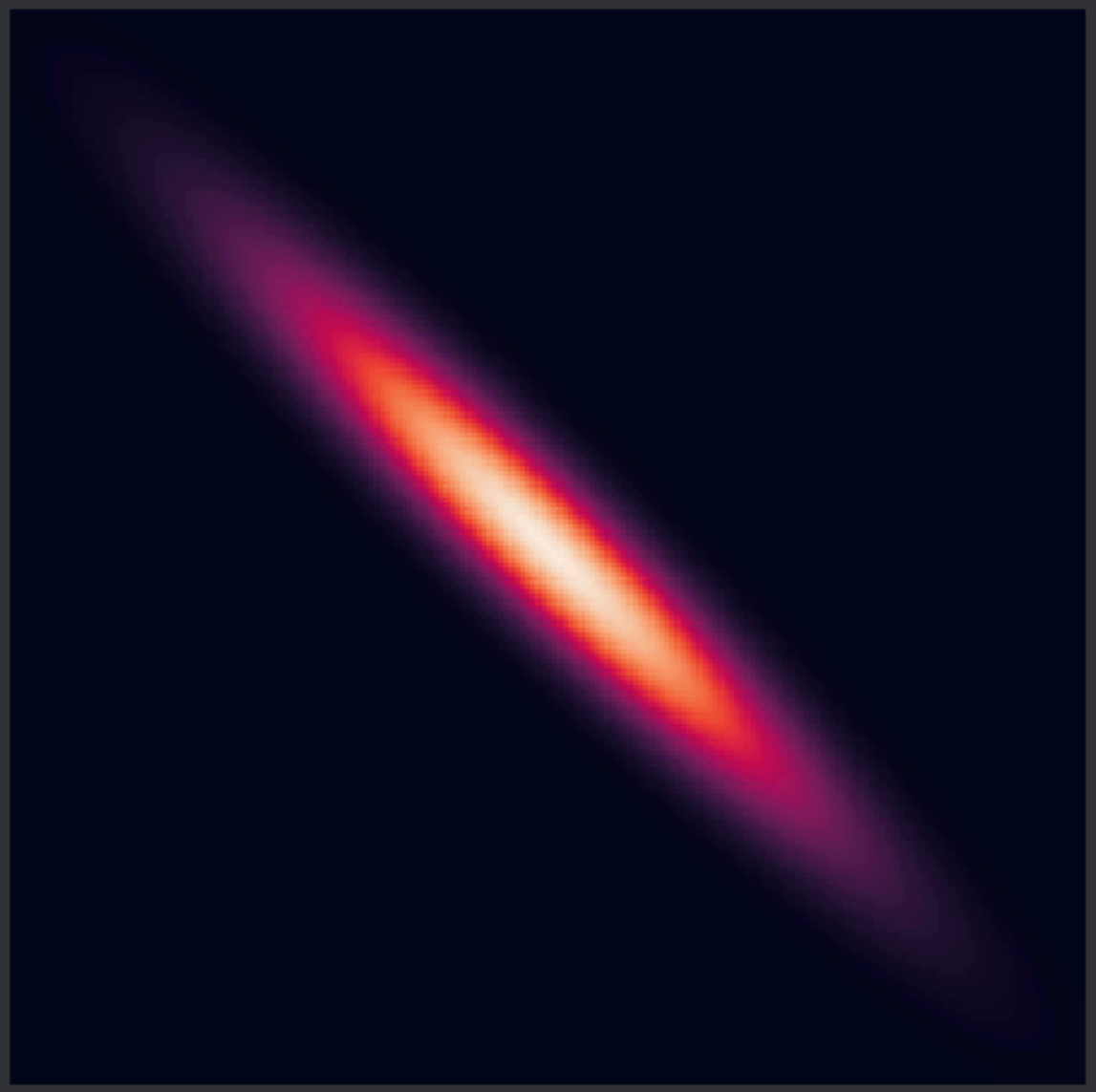
NeuTra-lizing Bad Geometry in HMC with Neural Transport

Matthew D. Hoffman*, Pavel Sountsov*,
Joshua Dillon, Ian Langmore, Dustin Tran, Srinivas Vasudevan

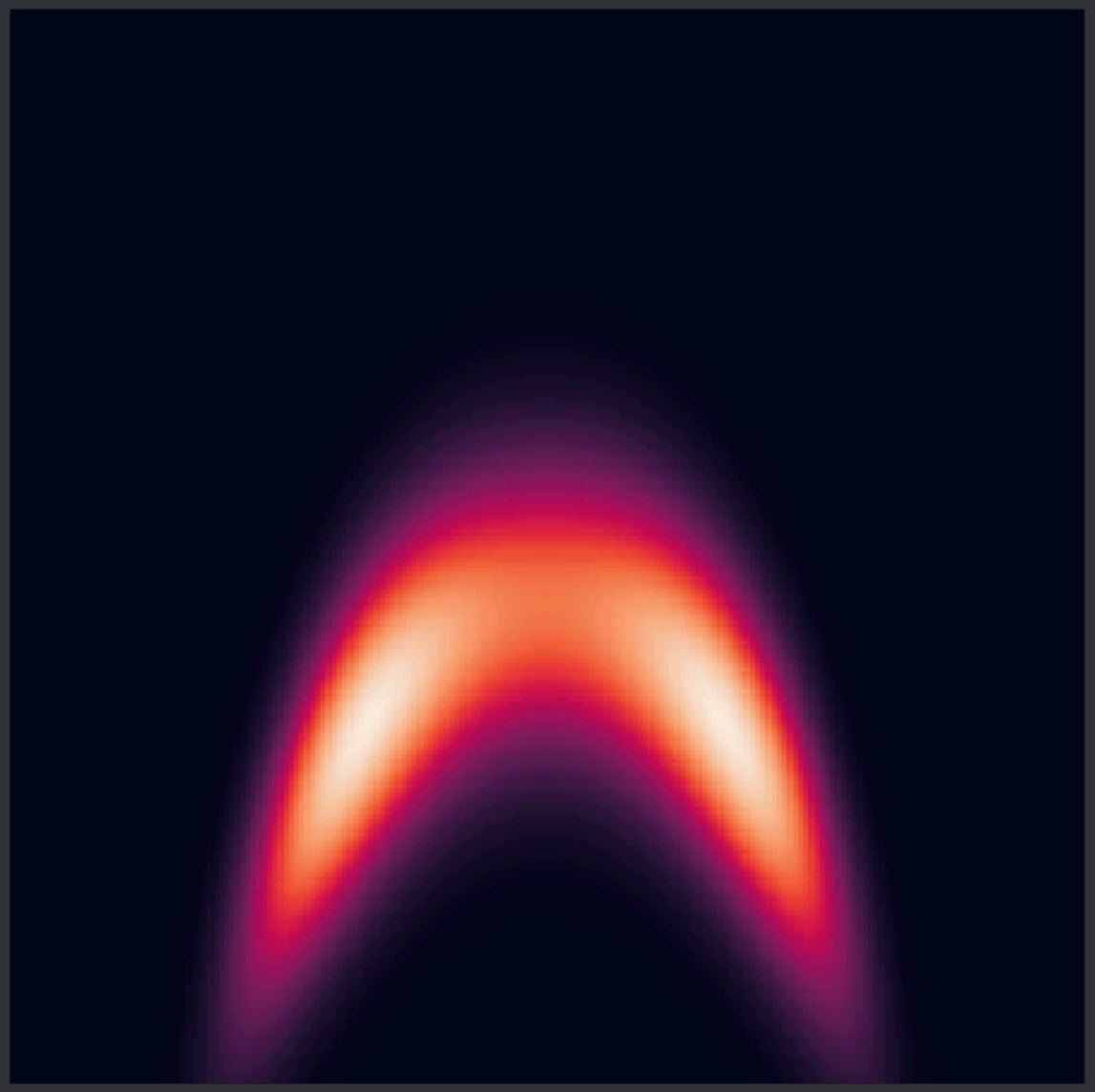
Google AI

* joint first authors

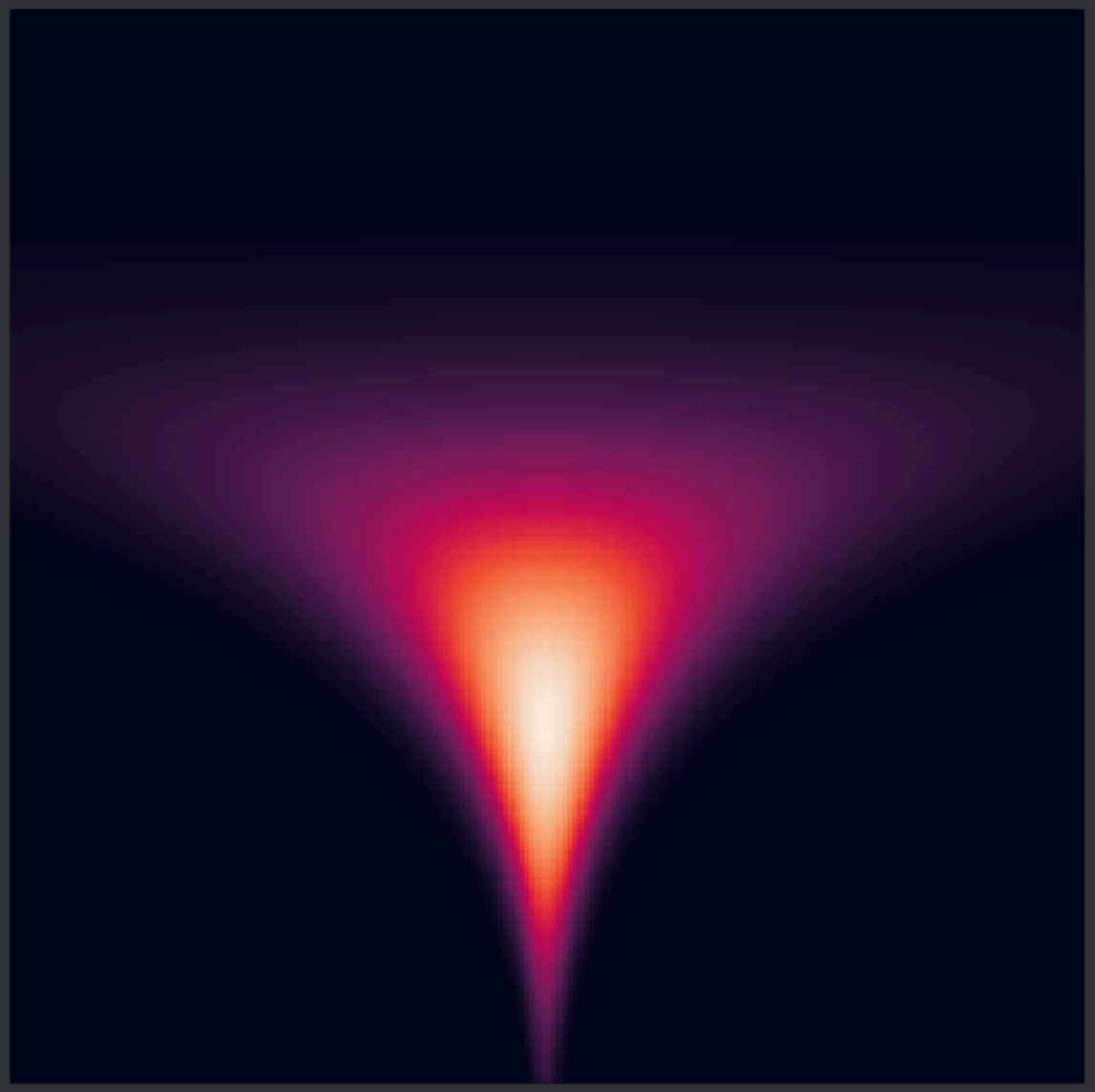
Which target distribution is easier for HMC to sample from?



Which target distribution is easier for HMC to sample from?



Which target distribution is easier for HMC to sample from?



Transport-Map MCMC

(Parno et al., 2014, Marzouk et al. 2016)

Consider a bijective map

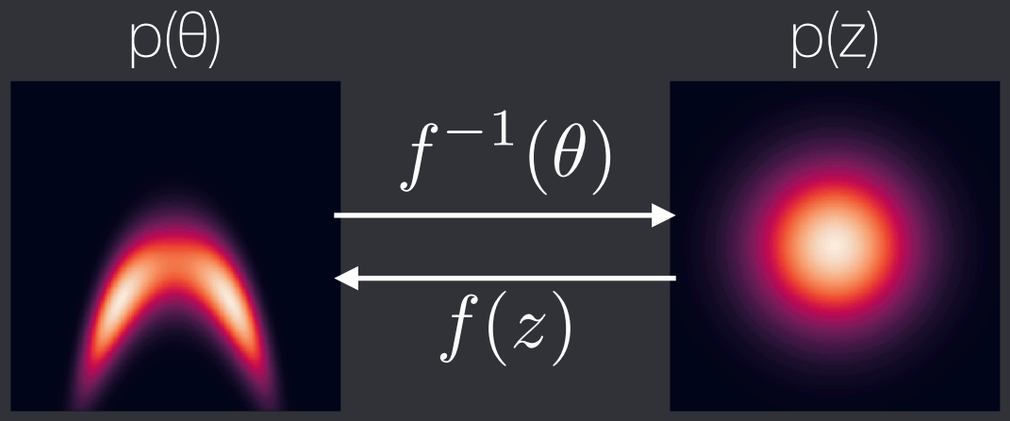
$$f(z) = \theta; \quad \theta = f^{-1}(z).$$

By the change-of-variables formula,

$$p(z) = p(\theta) \left| \frac{df}{dz} \right|.$$

Try to tune the parameters of f so that

$$p(z) \approx \mathrm{Normal}(z; 0, I).$$

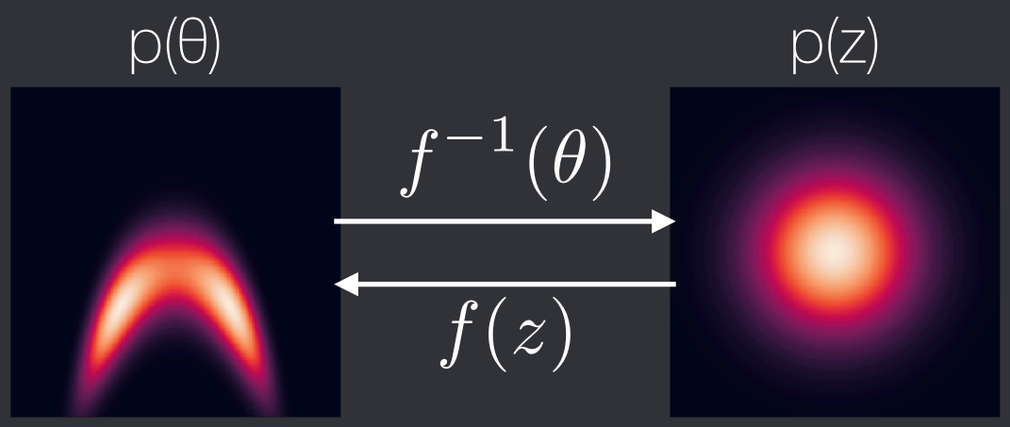


Transport-Map MCMC

(Parno et al., 2014, Marzouk et al. 2016)

Strategy:

1. Fit map f(z) so that p(z) ≈ Normal(z; 0, I).
2. Run MCMC on p(z).
3. Push z samples forward through f to get samples from p(θ).



Key Decisions

1. What form should we choose for f(z)? It needs to be
   - Flexible.
   - Bijective.
   - Tractable. (Computing θ and $\log \left| \frac{df}{dz} \right|$ needs to be efficient.)
2. How do we formalize “p(z) ≈ Normal(z; 0, I)”?
3. What sampler should we use?

Neural Transport (NeuTra) HMC

We use **Inverse autoregressive flows** (IAF; Kingma et al., 2016) trained using **stochastic variational inference** (SVI; Hoffman et al., 2013) and **Hamiltonian Monte Carlo** (HMC; Neal, 2011). Why?

- SVI is fast and stable.
- IAFs are powerful neural transport maps with an efficient ELBO estimator.
- HMC is one of the only samplers that's efficient in hard, high-dimensional problems.

Variational Inference

Let

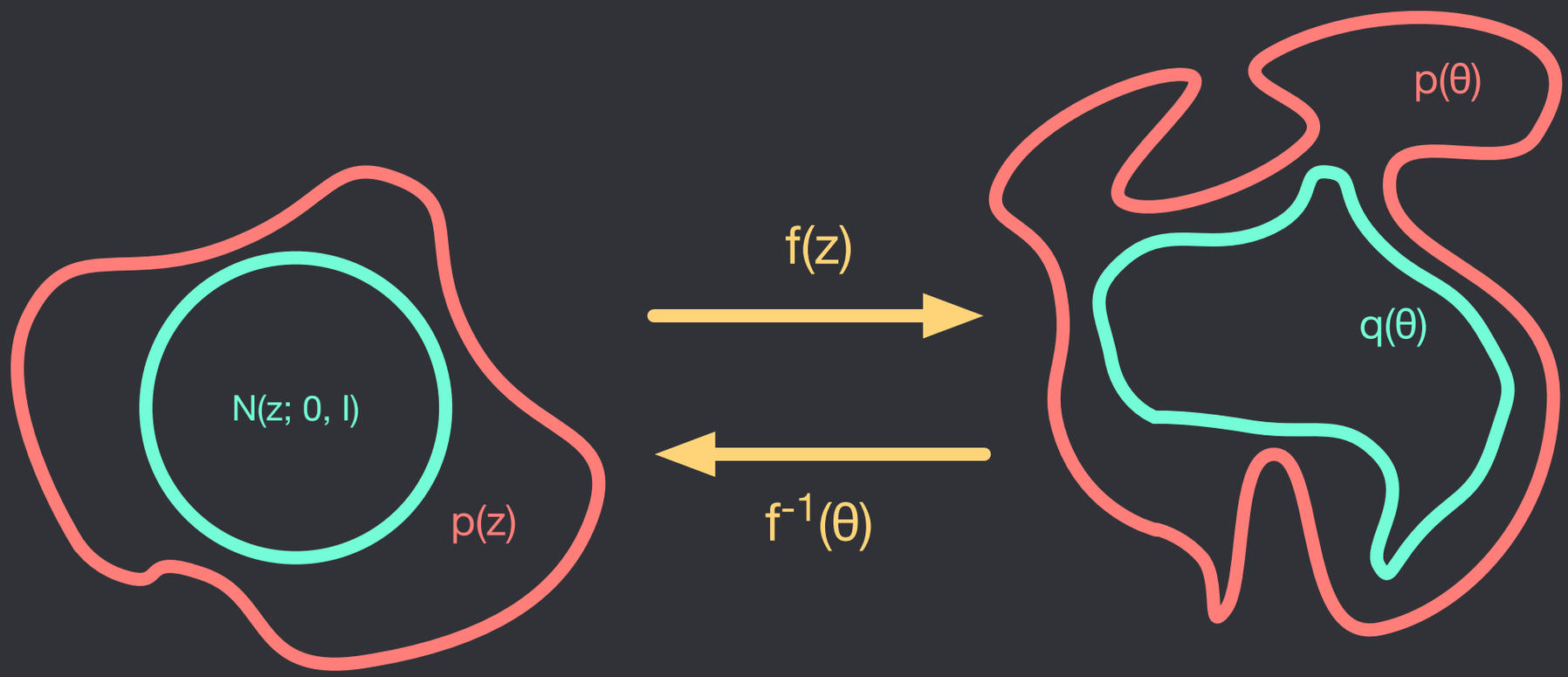
$$q(z) \triangleq \mathrm{Normal}(z; 0, I); \quad q(\theta) \equiv \mathrm{Normal}(f^{-1}(\theta); 0, I) \left| \frac{\partial f^{-1}}{\partial \theta} \right|^{-1}.$$

then

$$\mathbf{KL}(\mathrm{Normal}(0, I) \mid\mid p(z)) = \mathbf{KL}(q(\theta) \mid\mid p(\theta)).$$

So maximizing the standard ELBO to make q(θ) ≈ p(θ)

will make q(z) = Normal(z; 0, I) ≈ p(z).



Inverse Autoregressive Flows

An IAF takes the form

$$\theta_d \triangleq \mu_d(z_{1:d-1}) + z_d e^{\sigma_d(z_{1:d-1})}; \quad z_d = (\theta_d - \mu_d(z_{1:d-1}))e^{-\sigma_d(z_{1:d-1})}$$

where μ and σ are parameterized by a neural net.

Every element of θ can be computed directly from z.

The log-determinant of the Jacobian is just

$$\log \left|\frac{d\theta}{dz}\right| = \sum_d \sigma_d(z_{1:d-1}).$$

That's all we need to do SVI.

Stacking IAFs yields a more flexible family of transformations.

Hamiltonian Monte Carlo

HMC is a powerful MCMC sampler that uses gradient information. It generates proposals by simulating the Hamiltonian dynamics

$$\frac{\partial \theta}{\partial t} = m; \quad \frac{\partial m}{\partial t} = \nabla_\theta \log p(\theta).$$

HMC is known to mix very quickly on strongly log-concave target distributions (Mangoubi&Smith, 2017). It's often the only thing that works in high dimensions.

But it's sensitive to conditioning (just like SGD). And it's not geometrically ergodic on distributions with heavy or light tails (Livingstone et al., 2016).

Hamiltonian Monte Carlo in a Transformed Space

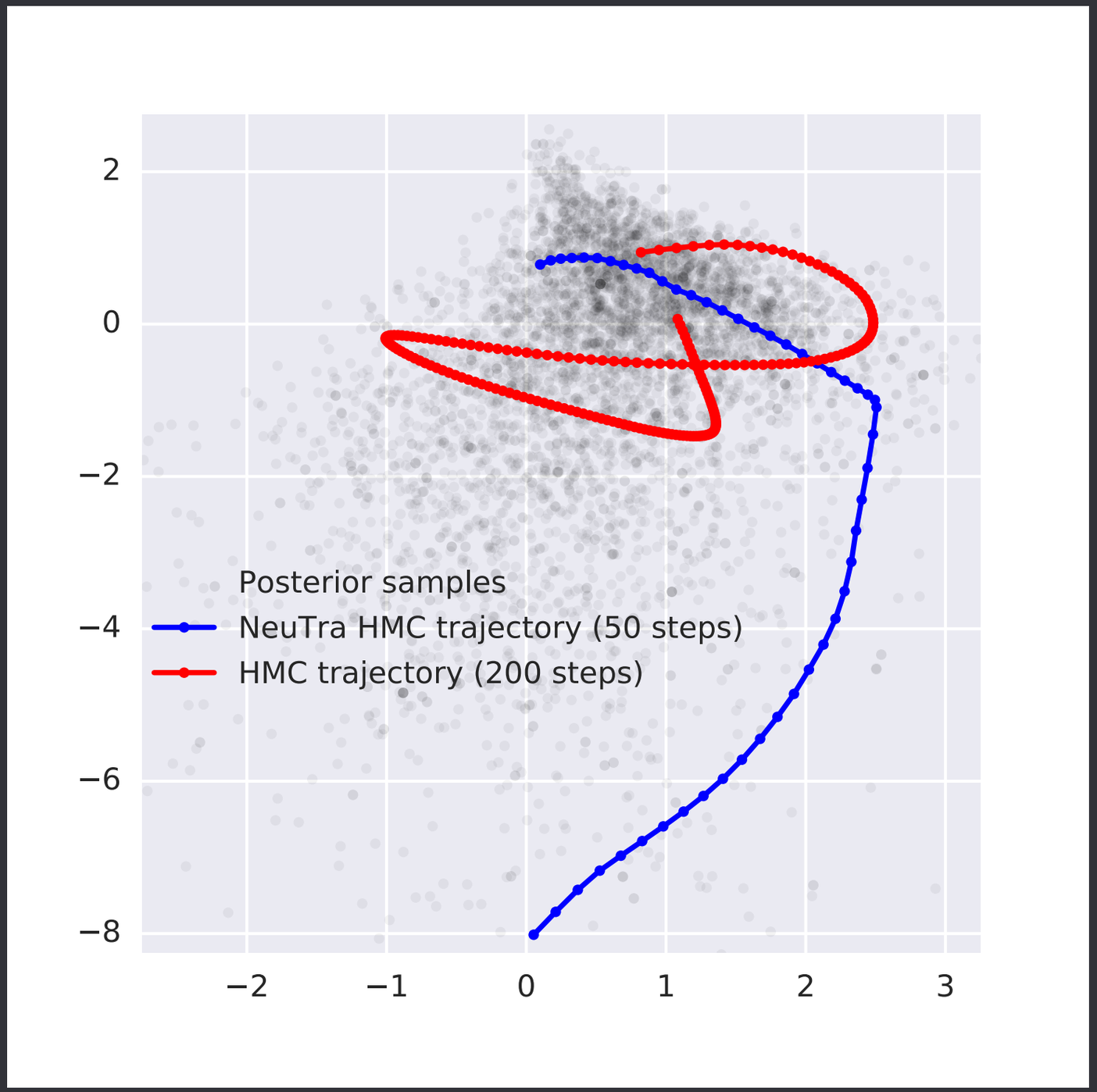
In continuous time, pushing the Hamiltonian dynamics in z-space forward to θ-space yields Hamiltonian dynamics on a Riemannian manifold with metric

$$G(\theta) = (JJ^\top)^{-1}; \quad J \triangleq \frac{\partial f}{\partial z}.$$

So NeuTra HMC is like Riemannian Manifold HMC (Girolami&Calderhead, 2011) except that

- We don't need finicky implicit integration schemes.
- We don't need to derive, implement, and compute Fisher matrices.
- We don't need to consider prior and likelihood separately.
- We do need to learn the metric.

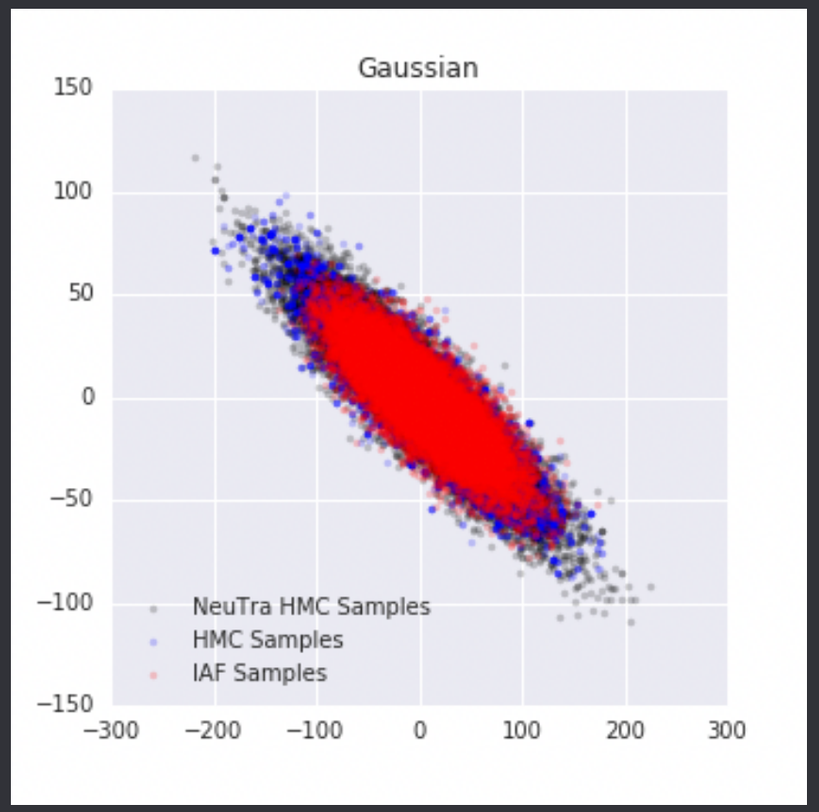
Example Trajectory

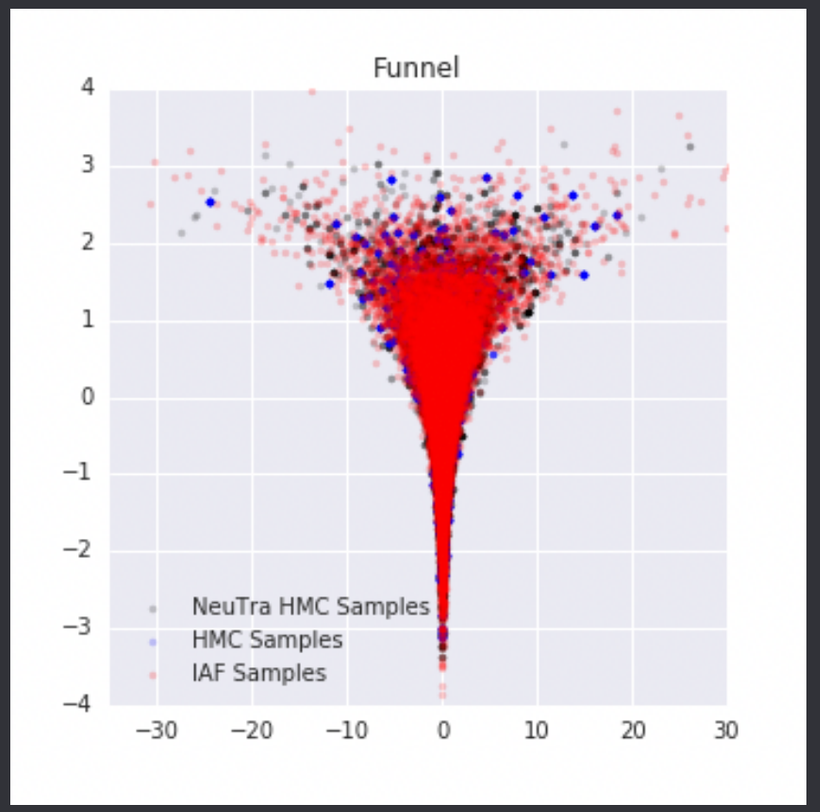


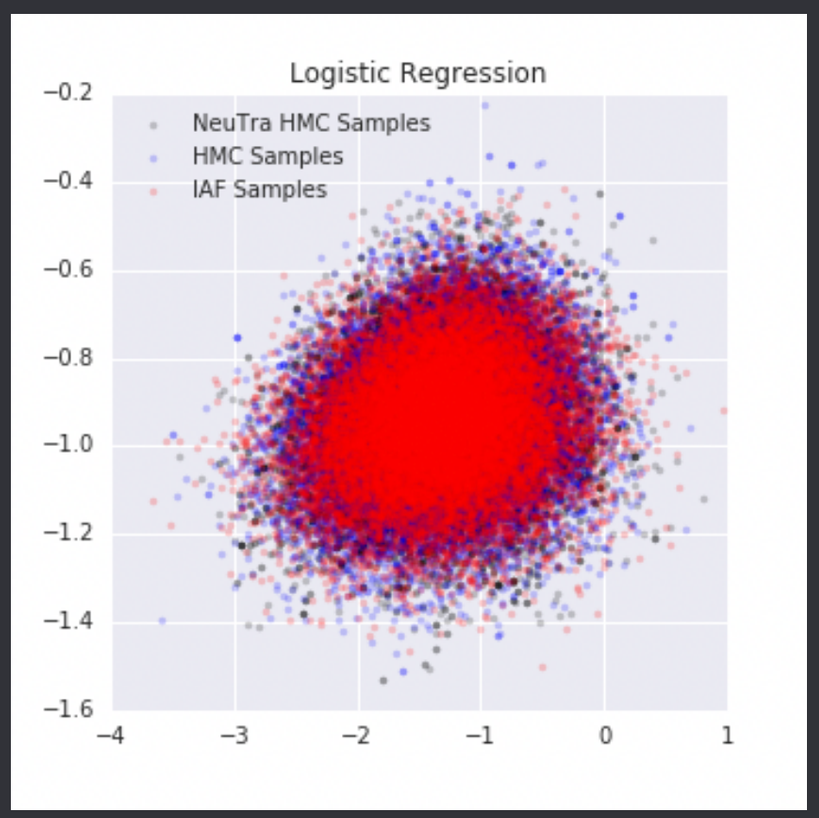
Some Qualitative Results

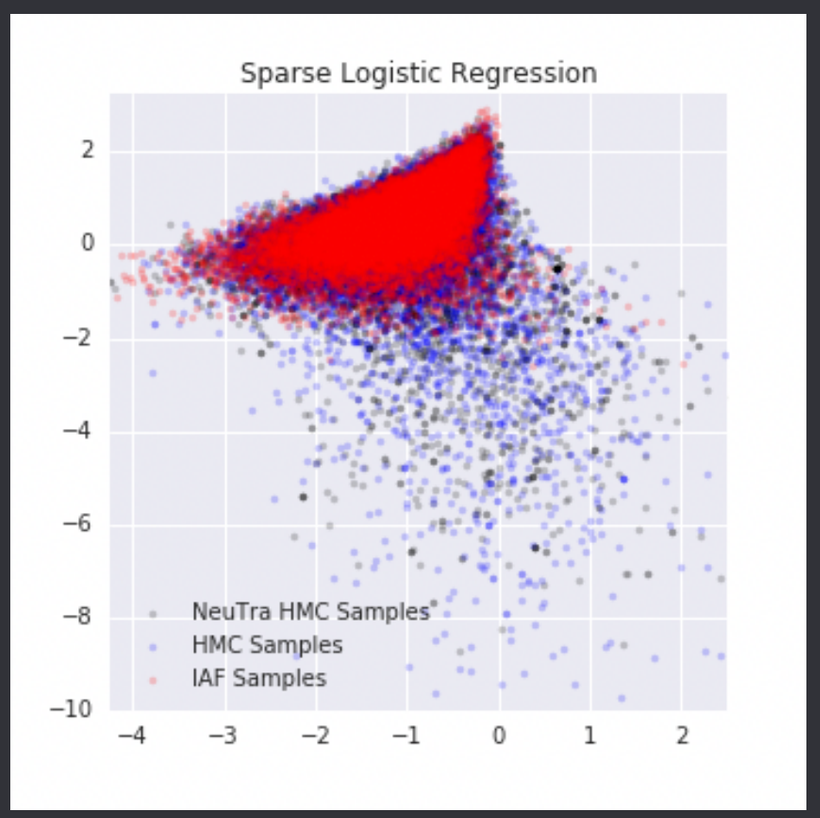
IAF approximation is good, but not perfect.

Vanilla HMC and NeuTra HMC results are qualitatively similar.

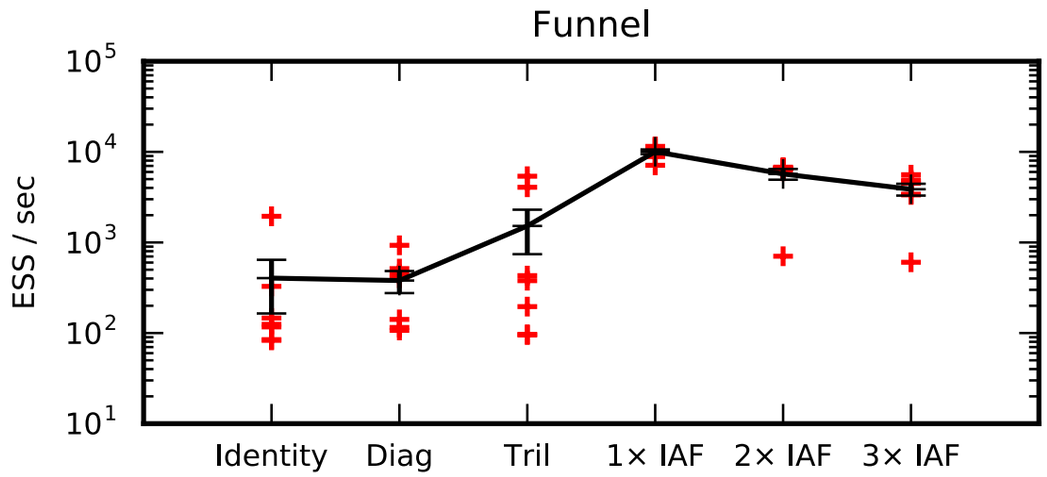


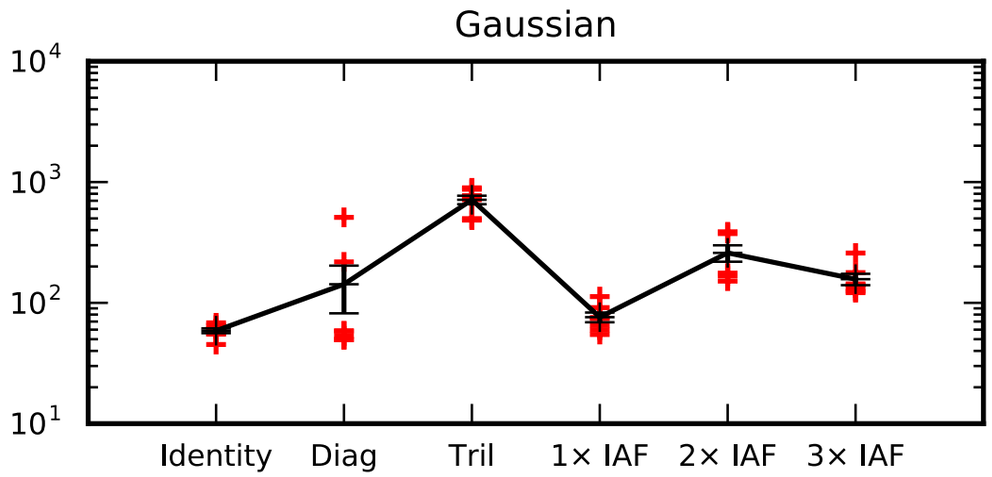


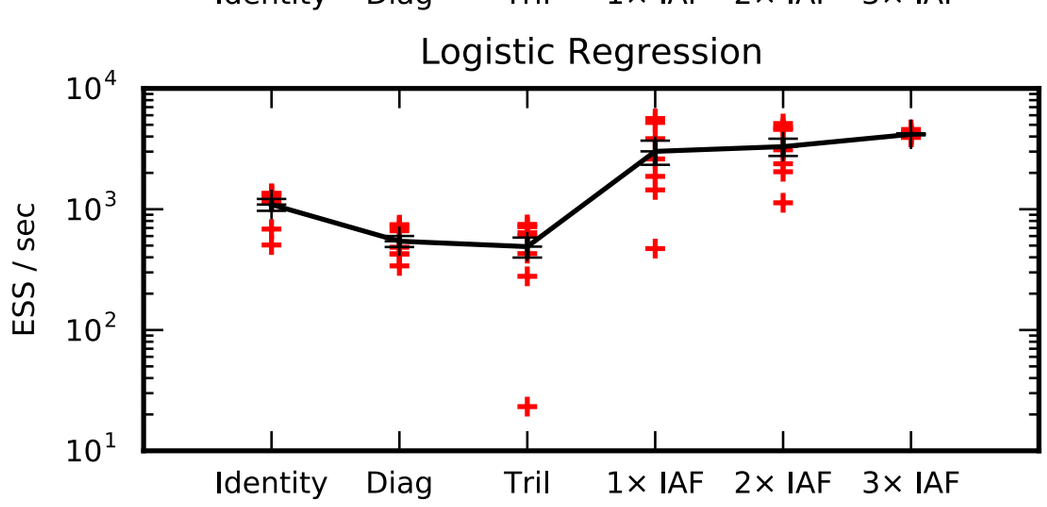


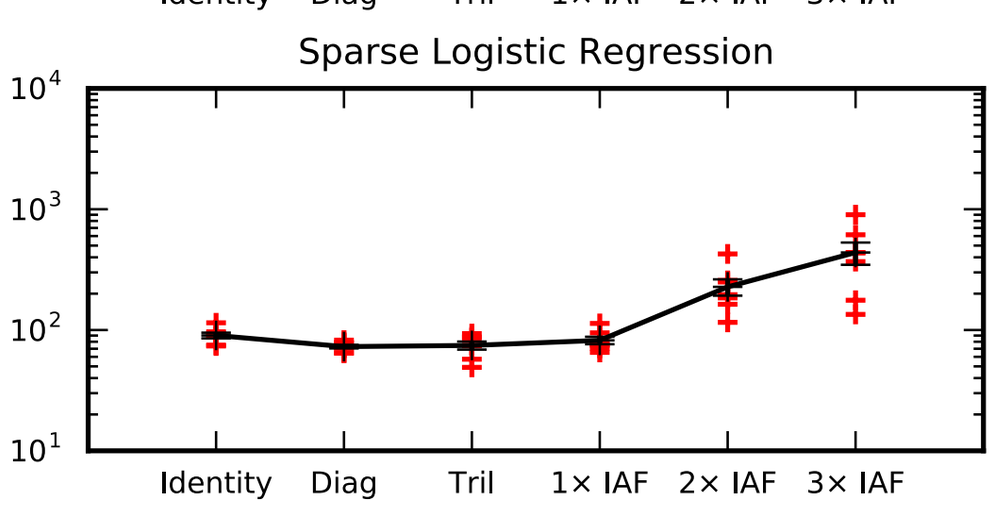


Effective Sample Size / Second









Identity: Vanilla HMC.

Diag/Tril: HMC with diagonal/full-rank linear preconditioning.

K x IAF: NeuTra with stack of K IAFs.

NeuTra HMC on a VAE

NeuTra plays well with amortized inference for latent-variable models too.

Training an MNIST VAE decoder with NeuTra HMC as in (Hoffman, 2017) gives good held-out NLLs:

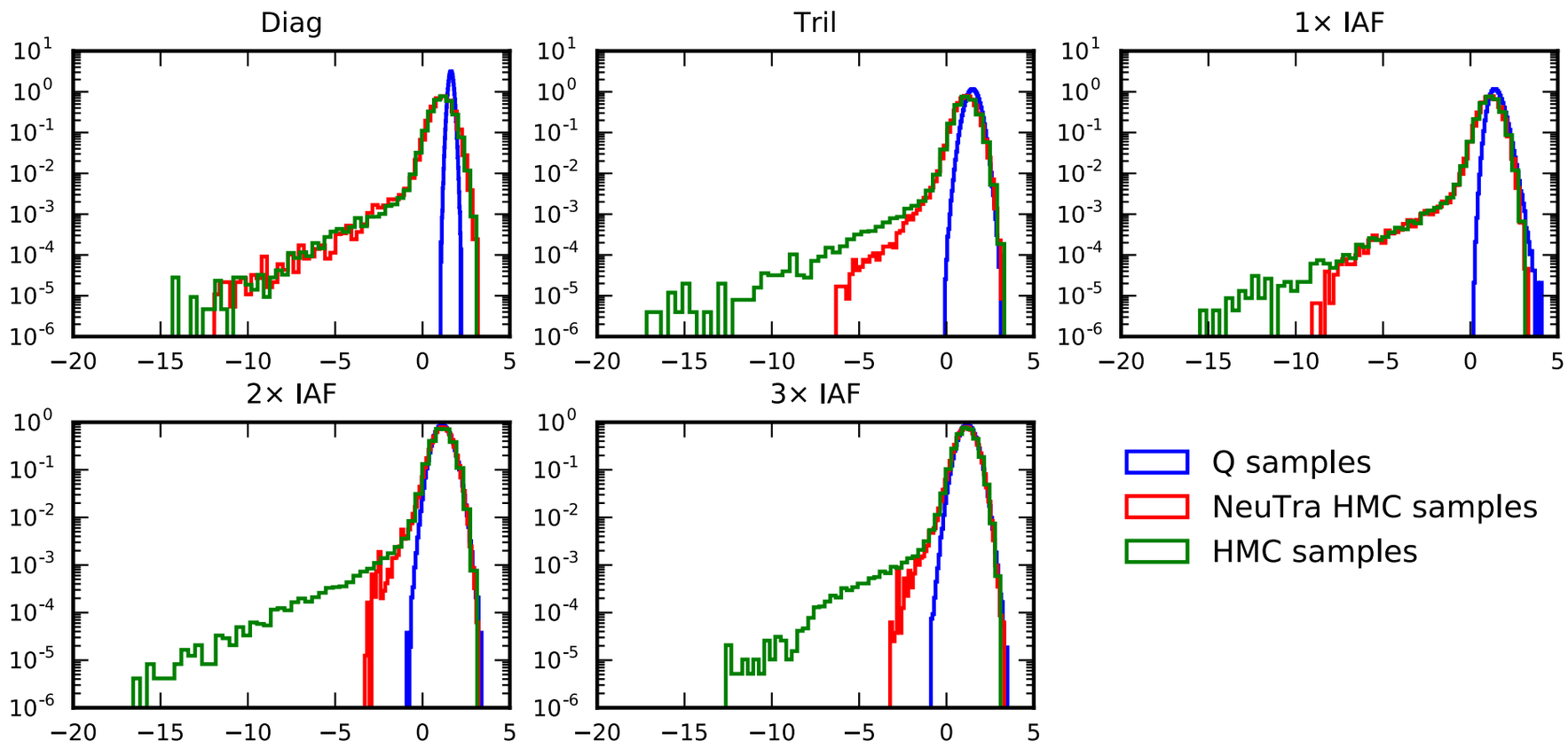
Posterior	$\log p(x)$
Independent Gaussian	80.84 ± 0.02
IAF	79.76 ± 0.03
IAF+NeuTra HMC (1 step)	79.54 ± 0.02
IAF+NeuTra HMC (2 steps)	79.42 ± 0.02
IAF+NeuTra HMC (4 steps)	79.35 ± 0.01

A Warning

Warping geometry can be dangerous! (Even linearly!)

If q(θ) doesn't get into the tails, the sampler might not either.

Solving this in general is a chicken-and-egg problem.



A Word on Relative Costs

Fitting the IAF is not free. When is it worth it?

- When we're drawing **lots of samples**, for example because we want to estimate credible intervals.
- When we're doing **amortized inference** in a latent-variable model, as in VAEs.
- When there's **lots of data** and we can get cheap unbiased gradients by subsampling observations.

Conclusion

- NeuTra uses modern variational inference techniques to speed up HMC.
  - Or if you prefer, it uses HMC to make variational inference more accurate.
- NeuTra offers many of the advantages of RMHMC with fewer drawbacks.
  - But (as with any MCMC method) be careful.
- It works with both full Bayes and amortized inference for latent-variable models.
- All the pieces you need to implement it are in TensorFlow Probability.