Where do we see this guy?

$\mathcal{L}(\theta) = \mathbb{E}_{p(b|\theta)}[f(b)]$

- Just about everywhere!
- Variational Inference
- Reinforcement Learning
- Hard Attention
- And so many more!
Gradient based optimization

- Gradient based optimization is the standard method used today to optimize expectations.
- Necessary if models are neural-net based.
- Very rarely can this gradient be computed analytically.
Otherwise, we estimate...

- A number of approaches exist to estimate this gradient
- They make varying levels of assumptions about the distribution and function being optimized
- Most popular methods either make strong assumptions or suffer from high variance
REINFORCE (Williams, 1992)

\[ \hat{g}_{\text{REINFORCE}}[f] = f(b) \frac{\partial}{\partial \theta} \log p(b|\theta), \quad b \sim p(b|\theta) \]

- Unbiased
- Has few requirements
- Easy to compute
- Suffers from high variance
Reparameterization (Kingma & Welling, 2014)

\[ \hat{g}_{\text{reparam}}[f] = \frac{\partial f}{\partial b} \frac{\partial b}{\partial \theta} \quad b = T(\theta, \epsilon), \epsilon \sim p(\epsilon) \]

- Lower variance empirically
- Unbiased
- Makes stronger assumptions
- Requires $f(b)$ is known and differentiable
- Requires $p(b|\theta)$ is reparameterizable
Concrete
(Maddison et al., 2016)

\[ \hat{g}_{\text{concrete}}[f] = \frac{\partial f}{\partial \sigma(z/t)} \frac{\partial \sigma(z/t)}{\partial \theta} \]

\[ z = T(\theta, \epsilon), \epsilon \sim p(\epsilon) \]

- Works well in practice
- Low variance from reparameterization
- Biased
- Adds temperature hyper-parameter
- Requires that \( f(b) \) is known, and differentiable
- Requires \( p(z|\theta) \) is reparameterizable
- Requires \( f(b) \) behaves predictably outside of domain
Control Variates

- Allow us to reduce variance of a Monte Carlo estimator

\[ \hat{g}_{\text{new}}(b) = \hat{g}(b) - c(b) + \mathbb{E}_{p(b)}[c(b)] \]

- Variance is reduced if \( \text{corr}(g, c) > 0 \)

- Does not change bias
Putting it all together

- We would like a general gradient estimator that is
  - unbiased
  - low variance
  - usable when \( f(b) \) is unknown
  - usable when \( p(b|\theta) \) is discrete
Backpropagation Through
Backpropagation Through The Void
Our Approach

\[ \hat{g}_{\text{LAX}} = g_{\text{REINFORCE}}[f] - g_{\text{REINFORCE}}[c_{\phi}] + g_{\text{reparam}}[c_{\phi}] \]
Our Approach

\[ \hat{g}_{\text{LAX}} = g_{\text{REINFORCE}}[f] - g_{\text{REINFORCE}}[c_{\phi}] + g_{\text{reparam}}[c_{\phi}] \]

\[ = \left[ f(b) - c_{\phi}(b) \right] \frac{\partial}{\partial \theta} \log p(b|\theta) + \frac{\partial}{\partial \theta} c_{\phi}(b) \]

- Start with the reinforce estimator for \( f(b) \)
- We introduce a new function \( c_{\phi}(b) \)
- We subtract the reinforce estimator of its gradient and add the reparameterization estimator
- Can be thought of as using the reinforce estimator of \( c_{\phi}(b) \) as a control variate
Optimizing the Control Variate

\[ \frac{\partial}{\partial \phi} \text{Variance}(\hat{g}) = \mathbb{E} \left[ \frac{\partial}{\partial \phi} \hat{g}^2 \right] \]

- For any unbiased estimator we can get Monte Carlo estimates for the gradient of the variance of \( \hat{g} \)
- Use to optimize \( c_\phi \)
What about discrete b?
Extension to discrete $p(b|\theta)$

\[
\hat{g}_{\text{RELAX}} = [f(b) - c_\phi(\tilde{z})] \frac{\partial}{\partial \theta} \log p(b|\theta) + \frac{\partial}{\partial \theta} c_\phi(z) - \frac{\partial}{\partial \theta} c_\phi(\tilde{z})
\]

\[
b = H(z), z \sim p(z|\theta), \tilde{z} \sim p(z|b, \theta)
\]

- When $b$ is discrete, we introduce a relaxed distribution $p(z|\theta)$ and a function $H$ where $H(z) = b \sim p(b|\theta)$

- We use the conditioning scheme introduced in REBAR (Tucker et al. 2017)

- Unbiased for all $c_\phi$
A Simple Example

\[ \mathbb{E}_{p(b|\theta)}[(t - b)^2] \]

- Used to validate REBAR (used \( t = .45 \))
- We use \( t = .499 \)
- REBAR, REINFORCE fail due to noise outweighing signal
- Can RELAX improve?
• RELAX outperforms baselines
• Considerably reduced variance!
• RELAX learns reasonable surrogate
Analyzing the Surrogate

- REBAR's fixed surrogate cannot produce consistent and correct gradients
- RELAX learns to balance REINFORCE variance and reparameterization variance
A More Interesting Application

\[
\log p(x) \geq \mathcal{L}(\theta) = \mathbb{E}_{q(b|x)}[\log p(x|b) + \log p(b) - \log q(b|x)]
\]

- Discrete VAE
- Latent state is 200 Bernoulli variables
- Discrete sampling makes reparameterization estimator unusable

\[
c_{\phi}(z) = f(\sigma_{\lambda}(z)) + r_{\rho}(z)
\]
Results
Reinforcement Learning

• Policy gradient methods are very popular today (A2C, A3C, ACKTR)

• Seeks to find $\arg\max_\theta E_{\tau \sim \pi(\tau | \theta)}[R(\tau)]$

• Does this by estimating $\frac{\partial}{\partial \theta} E_{\tau \sim \pi(\tau | \theta)}[R(\tau)]$

• $R$ is not known so many popular estimators cannot be used
Actor Critic

\[ \hat{g}_{AC} = \sum_{t=1}^{T} \frac{\partial \log \pi(a_t|s_t, \theta)}{\partial \theta} \left[ \sum_{t'=t}^{T} r_{t'} - c_\phi(s_t) \right] \]

- \( c_\phi \) is an estimate of the value function

- This is exactly the REINFORCE estimator using an estimate of the value function as a control variate

- Why not use action in control variate?

- Dependence on action would add bias
LAX for RL

\[ \hat{J}_{\text{LAX}} = \sum_{t=1}^{T} \frac{\partial \log \pi(a_t | s_t, \theta)}{\partial \theta} \left[ \sum_{t' = t}^{T} r_{t'} - c_\phi(s_t, a_t) \right] + \frac{\partial}{\partial \theta} c_\phi(s_t, a_t) \]

- Allows for action dependence in control variate
- Remains unbiased
- Similar extension available for discrete action spaces
Results

- Improved performance
- Lower variance gradient estimates
Future Work

- What does the optimal surrogate look like?
- Many possible variations of LAX and RELAX
- Which provides the best tradeoff between variance, ease of implementation, scope of application, performance
- RL
  - Incorporate other variance reduction techniques (GAE, reward bootstrapping, trust-region)
  - Ways to train the surrogate off-policy
- Applications
  - Inference of graph structure (coming soon)
  - Inference of discrete neural network architecture components (coming soon)
Directions

- Surrogate can take any form
  - can rely on global information even if forward pass only uses local info
- Can depend on order even if forward pass is invariant
- Reparameterization can take many forms, ongoing work on reparameterizing through rejection sampling, or distributions on permutations
Reparameterizing the Birkhoff Polytope for Variational Permutation Inference
Learning Latent Permutations with Gumbel-Sinkhorn Networks

\[ P_{\theta, \tilde{X}} = S(g(\tilde{X}, \theta)/\tau) \]

\[ P_{\theta, \tilde{X}}^{-1}(\tilde{X}) \approx X \]
Why are we optimizing policies anyways?

- Next week: Variational optimization