· Nonlinear Independent Components Estimation (NICE)

additive layer:

partition the variables z into two disjoint subsets $z[1:d]$ and $z[d:n]$

forward mapping $z \to x$ $\begin{cases} x[1:d] = z[1:d] \\ x[d:n] = z[d:n] + m_\theta(z[1:d]) \end{cases}$ identity transformation

m_θ is a neural network d inputs, $n-d$ outputs

inverse mapping $x \to z$ $\begin{cases} z[1:d] = x[1:d] \\ z[d:n] = x[d:n] + m_\theta(x[1:d]) \end{cases}$

Jacobian of forward mapping

$$J = \begin{array}{c c} & \begin{array}{cc} z[1:d] & z[d:n] \end{array} \\ \begin{array}{c} x[1:d] \\ x[d:n] \end{array} & \begin{bmatrix} I & 0 \\ \nabla_x m_\theta(x[1:d]) & I \end{bmatrix} \end{array}$$

$|J| = 1$ Volume preserving transformation

rescaling layer:

forward mapping $z \to x$ $\{ \ x_i = s_i z_i \qquad (s_i > 0)$

inverse mapping $x \to z$ $\{ \ z_i = x_i / s_i$

Jacobian $J = \mathrm{diag}(s)$ $\det(J) = \prod_i s_i$

Additive coupling layers are composed together (with arbitrary partitions of variable in each layer)

The final layer is a rescaling layer

· **Real-NVP**: Non-volume preserving extension of NICE

Forward mapping $z \to x$:

$$x[1:d] = z[1:d] \qquad \text{identity transformation}$$

$$x[d:n] = z[d:n] \odot \exp(\alpha_\theta(z[1:d])) + u_\theta(z[1:d]) \qquad \alpha_\theta(\cdot) \text{ and } u_\theta(\cdot) \text{ are both neural nets}$$

Inverse mapping $x \to z$:

$$z[1:d] = x[1:d]$$

$$z[d:n] = [x[d:n] - u_\theta(x[1:d])] \,/\, \exp(\alpha_\theta(x[1:d]))$$

Jacobian:

$$\begin{array}{c c} & \begin{array}{cc} z[1:d] & \quad z[d:n] \end{array} \\ \begin{array}{c} x[1:d] \\ \\ x[d:n] \end{array} & \begin{bmatrix} I & 0 \\ \\ \cdots & \text{diag}(\exp \alpha_\theta(z[1:d])) \end{bmatrix} \end{array} \qquad |J| = \exp\left(\sum_i \alpha_\theta(z[1:d])_i\right)$$

· **Autoregressive Model as Flow Models**

Consider a Gaussian autoregressive model

$$p(x) = \prod_{i=1}^{n} p(x_i \mid x_{<i})$$

$$p(x_i \mid x_{<i}) = \mathcal{N}\left(u_i(x_1 \cdots x_{i-1}),\ \exp(\alpha_i(x_1 \cdots x_{i-1}))^2\right)$$ where $\alpha_i(\cdot)$ and $u_i(\cdot)$ are neural nets for $i > 1$ and costants for $i = 1$

Sample from this model:

sample $z_i \sim \mathcal{N}(0,1)$ for $i = 1..n$ } latent space $\mathcal{N}(0, I)$

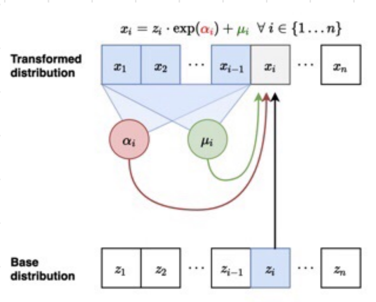
$$x_1 = \exp(\alpha_1) z_1 + u_1 \qquad \text{compute } u_2(x_1), \alpha_2(x_1)$$

$$x_2 = \exp(\alpha_2(x_1)) z_2 + u_2(x_1) \qquad \text{compute } u_3(x_1, x_2), \alpha_3(x_1, x_2)$$

$$x_3 = \exp(\alpha_3(x_1)) z_3 + u_3(x_1, x_2) \qquad \cdots$$

} invertible transformations

· Masked Autoregressive Flow



(α_1 and u_1 doesn't depend on anything)

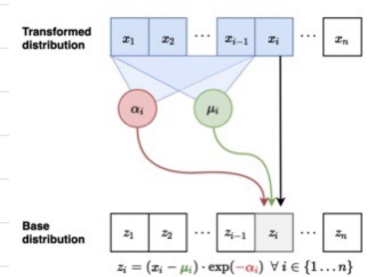
$x_1 = \exp(\alpha_1) z_1 + u_1$

$x_2 = \exp(\alpha_2(x_1)) z_2 + u_2(x_1)$

⋮

$x_i = \exp(\alpha_i(x_{<i})) z_i + u_i(x_{<i})$

Sequential sampling



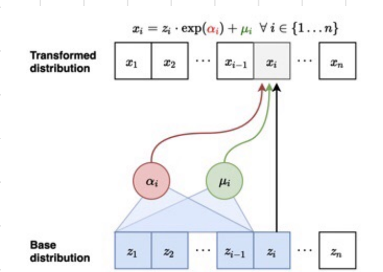
$$z_i = \frac{x_i - u_i(x_{<i})}{\exp(\alpha_i(x_{<i}))}$$

parallel $x \to z$

quick evaluation likelihood

Jacobian is lower-triangular, can compute det|J| efficiently

· Inverse Autoregressive Flow



α_1 and u_1 does not depend on anything

$x_1 = \exp(\alpha_1) z_1 + u_1$

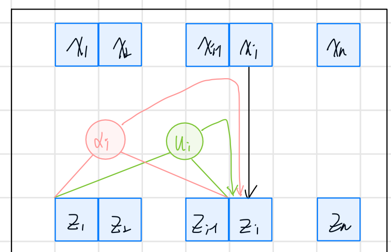
$x_2 = \exp(\alpha_2(z_1)) z_2 + u_2(z_1)$

⋮

$x_i = \exp(\alpha_i(z_{<i})) z_i + u_i(z_{<i})$

parallel $z \to x$

quick sampling



$z_1 = (x_1 - u_1)/\exp(\alpha_1)$

$z_2 = (x_2 - u_2(z_1))/\exp(\alpha_2(z_1))$

⋮

$z_i = (x_i - u_i(z_{<i}))/\exp(\alpha_i(z_{<i}))$

Sequential inverse mapping $x \to z$

Slow evaluation

(but quick evaluation for generated samples)

· Parallel Wavenet

two part training with a teacher and student model

teacher is parameterized by MAF (fast evaluation, slow sampling)

Student is parameterized by IAF (Slow evaluation, fast sampling, fast evaluation for generated samples)

probability density distillation :

first train a teacher model via MLE

then train a student model to minimize the KL divergence student(s) and teacher(t)

$$KL(S,t) = \underbrace{E_{x\sim s}}_{\text{IAF, quick sampling}} \Big[\log \overbrace{S(x)}^{\text{IAF, quick evaluation for generated samples}} - \underbrace{\log t(x)}_{\text{MAF, quick evaluation}} \Big]$$