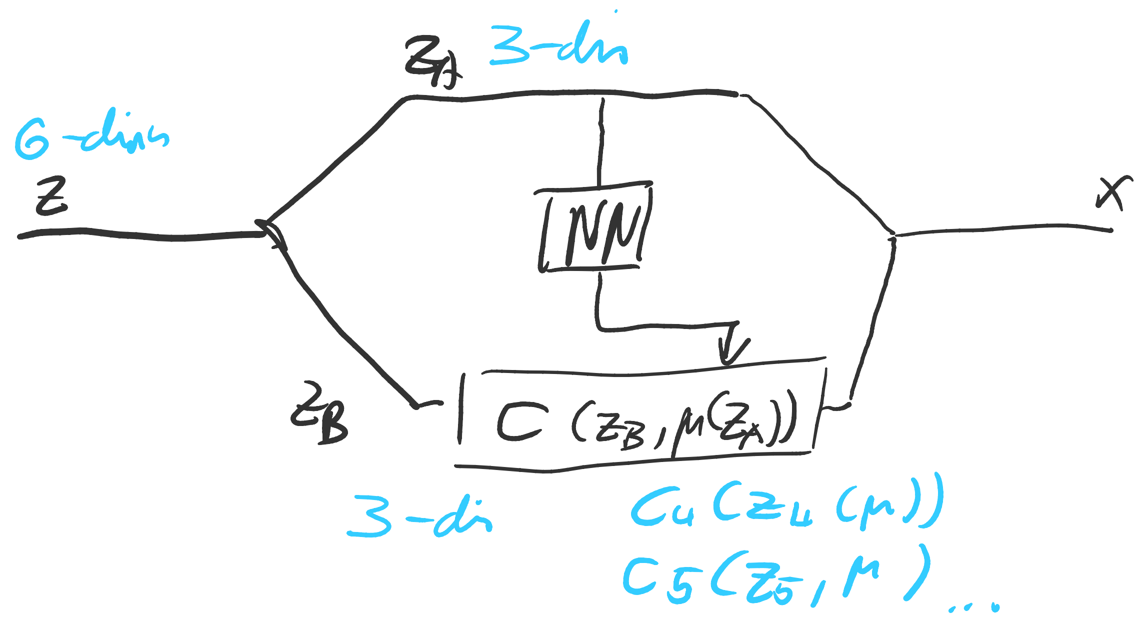
	MAF	IAF
fast	inference	sampling
slow	sampling	inference

MAF : masked autoregressive flow 1705.07057

IAF : inverse autoregressive flow 1606.04934

② Coupling Layer based flows (Real NVP 1605.08803)

split z in 2 sets z_A, z_B



forward

$x_A = z_A$

$x_B = C_B(z_B, \mu(z_A))$

inverse

$z_A = x_A$

$z_B = C^{-1}(x_B; \overset{\mu(x_A)}{\mu(z_A)})$

$$|\det J| = \begin{vmatrix} 1 & 0 \\ \frac{\partial C_B}{\partial z_A} & \frac{\partial C_B}{\partial z_B} \end{vmatrix}$$ ← diagonal

equal speed both directions

Friday, April 16, 2021 4:32 PM

now about $C(z, \vec{\mu})$

↳ detailed choice depends on domain

base gaussian : $-\infty, +\infty$

uniform : $0, 1$

logit

sigmoid

- affine coupling layer $C(z, \mu) = z \odot \exp(s) + t = x$

$\mu = s, t$

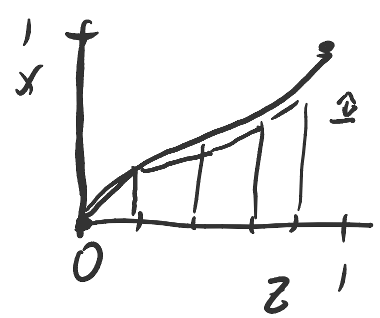
$z' \to NN \to s, t$

↳ inverse : $(x - t) \odot \exp(-s) = z$

(not as expressive)

$|\det J| = \exp(\Sigma s)$

- piecewise splines (for finite domain) 1808.03856



- pw - linear
  quadratic, cubic

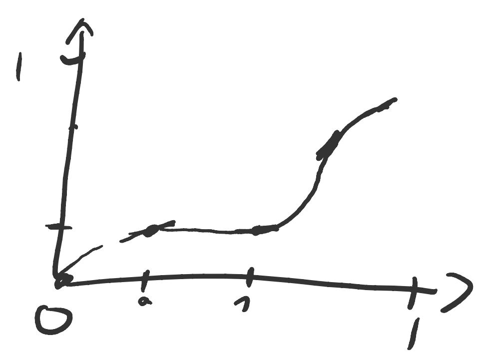
1906.04032

- Rational Quadratic spline

$C = \dfrac{a_2 \alpha^2 + a_1 \alpha + a_0}{b_2 \alpha^2 + b_1 \alpha + b_0}$

α ... where you are in the bin

a_i, b_i given by NN



Friday, April 16, 2021 4:49 PM

$$x_1 = C_1(z_1, \mu_1^{=\text{const}}) \longrightarrow p(x_1)$$

$$x_2 = C_2(z_2, \mu_2(z_1)) \qquad p(x_2 \mid x_1)$$

$$x_3 = C_3(z_3, \mu_3(z_1, z_2)) \qquad p(x_3 \mid x_1, x_2)$$

$$\ldots \qquad \ldots$$