Neural Inverse CDF Sampling

Andrew Stirn: andrew.stirn@columbia.edu
Tony Jebara: jebara@cs.columbia.edu

The Problem

Variational Bayesian inference frequently relies on Monte Carlo (MC) integration when intractable expectations appear in the evidence lower bound (ELBO). Variational inference evaluates these expectations wrt the proposed posterior approximations. In many cases, such as in Variational Autoencoders, we maximize the ELBO via gradient methods, however, to do so in conjunction MC integration requires that the samples from the posterior approximations are differentiable wrt their respective parameters. Closed-form "reparameterization tricks" unfortunately only exist for a subset of distribution families (e.g. Gaussian).

Inverse CDF Sampling

For continuous univariate distributions, inverse CDF sampling allows us to reparameterize a sample from a Uniform distribution to the target distribution:

$$\epsilon \sim \text{Uniform}(0,1)$$
$$z = F^{-1}(\epsilon;\, \theta)$$

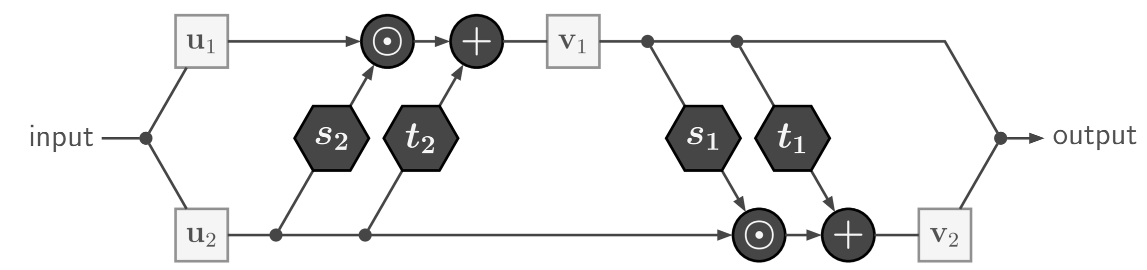
Unfortunately, not all continuous univariate distributions have a closed form quantile function, F^{-1}. In these cases, $\frac{\partial z}{\partial \theta}$ does not exist, prohibiting gradient methods.

Our Contributions

We leverage invertible neural network (INN) layers [1, 2] to approximate continuous univariate CDF's, allowing for inverse CDF sampling while ensuring the existence of $\frac{\partial z}{\partial \theta}$. Because the INN blocks are analytically invertible, our approximation maintains *monotonicity* and *one-to-oneness* ensuring: $z = \hat{F}^{-1}(\hat{F}(z;\theta);\theta)$.

Invertible Neural Network: Affine Coupling Layer

Forward Operation [3]

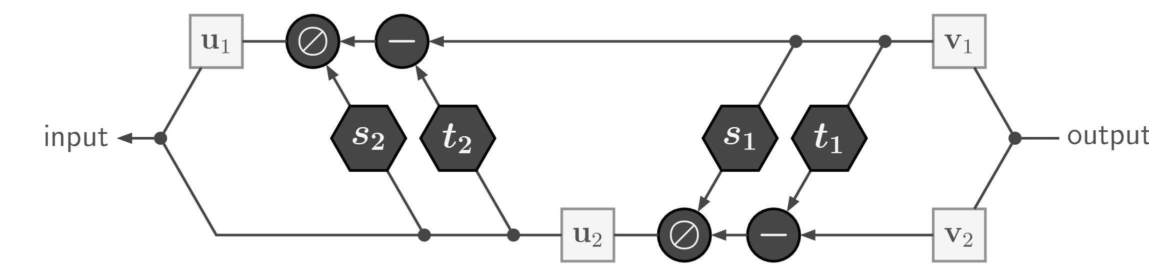


Bifurcate Input $= [\boldsymbol{u_1}, \boldsymbol{u_2}]$

$$\boldsymbol{v_1} = \boldsymbol{u_1} \odot \exp(s_2(\boldsymbol{u_2})) + t_2(\boldsymbol{u_2})$$
$$\boldsymbol{v_2} = \boldsymbol{u_2} \odot \exp(s_1(\boldsymbol{v_1})) + t_1(\boldsymbol{v_1})$$

Recombine Output $= [\boldsymbol{v_1}, \boldsymbol{v_2}]$

Inverse Operation [3]



Bifurcate Output $= [\boldsymbol{v_1}, \boldsymbol{v_2}]$

$$\boldsymbol{u_2} = (\boldsymbol{v_2} - t_1(\boldsymbol{v_1})) \odot \exp(-s_1(\boldsymbol{v_1}))$$
$$\boldsymbol{u_1} = (\boldsymbol{v_1} - t_2(\boldsymbol{u_2})) \odot \exp(-s_2(\boldsymbol{u_2}))$$

Recombine Input $= [\boldsymbol{u_1}, \boldsymbol{u_2}]$

The main requirement of this layer is that $\dim(\boldsymbol{u_i}) = \dim(\boldsymbol{v_i})$, such that, when cascading these layers, dimensions must be fixed and identical across layers. The main benefit of this layer is that invertibility is maintained even when s_i and t_i are non-invertible functions (i.e. affine + ReLU).

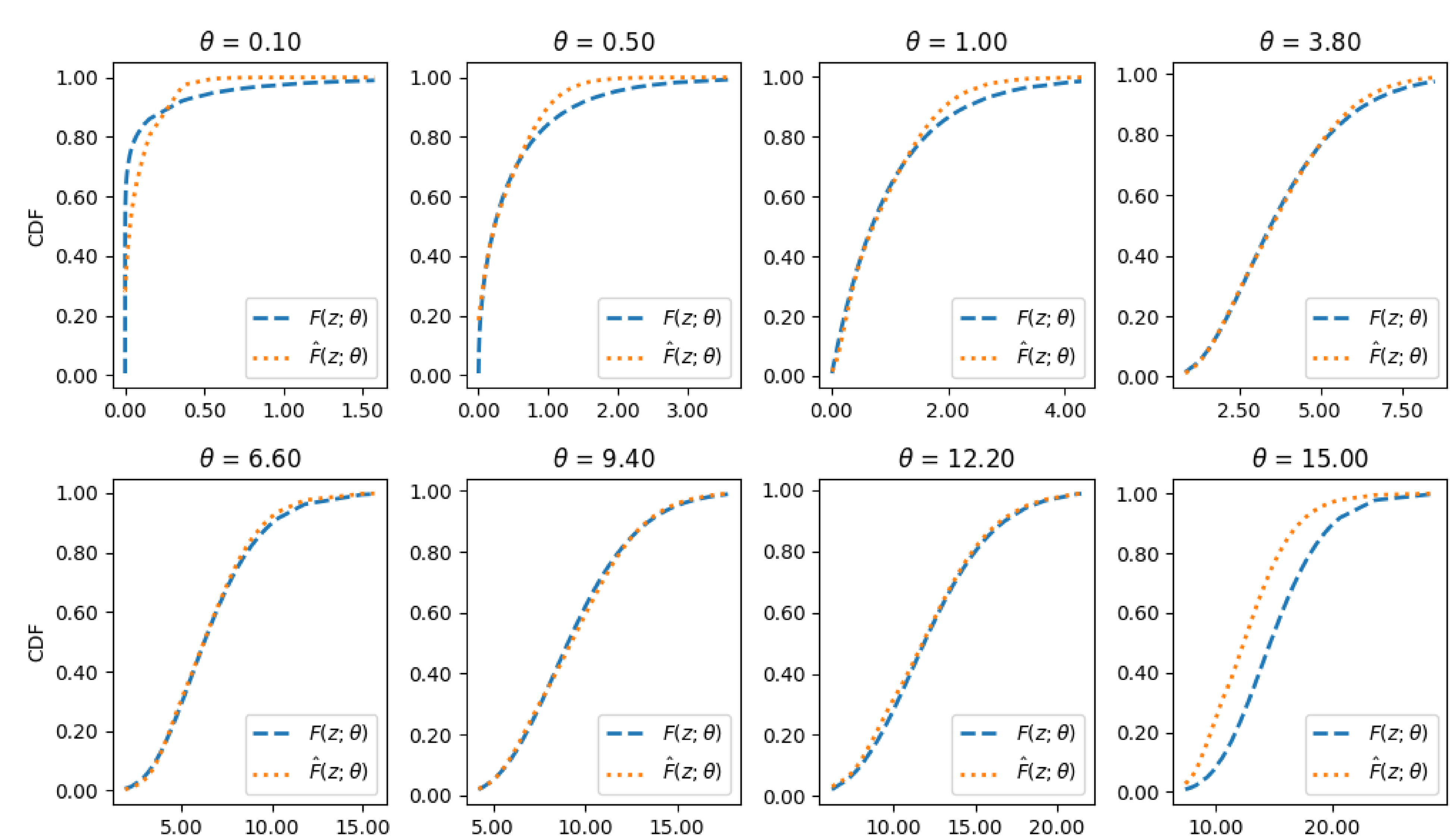
INN Modifications

Replacing $\exp(\cdot)$ with $b^{(\cdot)}$, improves numerical stability ($b < e$) and maintains invertibility. Let $f_i(x) = \text{elu}(W_{f_i} \cdot x + b_{f_i})$, where $f \in \{s, t\}$, and W_{f_i} and b_{f_i} are the outputs of multi-layer perceptrons operating on the distribution parameters θ. Define loss as $|F(z;\, \theta) - \hat{F}(z;\, \theta)|$.

Gamma CDF Training Batch

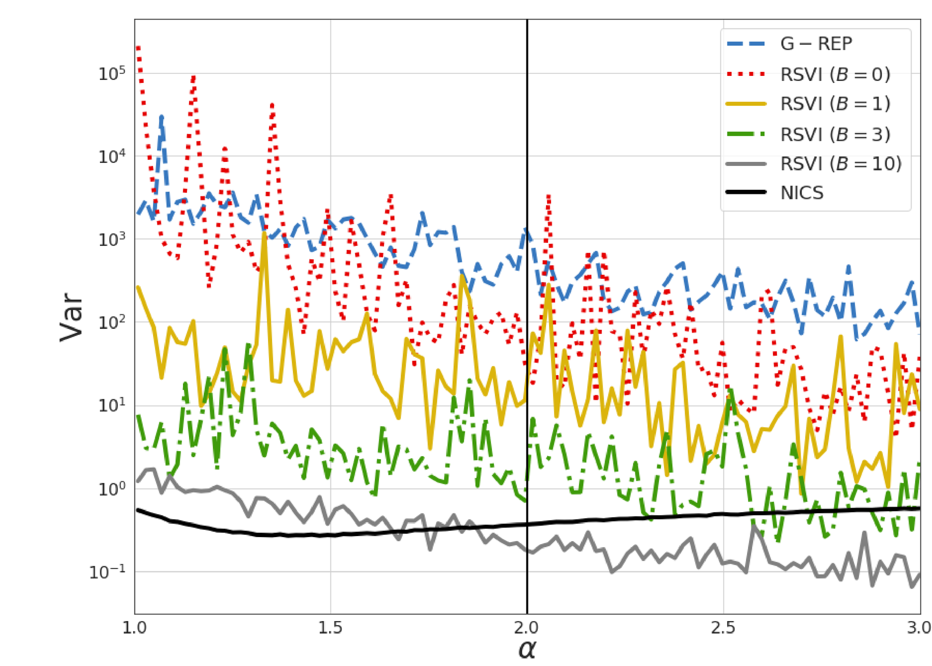
Initialize batch $\mathcal{B} = \{\emptyset\}$
for $n \in \{1, \dots, N_{\text{batch}}\}$ **do**
  Sample: $\theta_n \overset{iid}{\sim} p(\theta)$
  for $i \in \{1, \dots, N_z\}$ **do**
    Sample: $z_i \overset{iid}{\sim} \text{Gamma}(\theta_n, 1)$
    Append: $\mathcal{B} := \mathcal{B} \cup (z_i, \theta_n, F(z_i;\, \theta_n))$
  end for
end for

CDF Approximation for Gamma(θ,1) Distribution: $\theta \in (0, 15]$



Dirichlet Approximation

A Dirichlet($\boldsymbol{\alpha}$) sample can be generated by normalizing Gamma(α_k, 1) samples. We compare the variance of the gradient of our method to orther reparameterization methods [4] using their code [5].



Autoencoding LDA [7]

Perplexity scores after replacing the softmax-basis sampler [6] with ours.

	Softmax	Our Method
NVLDA	1099.96	1103.77
prodLDA	1161.69	1099.94

References & Code

[1] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," *CoRR*, vol. abs/1605.08803, 2016. [Online]. Available: http://arxiv.org/abs/1605.08803

[2] L. Ardizzone, J. Kruse, C. Rother, and U. KÃűthe, "Analyzing inverse problems with invertible neural networks," in *International Conference on Learning Representations*, 2019.

[3] V. L. L. Heidelberg, "Analyzing inverse problems with invertible neural networks," 2018. [Online]. Available: https://hci.iwr.uni-heidelberg.de/vislearn/inverse-problems

[4] C. A. Naesseth, F. J. R. Ruiz, S. W. Linderman, , and D. M. Blei, "Reparameterization gradients through acceptance-rejection sampling algorithms," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[5] B. Lab, "Ars reparameterization," 2017. [Online]. Available: https://github.com/blei-lab/ars-reparameterization/

[6] A. Srivastava, "Autoencoding variational inference for topic models," 2017. [Online]. Available: https://github.com/akashgit/autoencoding_vi_for_topic_models

[7] C. S. Akash Srivastava, "Autoencoding variational inference for topic models," in *International Conference on Learning Representations*, 2017.

Source code available at: `https://github.com/astirn/neural-inverse-cdf-sampling`