Parametric density model with invertible transformations

Xun Zheng
xzheng1@andrew.cmu.edu

1 Parametric density model with invertible transformations

1.1 Preliminaries

- Invertible transformation of random variable: $\mathbf{u} \underset{f^{-1}}{\overset{f}{\rightleftarrows}} \mathbf{x}$

$$p(\mathbf{x}) = p_{\text{base}}(\mathbf{u}) \left| \det \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}\mathbf{x}} \right| \tag{1}$$

$$= p_{\text{base}}(f^{-1}(\mathbf{x})) \left| \det(\mathsf{D}f^{-1})(\mathbf{x}) \right| \tag{2}$$

$$= p_{\text{base}}(\mathbf{u}) \left| \det(\mathsf{D}f)(\mathbf{u}) \right|^{-1} \tag{3}$$

  Last line is by inverse function theorem.

- Sampling $\mathbf{x} \sim p(\mathbf{x})$:
  1. Draw noise: $\mathbf{u} \sim p_{\text{base}}(\mathbf{u})$
  2. Forward transform: $\mathbf{x} = f(\mathbf{u})$
- Evaluating $p(\mathbf{x})$:
  1. Backward transform: $\mathbf{u} = f^{-1}(\mathbf{x})$
     (Clearly, can be skipped if $\mathbf{x}$ was simulated from $\mathbf{u}$.)
  2. Evaluate base: $p_{\text{base}}(\mathbf{u})$
  3. Evaluate Jacobian determinant: $\det(\mathsf{D}f^{-1})(\mathbf{x})$ or $\det(\mathsf{D}f)(\mathbf{u})$
- Key problem: design flexible f such that $f(\mathbf{u})$, $f^{-1}(\mathbf{x})$, $\det(\mathsf{D}f)(\mathbf{u})$ is easy to compute. Different trade-off depending on application.
- Application to density estimation: model $p(\mathbf{x})$ using $\mathbf{u} \underset{f^{-1}}{\overset{f}{\rightleftarrows}} \mathbf{x}$

  Need fast evaluation of $p(\mathbf{x})$ on arbitrary $\mathbf{x}$: *i.e.* fast f^{-1} and $\det(\mathsf{D}f)$.
  Fast sampling (*i.e.* fast f) is a plus but not a must.
- Application to variational inference: model $q(\mathbf{z})$ using $\mathbf{u} \underset{f^{-1}}{\overset{f}{\rightleftarrows}} \mathbf{z}$

  Make base distribution data dependent to get an inference network, *e.g.* $q_{\text{base}}(\mathbf{u}) = \mathsf{N}(\mathbf{u}; \mathbf{x}, \sigma^2 I)$.
  Need fast sampling for encoding data, *i.e.* fast f.
  Also need fast evaluation, but only on samples generated by itself. Hence only fast $\det(\mathsf{D}f)$ is needed.

1.2 Low rank Jacobian

- Design f such that $(\mathsf{D}f)$ is low rank, in which case fast $\det(\mathsf{D}f)$ may be available.
- Planar flows (Rezende and Mohamed, 2015):

$$\mathbf{x} = f(\mathbf{u}) = \mathbf{u} + \boldsymbol{\theta}_1 h(\boldsymbol{\theta}_2^T \mathbf{u} + \theta_3) \tag{4}$$
$$\mathbf{u} = f^{-1}(\mathbf{x}) = ??? \tag{5}$$
$$(\mathsf{D}f)(\mathbf{u}) = I + h'(\boldsymbol{\theta}_2^T \mathbf{u} + \theta_3) \cdot \boldsymbol{\theta}_1 \boldsymbol{\theta}_2^T \tag{6}$$
$$\det(\mathsf{D}f)(\mathbf{u}) = 1 + h'(\boldsymbol{\theta}_2^T \mathbf{u} + \theta_3) \cdot \boldsymbol{\theta}_1^T \boldsymbol{\theta}_2 \tag{7}$$

  Last line is by Sylvester's determinant theorem. See Appendix A of (Rezende and Mohamed, 2015) for specific choice of h and parameters that make f invertible.

  Suitable for variational inference: f and $\det(\mathsf{D}f)$ are efficient but f^{-1} is not available.

1.3 Triangular Jacobian via Bayesian network

- Impose a factorization of $p(\mathbf{x})$, *i.e.* a Bayesian network structure:

$$p(\mathbf{x}) = \prod_{j=1}^{d} p_{\text{cond}}(x_j | \mathbf{x}_{\text{pa}(j)}) \tag{8}$$

- Let conditional *reparameterized* as invertible transformation of base distribution:

$$x_j \sim p_{\text{cond}}(x_j | \mathbf{x}_{\text{pa}(j)}) \quad \Longleftrightarrow \quad \begin{aligned} u_j &\sim p_{\text{base}}(u) \\ \alpha_j &= f_{\alpha_j}(?) \\ x_j &= f_{\text{cond}}(u_j; \alpha_j), \quad f_{\text{cond}} \text{ invertible} \end{aligned} \tag{9}$$

  The input to f_{α_j} is left unspecified but we know it must contain information about $\mathbf{x}_{\text{pa}(j)}$.

- f_{α_j} is usually chosen to be a neural network, as it need not be invertible.

  For instance, in the straightforward recurrent autoregressive model, f_{α_j} is the state transition of RNN. However, computation of f^{-1} involves a sequential pass of RNN over d time steps, which is not parallelizable, hence it is considered inefficient.

  Another choice is the masked autoencoder for density estimation (MADE) (Germain et al., 2015), which computes all α_j's in a single pass in parallel using appropriate weight masking.

- f_{cond} is usually chosen to be a simple invertible function, *e.g.* affine coupling

$$f_{\text{cond}}(u_j; \alpha_j) = \alpha_{j,1} u_j + \alpha_{j,2} \tag{10}$$
$$f'_{\text{cond}}(u_j; \alpha_j) = \alpha_{j,1} \tag{11}$$
$$f^{-1}_{\text{cond}}(x_j; \alpha_j) = \frac{x_j - \alpha_{j,2}}{\alpha_{j,1}} \tag{12}$$

- The BN structure and the invertible transformation p_{cond} together define a flow $\mathbf{u} \xrightarrow{f} \mathbf{x}$. Thanks to the DAG structure, Jacobian of f is permutation of lower triangular matrix:

$$(\mathsf{D}f)(\mathbf{u}) = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{u}} = P^T \begin{bmatrix} \frac{\mathrm{d}x_1}{\mathrm{d}u_1} & 0 & 0 \\ \frac{\mathrm{d}x_2}{\mathrm{d}u_1} & \frac{\mathrm{d}x_2}{\mathrm{d}u_2} & 0 \\ \frac{\mathrm{d}x_3}{\mathrm{d}u_1} & \frac{\mathrm{d}x_3}{\mathrm{d}u_2} & \frac{\mathrm{d}x_3}{\mathrm{d}u_3} \end{bmatrix} P \tag{13}$$

where $x_1 \succ x_2 \succ x_3$ is in topological order. Hence Jacobian determinant is product of diagonals:

$$\det(\mathsf{D}f)(\mathbf{u}) = \prod_{j=1}^{d} \frac{\mathrm{d}x_j}{\mathrm{d}u_j} = \prod_{j=1}^{d} f'_{\mathrm{cond}}(u_j; \alpha_j) \tag{14}$$

- *Autoregressive flow* is essentially an autoregressive density model, *i.e.* the underlying structure is a complete DAG for a given topological order. Parameter α_j directly depends on $\mathbf{x}_{\mathrm{pa}(j)} = \mathbf{x}_{1:j-1}$:

$$\alpha_j = f_{\alpha_j}(\mathbf{x}_{1:j-1}) \tag{15}$$

When f_{α_j} is a MADE, the resulting model is masked autoregressive flow (MAF) Papamakarios et al. (2017).

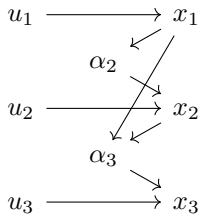
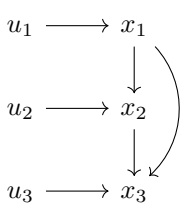
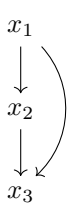
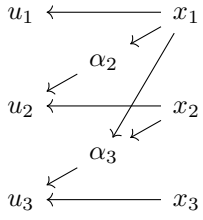
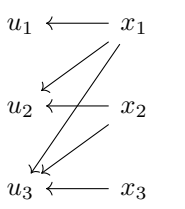
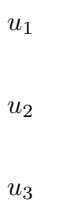
direction	transform	full	partial	reduced	comment
$\mathbf{u} \xrightarrow{f} \mathbf{x}$	$\alpha_j = f_{\alpha_j}(\mathbf{x}_{1:j-1})$ $x_j = f_{\mathrm{cond}}(u_j; \alpha_j)$				sampling: slow
$\mathbf{u} \xleftarrow{f^{-1}} \mathbf{x}$	$\alpha_j = f_{\alpha_j}(\mathbf{x}_{1:j-1})$ $u_j = f^{-1}_{\mathrm{cond}}(x_j; \alpha_j)$				evaluation: fast

Table 1: Autoregressive flow

- *Inverse autoregressive flow* (Kingma et al., 2016) also uses a complete DAG as the underlying structure, but the parameter α_j indirectly depends on $\mathbf{x}_{1:j-1}$ via $\mathbf{u}_{1:j-1}$:

$$\alpha_j = f_{\alpha_j}(\mathbf{u}_{1:j-1}) \tag{16}$$

IAF also uses MADE for f_{α_j}.

The computation graph of IAF is roughly the opposite of AF, so do their pros and cons. Since IAF has fast f and $\det(\mathsf{D}f)$, it is suitable for variational inference.

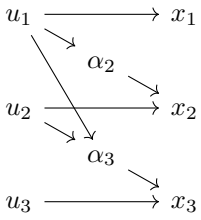
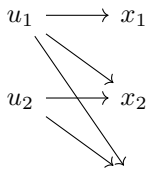
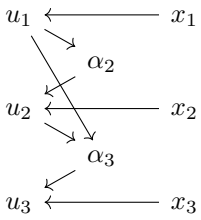
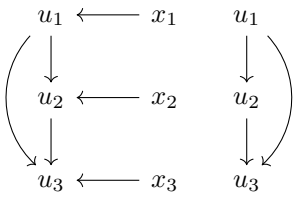
direction	transform	full	partial	reduced	comment
$\mathbf{u} \xrightarrow{f} \mathbf{x}$	$\alpha_j = f_{\alpha_j}(\mathbf{u}_{1:j-1})$ $x_j = f_{\text{cond}}(u_j; \alpha_j)$			x_1 x_2 x_3	sampling: fast
$\mathbf{u} \xleftarrow{f^{-1}} \mathbf{x}$	$\alpha_j = f_{\alpha_j}(\mathbf{u}_{1:j-1})$ $u_j = f^{-1}_{\text{cond}}(x_j; \alpha_j)$				evaluation: slow

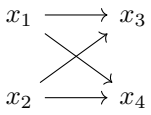
Table 2: Inverse autoregressive flow

- *RealNVP* (Dinh et al., 2017) splits variables into two groups $\mathbf{x} = (\mathbf{x}_L, \mathbf{x}_R)$, where $\mathbf{x}_L$ is set to be the noise and only $\mathbf{x}_R$ goes through nontrivial transformation:

$$x_j = \begin{cases} u_j & j \in L \\ f_{\text{cond}}(u_j; \alpha_j) & j \in R \end{cases} \tag{17}$$

$$\frac{\mathrm{d}x_j}{\mathrm{d}u_j} = \begin{cases} 1 & j \in L \\ f'_{\text{cond}}(u_j; \alpha_j) & j \in R \end{cases} \tag{18}$$

The underlying BN structure is a complete bipartite graph between $\mathbf{x}_L$ and $\mathbf{x}_R$:



RealNVP inherits both AF and IAF: since $\mathbf{x}_L = \mathbf{u}_L$, we have $f_{\alpha_r}(\mathbf{u}_L) = f_{\alpha_r}(\mathbf{x}_L)$, hence one can choose the direction that makes computation fast. As a result, both sampling and evaluation is fast, at the cost of less flexible f.

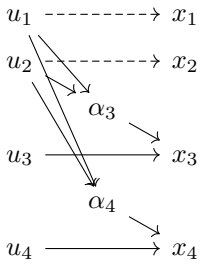
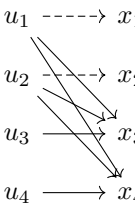
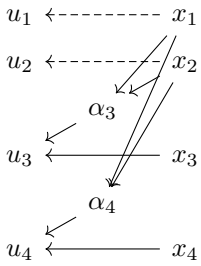
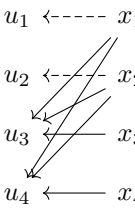
direction	transform	full	partial	reduced	comment
$\mathbf{u} \xrightarrow{f} \mathbf{x}$	$x_l = u_l$ $\alpha_r = f_{\alpha_r}(\mathbf{u}_L) = f_{\alpha_r}(\mathbf{x}_L)$ $x_r = f_{\text{cond}}(u_r; \alpha_r)$			x_1 x_2 x_3 x_4	sampling: fast
$\mathbf{u} \xleftarrow{f^{-1}} \mathbf{x}$	$u_l = x_l$ $\alpha_r = f_{\alpha_r}(\mathbf{u}_L) = f_{\alpha_r}(\mathbf{x}_L)$ $u_r = f^{-1}_{\text{cond}}(x_r; \alpha_r)$			u_1 u_2 u_3 u_4	evaluation: fast

Table 3: RealNVP

References

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. In *International Conference on Learning Representations*, 2017.

M. Germain, K. Gregor, I. Murray, and H. Larochelle. MADE: Masked Autoencoder for Distribution Estimation. In *International Conference on Machine Learning*, 2015.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved Variational Inference with Inverse Autoregressive Flow. In *Advances in Neural Information Processing Systems*, 2016.

G. Papamakarios, T. Pavlakou, and I. Murray. Masked Autoregressive Flow for Density Estimation. In *Advances in Neural Information Processing Systems*, 2017.

D. J. Rezende and S. Mohamed. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*, 2015.