REINFORCing Concrete with REBAR

Low variance, unbiased gradient estimates for discrete latent variable models

George Tucker (gjt@google.com), Andriy Mnih, Chris J Maddison, Dieterich Lawson, Jascha Sohl-Dickstein

Github - goo.gl/fVJKy8

Abstract

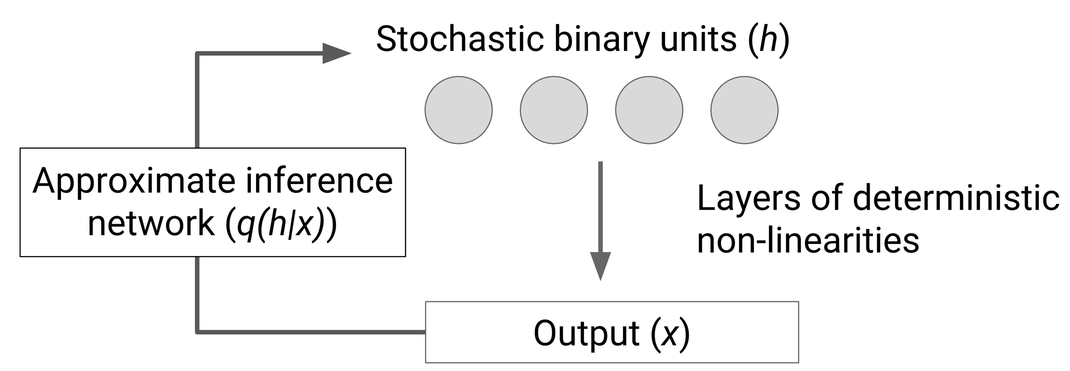
Learning in discrete latent variable models is challenging due to high variance gradient estimators. Generally, approaches have relied on control variates to reduce the variance of the REINFORCE estimator. Recent work (Jang *et al.* 2016; Maddison *et al.* 2016) has taken a different approach, introducing a continuous relaxation of discrete variables to produce low-variance, but biased, gradient estimates. **We combine the approaches through a novel control variate that produces low-variance, *unbiased* gradient estimates.**

Background

Fit a sigmoid belief network, *p(x, h)*, by maximizing a variational lower bound on the log-likelihood

$$E_{q(h|x)}\boxed{\left[\log \frac{p(x,h)}{q(h|x)}\right]}$$

f(x, h)



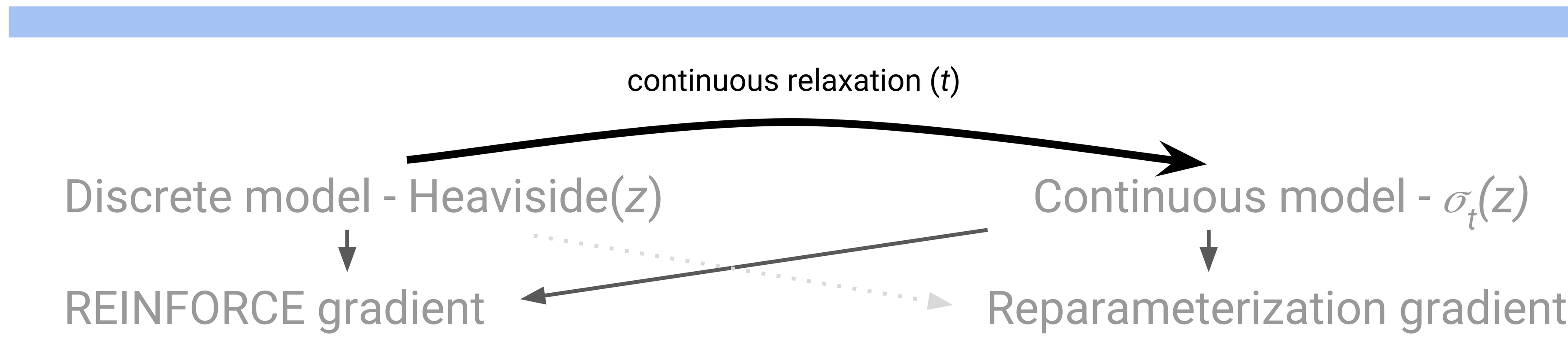
MuProp (Gu *et al.* 2016) estimates the gradient using control variates from a first order Taylor expansion.

Alternatively, Maddison *et al.* introduce the **Concrete** distribution

$$z := \log(\theta/(1-\theta)) + \log(u) - \log(1-u),$$

where $u \sim$ Uniform(0,1), $h = z > 0 \sim$ Bernoulli(θ). Relaxing the hard threshold to a tempered softmax ($\sigma_t(z) := \sigma(z/t)$), results in a differentiable reparameterizable approximate computation graph which is amenable to the reparameterization trick.

REBAR



We use the relaxed graph to decompose the objective into "hard" and"easy" terms:

$$E_{q(h|x)}[f(x,h)] = E_{q(h|x)}[f(x,h)] - E_{q(h|x)}E_{q(z|h,x)}[f(x,\sigma_t(z))]] + E_{q(z|x)}[f(x,\sigma_t(z))]$$

Use **REINFORCE**

Use **reparameterization trick**

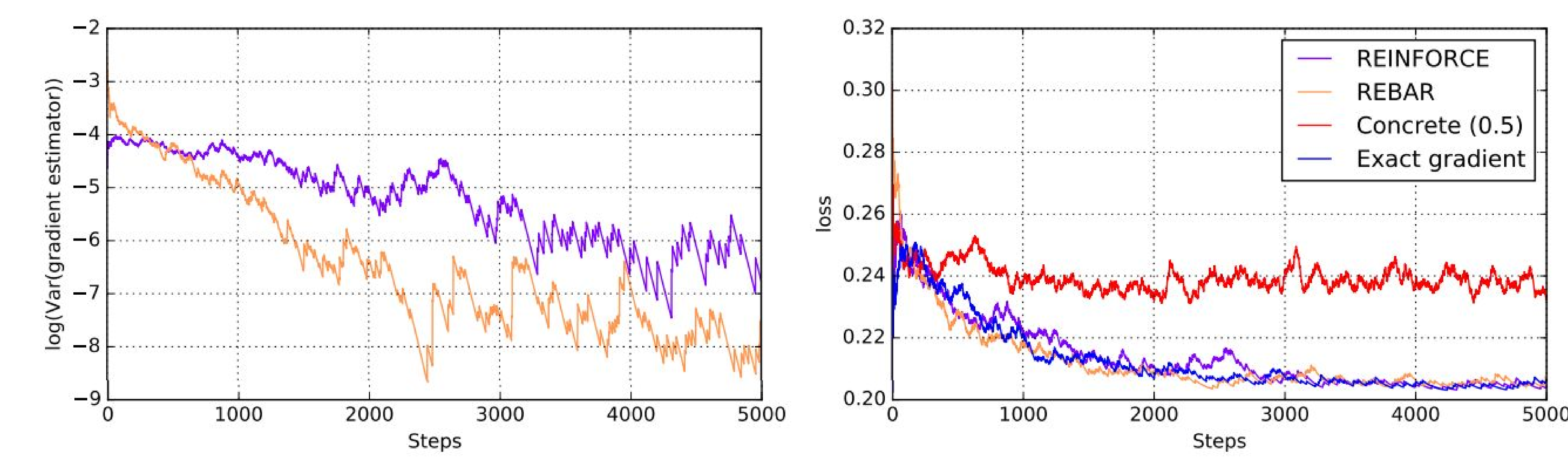
t adjusts the tightness of the relaxation, thus balancing sources of variance

REBAR is unbiased for *any* choice of *t*, so we can perform gradient descent on the variance to dynamically select the optimal *t*.

Results

Toy problem

Minimize E[(*b* - *t*)^2] p(*b*|θ) = Bernoulli(θ)



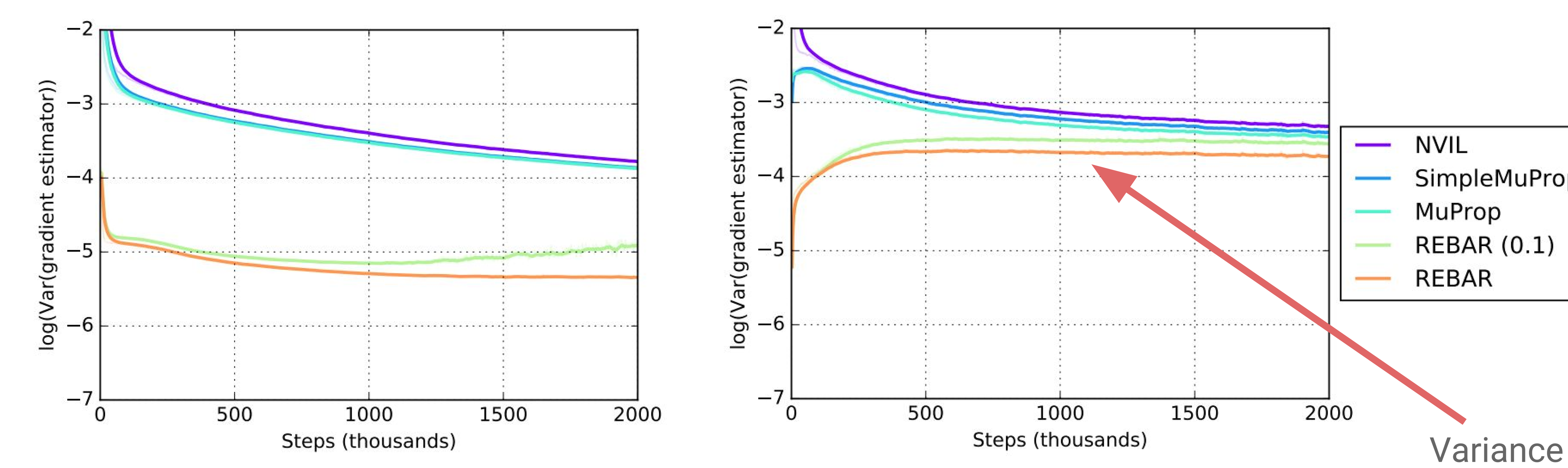
Concrete (Maddison 2016); Gumbel-Softmax (Jang 2016) optimize a continuous relaxation, which leads to low-variance, but ***biased*** gradients.

Relaxation "temperature" must be tuned to balance bias and variance.

Variance reduction for MNIST Generative Modeling

Task: Learning a density model of 28 x 28 binary MNIST images

Model: Binary stochastic layers followed by a single linear layer or 2x 200 unit tanh layers



Variance further reduced by online adaptation of t

2 stochastic layers, 200 nodes, linear

1 stochastic layer, 200 nodes, nonlinear

REBAR provides significant variance reduction over the previous state-of-the-art unbiased estimators. Dynamically adjusting the temperature during training reduces gradient variance even further. SimpleMuProp is a surprisingly strong baseline that REBAR reduces to in the large t limit.

Log likelihood lower bound

MNIST gen.	NVIL	MuProp	REBAR (0.1)	REBAR	Concrete (0.1)
Linear 1 layer	-112.5 ± 0.1	-111.7 ± 0.1	-111.7 ± 0.2	-111.6 ± 0.03	$\mathbf{-111.3 \pm 0.1}$
Linear 2 layer	-99.6 ± 0.1	-99.07 ± 0.02	-99 ± 0.1	$\mathbf{-98.8 \pm 0.03}$	-99.62 ± 0.09
Nonlinear	-102.2 ± 0.1	-101.5 ± 0.3	-101.4 ± 0.2	$\mathbf{-101.1 \pm 0.1}$	-102.8 ± 0.2
Omniglot gen.					
Linear 1 layer	-117.44 ± 0.03	-117.09 ± 0.06	-116.93 ± 0.03	$\mathbf{-116.83 \pm 0.02}$	-117.23 ± 0.04
Linear 2 layer	-109.98 ± 0.09	-109.55 ± 0.02	$\mathbf{-109.12 \pm 0.07}$	$\mathbf{-108.99 \pm 0.06}$	-109.95 ± 0.04
Nonlinear	-110.4 ± 0.2	-109.58 ± 0.09	-109 ± 0.1	$\mathbf{-108.72 \pm 0.06}$	-110.64 ± 0.08
MNIST struct. pred.					
Linear 1 layer	-69.15 ± 0.02	$\mathbf{-64.31 \pm 0.01}$	-65.75 ± 0.02	-65.244 ± 0.009	-65.53 ± 0.01
Linear 2 layer	-68.88 ± 0.04	-63.68 ± 0.02	-65.525 ± 0.004	$\mathbf{-61.74 \pm 0.02}$	-66.89 ± 0.04
Nonlinear layer	-54.01 ± 0.03	-47.58 ± 0.04	-47.34 ± 0.02	$\mathbf{-46.44 \pm 0.03}$	-47.09 ± 0.02

The reduction in gradient estimator variance translates into faster convergence to a better local optimum. Notably, the biased gradient estimator underperforms on the nonlinear model.