My First Thesis

Smart Things as Said by Me

Daan van Stigt
Institute for Logic, Language and Computation (ILLC)

Contents

My First Thesis

ABSTRACT

This document describes how to prepare a Foundations and Trends® article in LaTeX . The accompanying LaTeX source file FnTarticle.tex (that produces this output) is an example of such a file.

1

Inside and outside recursion for minimal span-based parser

1.1 Semiring formulation

So yeah, the highlights are: an edge connects three nodes, a parent and two CHILDREN, each node is a labelled SPAN; you need to identify the scoring function for an edge, let's call it $w(e)$, in this case we have

$$w(e) = f(\textsc{head}(e)) \bigotimes_{c \in \textsc{children}(e)} g(\textsc{span}(c)) \tag{1.1}$$

where f and g are parametric functions; then you can compute the Inside recursion for a node v

$$I(v) = \bigoplus_{e \in BS(v)} w(e) \otimes \bigotimes_{c \in \textsc{children}(e)} I(c) \tag{1.2}$$

where I'm using $BS(v)$ to denote the set of edges incoming to v; note that BS here basically enumerates the different ways to segment the string under (i, j) into two adjacent parts and the different labels of each child SPAN (let's call these a and b, each an element in the labelset

L), thus we can write

$$I(v = [i,j,l]) = \bigoplus_{\substack{e=[i,j,l,k,a,b]: \\ a \in L, \\ b \in L, \\ k \in \{i+1,\ldots,j-1\}}} w(e) \otimes I([i,k,a]) \otimes I([k+1,j,b]) \quad (1.3)$$

Now the key is to realise that $w(e)$ factorises and therefore we can rewrite this as

$$I(v = [i,j,l]) = f(i,j,l) \otimes \bigoplus_{k=i+1}^{j-1} g(i,k) \otimes g(k+1,j) \quad (1.4)$$

$$\otimes \bigoplus_{a \in L} I([i,k,a]) \quad (1.5)$$

$$\otimes \bigoplus_{b \in L} I([k+1,j,b]) \quad (1.6)$$

and this finally motivates having an inside table for the SPANS (with labels summed out), let's call that

$$S(i,j) = \bigoplus_{l \in L} I(i,j,l) \quad (1.7)$$

and then we have the result

$$I(i,j,l) = f(i,j,l) \otimes \bigoplus_{k=i+1}^{j-1} g(i,k) \otimes g(k+1,j) \otimes S(i,k) \otimes S(k+1,j). \quad (1.8)$$

1.2 Alternative formulation

In this derivation we follow Michael Collins notes on the Inside-Outside Algorithm.[1]

Let a sentence be $x_1, \ldots, x_n$, where each x_i is a word. We are given a CFG (N, Σ, R, S) in Chomsky normal form. Let ψ be a function that maps any rule production $r \in R$ of the form $\langle A \to B\ C, i, k, j \rangle$ or $\langle A, i, i+1 \rangle$ to a value $\psi(r) \geq 0$. Let a tree T be a set of such rules r with the only constraint that these rules make up a tree.

[1] http://www.cs.columbia.edu/~mcollins/io.pdf

Following the minimal span parser we define ψ as

$$\log \psi(A \to B\ C, i, k, j) \triangleq s_{label}(i, j, A) + s_{span}(i, j) \tag{1.9}$$

$$\text{and} \tag{1.10}$$

$$\log \psi(A, i, i+1) \triangleq s_{label}(i, i+1, A) + s_{span}(i, i+1), \tag{1.11}$$

and thus the potential of a tree as

$$\log \Psi(T) = \sum_{r \in T} \log \psi(r) \tag{1.12}$$

$$= \sum_{\langle A,i,j \rangle \in T} s_{label}(i, j, A) + s_{span}(i, j), \tag{1.13}$$

$$\tag{1.14}$$

Note that the potential function as defined in 1.9 disregards most of the information in a binary rule. In particular we see that B, C and k, the labels and split-point of the children, are discarded.

Now note that equation corresponds exactly to the second formula in section 3 of the minimal span-based parser paper

$$s_{tree}(T) = \sum_{(\ell(i,j)) \in T} [s_{label}(i, j, \ell) + s_{span}(i, j)]. \tag{1.15}$$

which is how I derived that 1.9 is the correct formula for the rule score.

We obtain our CRF objective when we normalize this score globally

$$P(T) = \frac{\prod_{r \in T} \psi(r)}{\sum_{T' \in \mathcal{T}} \prod_{r' \in T'} \psi(r')} \tag{1.16}$$

$$\tag{1.17}$$

or equivalently

$$\log P(T) = \sum_{r \in T} \log \psi(r) - \log \sum_{T \in \mathcal{T}} \prod_{r \in T} \psi(r) \tag{1.18}$$

$$\tag{1.19}$$

From the aforementioned notes we get the following general result for the inside value α. For all $A \in N$, for all $0 \leq i < n$

$$\alpha(A, i, i+1) = \psi(A, i, i+1) \tag{1.20}$$

and for all (i, j) such that $1 \leq i < j \leq n$:

$$\alpha(A, i, j) = \sum_{A \to BC} \sum_{k=i+1}^{j-1} \psi(A \to B\ C, i, k, j) \cdot \alpha(B, i, k) \cdot \alpha(C, k, j) \tag{1.21}$$

Note that we are considering a CFG in which the rule set is complete, i.e.

$$\langle A \to B\ C \rangle \in R \text{ for each } (A, B, C) \in N^3, \tag{1.22}$$

and recall that the labels B and C do not appear in the scoring functions in 1.9. These facts will allow us to simplify the expression in formula 1.21 as

$$\alpha(A, i, j) = \sum_{B \in N} \sum_{C \in N} \sum_{k=i+1}^{j-1} \tilde{s}_{label}(i, j, A) \cdot \tilde{s}_{span}(i, j) \alpha(B, i, k) \cdot \alpha(C, k, j) \tag{1.23a}$$

$$= \tilde{s}_{label}(i, j, A) \cdot \tilde{s}_{span}(i, j) \sum_{k=i+1}^{j-1} \sum_{B \in N} \alpha(B, i, k) \cdot \sum_{C \in N} \alpha(C, k, j) \tag{1.23b}$$

$$= \tilde{s}_{label}(i, j, A) \cdot \tilde{s}_{span}(i, j) \sum_{k=i+1}^{j-1} S(i, k) \cdot S(k, j) \tag{1.23c}$$

where we've introduced a number of notational abbreviations

$$\tilde{s}_{label}(i, j, A) = \exp(s_{label}(i, j, A)) \tag{1.24}$$

$$\tilde{s}_{span}(i, j) = \exp(s_{span}(i, j)) \tag{1.25}$$

$$S(i, j) = \sum_{A \in N} \alpha(A, i, j) \tag{1.26}$$

Note that this is the exact same formula as 1.8.

From equation 1.23c we can deduce that we in fact do even need to store the values $\alpha(i, j, A)$ but that it suffices to only store the marginalized values $S(i, j)$. In this case, the recursion simplifies even further:

$$\begin{aligned}
S(i,j) &= \sum_{A \in N} \alpha(A,i,j) && (1.27a)\\
&= \sum_{A \in N} \tilde{s}_{label}(i,j,A) \cdot \tilde{s}_{span}(i,j) \sum_{k=i+1}^{j-1} S(i,k) \cdot S(k,j) && (1.27b)\\
&= \left[\sum_{A \in N} \tilde{s}_{label}(i,j,A) \cdot \tilde{s}_{span}(i,j) \right] \left[\sum_{k=i+1}^{j-1} S(i,k) \cdot S(k,j) \right] && (1.27c)
\end{aligned}$$

where we put explicit brackets to emphasize that independence of the subproblems of labeling and splitting. We can now recognize this as the 'inside' equivalent of the expression from the paper[2]

$$s_{best}(i,j) = \max_{\ell}[s_{label}(i,j,\ell)] + \max_{k}[s_{split}(i,k,j)]. \tag{1.28}$$

The recursions are the same; the semirings are different. The viterbi recursion given above is in the ViterbiSemiring, which uses the max operator as $\oplus$; the inside recursion given in 1.23c has standard addition $(+)$ instead.

[2]I believe there is actually an error in this equation: it should read $s_{label}(i,j,\ell) + s_{span}(i,j)$ instead of just $s_{label}(i,j,\ell)$. This is implied by the score for a single node, which is given by equation 1.9, taken directly from the paper.

1.3 Outside

$$
\begin{aligned}
\beta(A,i,j) &= \sum_{B\to CA\in R}\sum_{k=1}^{i-1}\psi(B\to CA,k,i-1,j)\cdot\alpha(C,k,i-1)\cdot\beta(B,k,j)\\
&\quad+\sum_{B\to AC\in R}\sum_{k=j+1}^{n}\psi(B\to A,C,i,j,k)\cdot\alpha(C,j+1,k)\cdot\beta(B,i,k)\\
&=\sum_{B\in N}\sum_{C\in N}\sum_{k=1}^{i-1}\psi(B,k,j)\cdot\alpha(C,k,i-1)\cdot\beta(B,k,j)\\
&\quad+\sum_{B\in N}\sum_{C\in N}\sum_{k=j+1}^{n}\psi(B,i,k)\cdot\alpha(C,j+1,k)\cdot\beta(B,i,k)\\
&=\sum_{k=1}^{i-1}\left[\sum_{B\in N}\psi(B,k,j)\cdot\beta(B,k,j)\right]\cdot\left[\sum_{C\in N}\alpha(C,k,i-1)\right]\\
&\quad+\sum_{k=j+1}^{n}\left[\sum_{B\in N}\psi(B,i,k)\cdot\beta(B,i,k)\right]\cdot\left[\sum_{C\in N}\alpha(C,j+1,k)\right]\\
&=\sum_{k=1}^{i-1}S'(k,j)\cdot S(k,i-1)+\sum_{k=j+1}^{n}S'(i,k)\cdot S(j+1,k)
\end{aligned}
$$

where

$$
\begin{aligned}
S(i,j) &= \sum_{A\in N}\alpha(A,i,j)\\
S'(i,j) &= \sum_{A\in N}\psi(A,i,j)\beta(A,i,j)
\end{aligned}
$$

2

Semisupervised learning

Throughout these note we write x for a sentence, y for a (latent) constituency tree, and $\mathcal{Y}(x)$ for the *yield* of x, that is, all trees that can be assigned to x. Furthermore, let $\mathbb{L}$ be a set of pairs (x, y) of sentences with gold trees, and let $\mathbb{U}$ be a set of unlabeled sentences x.

We define the following semi-supervised objective

$$\mathcal{J} \triangleq \mathcal{J}_{\mathcal{S}} + \alpha \mathcal{J}_{\mathcal{U}}, \tag{2.1}$$

were $\mathcal{J}_{\mathcal{S}}$ is the supervised objective optimized over $\mathbb{L}$ and $\mathcal{J}_{\mathcal{U}}$ the unsupervised objective optimized over $\mathbb{U}$. We introduce $\alpha \in \mathbb{R}$ as an arbitrary scalar controlling the contribution of the unsupervised objective.

2.1 Supervised objective

Let $p_\theta(x, y)$ be parametrized by a Generative RNNG (Dyer *et al.*, 2016). Then our supervised objective is

$$\mathcal{J}_{\mathcal{S}} \triangleq \sum_{(x,y)\in\mathbb{L}} \log p_\theta(x, y) \tag{2.2}$$

$$\tag{2.3}$$

This objective is optimized as usual using stochastic gradient estimates:

$$\nabla_\theta \mathcal{J}_\mathcal{S} \approx \frac{|\mathbb{L}|}{n} \sum_{i=1}^{n} \nabla_\theta \log p_\theta(x^{(i)}, y^{(i)}), \tag{2.4}$$

$$\tag{2.5}$$

where $\{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$ is a mini-batch sampled uniformly from $\mathbb{L}$. To compute $\nabla_\theta \log p_\theta(x^{(i)}, y^{(i)})$ we rely on automatic differentiation (**Baydin+2015:AD**).

2.2 Unsupervised objective

Consider the following objective to be maximized:

$$\mathcal{J}_\mathcal{U} \triangleq \sum_{x \in \mathbb{U}} \log p(x) \tag{2.6a}$$

$$= \sum_{x \in \mathbb{U}} \log \sum_{y \in \mathcal{Y}(x)} p_\theta(x, y) \tag{2.6b}$$

This is a language modelling objective, in which we treat y as latent, and $p_\theta(x, y)$ is a generative RNNG. A consequence the independence assumptions of the RNNG (or better: lack thereof) is that the sum over trees y is no longer tractable. To optimize this objective we must fall back on approximate methods.

Variational approximation We optimize the objective using variational inference (Blei *et al.*, 2016). We introduce a posterior $q_\lambda(y|x)$ parametrised by λ and use Jensen's inequality to derive a variational lower bound:

$$\log p(x) = \log \sum_{y \in \mathcal{Y}(x)} q_\lambda(y|x) \frac{p_\theta(x, y)}{q_\lambda(y|x)} \tag{2.7a}$$

$$= \log \mathbb{E}_{q_\lambda}\left[\frac{p_\theta(x, y)}{q_\lambda(y|x)}\right] \tag{2.7b}$$

$$\geq \mathbb{E}_{q_\lambda}\left[\log \frac{p_\theta(x, y)}{q_\lambda(y|x)}\right] \tag{2.7c}$$

$$= \mathbb{E}_{q_\lambda}\left[\log p_\theta(x, y) - \log q_\lambda(y|x)\right] \triangleq \mathcal{L}(\theta, \lambda) \tag{2.7d}$$

The only requirement for q_λ is that

$$p(x, y) > 0 \Rightarrow q(y|x) > 0 \qquad \text{for all } x \text{ and } y \in \mathcal{Y}(x). \tag{2.8}$$

Any discriminatively trained parser fulfills this requirement.

The lower bound $\mathcal{L}(\theta, \lambda)$ will be optimized by gradient optimization, which means we will need to take the gradients $\nabla_\theta \mathcal{L}(\theta, \lambda)$ and $\nabla_\lambda \mathcal{L}(\theta, \lambda)$.

Gradients of joint parameters The first gradient is easy and permits a straightforward Monte-Carlo estimate:

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta, \lambda) &= \nabla_\theta \mathbb{E}_{q_\lambda}\left[\log p_\theta(x, y) - \log q_\lambda(y|x)\right] && (2.9a)\\
&= \mathbb{E}_{q_\lambda}\left[\nabla_\theta \log p_\theta(x, y) - \nabla_\theta \log q_\lambda(y|x)\right] && (2.9b)\\
&= \mathbb{E}_{q_\lambda}\left[\nabla_\theta \log p_\theta(x, y)\right] && (2.9c)\\
&\approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \log p_\theta(x, y^{(i)}) && (2.9d)
\end{aligned}$$

where $y_i \sim q_\lambda(y|x)$ for $i = 1, \ldots, n$. We can move the gradient inside the expectation (second equality) because q does not depend on θ.

Gradients of posterior parameters The second gradient is harder and requires us to rewrite the objective:

$$\begin{aligned}
\nabla_\lambda \mathcal{L}(\theta, \lambda) &= \nabla_\lambda \mathbb{E}_{q_\lambda} \left[\log p_\theta(x, y) - \log q_\lambda(y|x) \right] && \text{(2.10a)} \\
&= \nabla_\lambda \sum_y \{ q_\lambda(y|x) \log p_\theta(x, y) - q_\lambda(y|x) \log q_\lambda(y|x) \} && \text{(2.10b)} \\
&= \sum_y \{ \nabla_\lambda q_\lambda(y|x) \log p_\theta(x, y) && \text{(2.10c)} \\
&\qquad - \nabla_\lambda q_\lambda(y|x) \log q_\lambda(y|x) && \text{(2.10d)} \\
&\qquad - q_\lambda(y|x) \nabla_\lambda \log q_\lambda(y|x) \} && \text{(2.10e)} \\
&= \sum_y \{ \nabla_\lambda q_\lambda(y|x) \log p_\theta(x, y) - \nabla_\lambda q_\lambda(y|x) \log q_\lambda(y|x) \} && \text{(2.10f)} \\
&= \sum_y \{ (\log p_\theta(x, y) - \log q_\lambda(y|x)) \nabla_\lambda q_\lambda(y|x) \} && \text{(2.10g)} \\
&= \sum_y \{ (\log p_\theta(x, y) - \log q_\lambda(y|x)) q_\lambda(y|x) \nabla_\lambda \log q_\lambda(y|x) \} && \text{(2.10h)} \\
&= \mathbb{E}_{q_\lambda} [(\log p_\theta(x, y) - \log q_\lambda(y|x)) \nabla_\lambda \log q_\lambda(y|x) \} && \text{(2.10i)} \\
&= \mathbb{E}_{q_\lambda} [l(x, y) \nabla_\lambda \log q_\lambda(y|x)] && \text{(2.10j)}
\end{aligned}$$

Where we've defined a learning signal $l(x, y) \triangleq \log p_\theta(x, y) - \log q_\lambda(y|x)$.

In this derivation we used the identity

$$\nabla_\lambda \log q_\lambda(y|x) = \frac{\nabla_\lambda q_\lambda(y|x)}{q_\lambda(y|x)} \tag{2.11}$$

or equivalently

$$\nabla_\lambda q_\lambda(y|x) = q_\lambda(y|x) \nabla_\lambda \log q_\lambda(y|x) \tag{2.12}$$

and the fact that

$$\begin{aligned}
\sum_y q_\lambda(y|x)\nabla_\lambda \log q_\lambda(y|x) &= \sum_y q_\lambda(y|x)\frac{\nabla_\lambda q_\lambda(y|x)}{q_\lambda(y|x)} && (2.13a)\\
&= \sum_y \nabla_\lambda q_\lambda(y|x) && (2.13b)\\
&= \nabla_\lambda \sum_y q_\lambda(y|x) && (2.13c)\\
&= \nabla_\lambda 1 && (2.13d)\\
&= 0. && (2.13e)\\
& && (2.13f)
\end{aligned}$$

This rewritten objective permits a straightforward MC estimate:

$$\mathbb{E}_{q_\lambda}\left[l(x,y)\nabla_\lambda \log q_\lambda(y|x)\right] \approx \frac{1}{n}\sum_{i=1}^{n} l(x,y^{(i)})\nabla_\lambda \log q_\lambda(x|y^{(i)}) \qquad (2.14)$$

where $y_i \sim q_\lambda(y|x)$ for $i = 1,\dots,n$.

This estimator has been derived (in slightly different forms) in among others (Williams, 1992), (Paisley *et al.*, 2012), (Mnih and Gregor, 2014), (Ranganath *et al.*, 2014), and (Miao and Blunsom, 2016) and is known as the REINFORCE estimator (Williams, 1992), or score function estimator (after the score function $\nabla_\theta \log p_\theta(x)$) (Fu, 2006).

2.3 Optimization

We use automatic differentiation (**Baydin+2015:AD**) to obtain all our gradient estimates.

To get the gradients in formula 2.14 we rewrite it in the form of a surrogate objective (Schulman *et al.*, 2015)

$$L(\lambda) = \frac{1}{n}\sum_{i=1}^{n} \log q_\lambda(x|y^{(i)})\textsc{blockgrad}(l(x,y^{(i)})) \qquad (2.15)$$

where BLOCKGRAD is function that 'detaches' a parametrized function from the computation graph effectively turning it into a scalar. That is, loosely speaking

$$\textsc{blockgrad}(f_\theta(x)) = f(x) \qquad (2.16)$$

such that

$$\nabla_\theta \text{BLOCKGRAD}(f_\theta(x)) = \nabla_\theta f(x) \tag{2.17}$$
$$= 0 \tag{2.18}$$

Then differentiation of L gives us the unbiased estimator

$$\nabla_\lambda L(\lambda) = \frac{1}{n}\sum_{i=1}^{n} l(x, y^{(i)})\nabla_\lambda \log q_\lambda(x|y^{(i)}) \tag{2.19}$$

2.4 Variance reduction

We have derived an estimator for the gradient of the posterior parameters in the unsupervised objective. This estimator is unbiased, but is known to have high variance, often too much to be useful (Paisley *et al.*, 2012). Two effective methods to counter this are control variates and baselines (Ross, 2006).

Variance of estimator First, let's analyze the variance of our estimator. Note that our expectation is of the general form

$$\mu \triangleq \mathbb{E}[f(X)] \tag{2.20}$$

and that we estimate this quantity by generating n independent samples $X_1, \ldots, X_n \sim P(X)$ and computing

$$\hat{\mu} \triangleq \frac{1}{n}\sum_{i=1}^{n} f(X_i). \tag{2.21}$$

This is an unbiased estimator for μ with error

$$\text{MSE} = \mathbb{E}[(\mu - \hat{\mu})^2] = \text{Var}[\hat{\mu}] = \frac{\text{Var}[\hat{\mu}]}{n}, \tag{2.22}$$

which means that the error is of the order

$$\mu - \hat{\mu} \sim \sqrt{\frac{\text{Var}[\hat{\mu}]}{n}} \tag{2.23}$$

and reducing it linearly requires a quadratic number of samples.

In our particular case, the function f is

$$f_{X=x}(Y) \triangleq l(X, Y)\nabla_\lambda \log q_\lambda(Y|X = x) \tag{2.24}$$

where we have made explicit that y is the random variable, and x is given.

Control variates Consider a function $\phi(X)$ with known expectation

$$\mu_\phi \triangleq \mathbb{E}[\phi(X)] \tag{2.25}$$

Then we can define a new function $\hat{f}$ such that

$$\hat{f}(X) \triangleq f(X) - \phi(X) + \mu_\phi. \tag{2.26}$$

This function is also an estimator for μ, since

$$\begin{aligned}\mathbb{E}[\hat{f}(X)] &= \mathbb{E}[f(X)] - \mu_\phi + \mu_\phi &(2.27)\\ &= \mathbb{E}[f(X)], &(2.28)\end{aligned}$$

and a computation shows that the variance of the new function is

$$\begin{aligned}\mathrm{Var}[\hat{f}(X)] &= \mathbb{E}[(f(X) - \phi(X) + \mu_\phi) - \mu)^2]\\ &= \mathbb{E}[(f(X) - \phi(X) + \mu_\phi)^2] - 2\mathbb{E}[(f(X) - \phi(X) + \mu_\phi)\mu] + \mathbb{E}[\mu^2]\\ &= \mathbb{E}[(f(X) - \phi(X) + \mu_\phi)^2] - 2\mathbb{E}[(f(X) - \phi(X) + \mu_\phi)]\mu + \mu^2\\ &= \mathbb{E}[(f(X) - \phi(X) + \mu_\phi)^2] - 2\mu^2 + \mu^2\\ &= \mathbb{E}[f(X)^2 + \phi(X)^2 + \mu_\phi^2 - 2f(X)\phi(X) + 2f(X)\mu_\phi - 2\phi(X)\mu_\phi] - \mu^2\\ &= \mathbb{E}[f(X)^2] - \mathbb{E}[f(X)]^2\\ &\quad - 2(\mathbb{E}[f(X)\phi(X)] - \mathbb{E}[f(X)]\mathbb{E}[\phi(X)])\\ &\quad + \mathbb{E}[\phi(X)^2] - \mathbb{E}[\phi(X)]^2\\ &= \mathrm{Var}[f(X)] - 2\ \mathrm{Cov}[f(X), \phi(X)] + \mathrm{Var}[\phi(X)]\end{aligned}$$

This means we can get a reduction in variance whenever

$$\mathrm{Cov}[f(X), \phi(X)] > \frac{1}{2}\ \mathrm{Var}[\phi(X)]. \tag{2.29}$$

The function ϕ is called a *control variate*—it allows us to control the variance of f.

From the equality above we can see that this will be the case whenever $f(X)$ and $\phi(X)$ are strongly correlated. Our choice of control variate will be made with the that in mind. Furthermore, $\mathbb{E}[\phi(X)]$ must be known. What is an optimal control variate? Typically a control variate of the form $a\phi$ is chosen with fixed, and a is optimized to maximize the correlation. This brings us to the generic formulation of a control variate:

$$\hat{f}(X) \triangleq f(X) - a(\phi(X) - \mathbb{E}[\phi(X)]) \tag{2.30}$$

with variance

$$\mathrm{Var}[\hat{f}(X)] = \mathrm{Var}[f(X)] - 2a\ \mathrm{Cov}[f(X), \phi(X)] + a^2\ \mathrm{Var}[\phi(X)] \tag{2.31}$$
$$\tag{2.32}$$

We take a derivative of this with respect to a

$$\frac{\partial}{\partial a}\mathrm{Var}[\hat{f}(X)] = -2\ \mathrm{Cov}[f(X), \phi(X)] + 2a\ \mathrm{Var}[\phi(X)] \tag{2.33}$$

Setting this to zero and solving for a we obtain the optimal choice for a

$$a = \frac{\mathrm{Cov}[f(X), \phi(X)]}{\mathrm{Var}[\phi(X)]}. \tag{2.34}$$

Plugging in this solution into the expression for $\mathrm{Var}[\hat{f}(X)]$ and dividing by $\mathrm{Var}[f(X)]$ we get

$$\frac{\mathrm{Var}[\hat{f}(X)]}{\mathrm{Var}[f(X)]} = 1 - \frac{\mathrm{Cov}[f(X), \phi(X)]}{\mathrm{Var}[f(X)]\ \mathrm{Var}[\phi(X)]} \tag{2.35}$$
$$= 1 - \mathrm{corr}^2[f(X), \phi(X)], \tag{2.36}$$

which shows that given this choice of a the reduction in variance is directly determined by the correlation between $f(X)$ and $\phi(X)$.

Bringing this all together, we let our new estimator be

$$\mathbb{E}[f(X)] = \mathbb{E}[\hat{f}(X)] \approx \frac{1}{n}\sum_{i=1}^{n}[f(X_i) - a\phi(X_i)] - \mu_\phi \tag{2.37}$$

Example (Ross, 2006) Suppose we want to use simulation to determine

$$\mathbb{E}[f(X)] = \mathbb{E}[e^X] = \int_0^1 e^x dx = e - 1 \tag{2.38}$$

with $X \sim \mathcal{U}(0, 1)$. A natural control variate to use in this case is the random variable X itself: $\phi(X) \triangleq X$. We thus define the new estimator

$$\hat{f}(X) = f(X) - \phi(X) + \mathbb{E}[\phi(X)] \tag{2.39}$$
$$= e^X - X + \frac{1}{2}. \tag{2.40}$$

To compute the decrease in variance with this new estimator, we first note that

$$\begin{aligned}
\mathrm{Cov}(e^X, X) &= \mathbb{E}[Xe^X] - \mathbb{E}[X]\mathbb{E}[e^X] \\
&= \int_0^1 xe^x dx - \frac{e-1}{2} \\
&= 1 - \frac{e-1}{2} \approx 0.14086 \\
\mathrm{Var}[e^X] &= \mathbb{E}[e^{2X}] - (\mathbb{E}[e^X])^2 \\
&= \int_0^1 e^{2x} dx - (1-e^x)^2 \\
&= \frac{e^2-1}{2} - (1-e^x)^2 \approx 0.2420 \\
\mathrm{Var}[X] &= \mathbb{E}[X^2] - (\mathbb{E}[X])^2 \\
&= \int_0^1 x^2 dx - \frac{1}{4} \\
&= \frac{1}{3} - \frac{1}{4} = \frac{1}{12}.
\end{aligned}$$

When we choose a as in formula 2.34 we can use formula 2.35 to compute that

$$\frac{\mathrm{Var}[\hat{f}(X)]}{\mathrm{Var}[f(X)]} = 1 - \frac{(0.14086)^2}{\frac{0.2420}{12}} \tag{2.41}$$

$$\approx 0.0161. \tag{2.42}$$

This is a reduction of 98.4 percent! A simulation illustrates what this looks like in practice with ... samples:

Acknowledgements

The author is grateful to the author, for he is his own eternal flame of inspiration.

Appendices

A

Figures

References

Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (2016). "Variational Inference: A Review for Statisticians". *ArXiv e-prints*. Jan.

Dyer, C., A. Kuncoro, M. Ballesteros, and N. A. Smith (2016). "Recurrent Neural Network Grammars". *CoRR*. abs/1602.07776. arXiv: 1602.07776. URL: http://arxiv.org/abs/1602.07776.

Fu, M. C. (2006). "Gradient Estimation". In: *Handbooks in Operations Research and Management Science (Volume 13)*. Ed. by B. L. N. Edited by Shane G. Henderson. Elsevier. Chap. 19. 575–616.

Miao, Y. and P. Blunsom (2016). "Language as a Latent Variable: Discrete Generative Models for Sentence Compression". *CoRR*. abs/1609.07317.

Mnih, A. and K. Gregor (2014). "Neural Variational Inference and Learning in Belief Networks". *CoRR*. abs/1402.0030. arXiv: 1402.0030. URL: http://arxiv.org/abs/1402.0030.

Paisley, J., D. Blei, and M. Jordan (2012). "Variational Bayesian Inference with Stochastic Search". In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. Ed. by J. Langford and J. Pineau. *ICML '12*. New York, NY, USA: Omnipress. 1367–1374.

Ranganath, R., S. Gerrish, and D. Blei (2014). “Black Box Variational Inference”. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*. Ed. by S. Kaski and J. Corander. Vol. 33. *Proceedings of Machine Learning Research*. Reykjavik, Iceland: PMLR. 814–822. URL: http://proceedings.mlr.press/v33/ranganath14.html.

Ross, S. M. (2006). *Simulation, Fourth Edition*. Orlando, FL, USA: Academic Press, Inc.

Schulman, J., N. Heess, T. Weber, and P. Abbeel (2015). “Gradient Estimation Using Stochastic Computation Graphs”. In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. Curran Associates, Inc. 3528–3536. URL: http://papers.nips.cc/paper/5899-gradient-estimation-using-stochastic-computation-graphs.pdf.

Williams, R. J. (1992). “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning”. *Mach. Learn.* 8(3-4): 229–256. ISSN: 0885-6125. DOI: 10.1007/BF00992696. URL: https://doi.org/10.1007/BF00992696.