A Baseline for Any Order Gradient Estimation in Stochastic Computation Graphs

Jingkai Mao [*1] **Jakob Foerster** [*2] **Tim Rocktäschel** [3] **Maruan Al-Shedivat** [4] **Gregory Farquhar** [2] **Shimon Whiteson** [2]

Abstract

By enabling correct differentiation in *stochastic computation graphs* (SCGs), the *infinitely differentiable Monte-Carlo estimator* (DiCE) can generate correct estimates for the higher order gradients that arise in, e.g., multi-agent reinforcement learning and meta-learning. However, the baseline term in DiCE that serves as a control variate for reducing variance applies only to first order gradient estimation, limiting the utility of higher-order gradient estimates. To improve the sample efficiency of DiCE, we propose a new baseline term for higher order gradient estimation. This term may be easily included in the objective, and produces unbiased variance-reduced estimators under (automatic) differentiation, without affecting the estimate of the objective itself or of the first order gradient estimate. It reuses the same baseline function (e.g., the state-value function in reinforcement learning) already used for the first order baseline. We provide theoretical analysis and numerical evaluations of this new baseline, which demonstrate that it can dramatically reduce the variance of DiCE's second order gradient estimators and also show empirically that it reduces the variance of third and fourth order gradients. This computational tool can be easily used to estimate higher order gradients with unprecedented efficiency and simplicity wherever automatic differentiation is utilised, and it has the potential to unlock applications of higher order gradients in reinforcement learning and meta-learning.

[*]Equal contribution [1]Man AHL, London, UK. Work done at the University of Oxford. [2]Department of Computer Science, University of Oxford, Oxford, UK [3]Department of Computer Science, University College London, London, UK [4]School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA. Correspondence to: Jakob Foerster <jnf@fb.com>, Jingkai Mao <jingkai.mao@man.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

1. Introduction

Machine learning problems with intractable stochasticity often yield objectives that are not directly differentiable. In reinforcement learning, for example, the expected return involves an expectation over the stochasticity induced by both the policy and the environment dynamics. As a result, the gradient of this objective with respect to the policy parameters cannot be directly calculated.

Instead, we must construct Monte Carlo estimates of the gradient from samples, and then apply gradient-based optimisation methods. In such problems, the score function trick (Fu, 2006) may be used to easily define estimates of the first order gradients of the stochastic objective. However, to calculate these gradient estimates in practice, we need to leverage the powerful toolbox of automatic differentiation. To this end, Schulman et al. (2015) introduce the *surrogate loss* (SL) within the formalism of *stochastic computation graphs* (SCG). Single differentiation of the SL produces an unbiased gradient estimator for the first order gradient of the primary objective.

However, in many settings this is not sufficient because it might also be essential to estimate higher order gradients of these stochastic objectives. For example, in the multi-agent learning method *learning with opponent-learning awareness* (LOLA) (Foerster et al., 2018a), the expected return of one agent is differentiated through the learning step of another agent. Similarly, when gradient-based approaches to meta-learning are applied to reinforcement learning problems, (Finn et al., 2017; Al-Shedivat et al., 2017), the objective function is repeatedly differentiated through a hierarchy of learning processes. Furthermore, higher order gradients are used directly by some optimisation algorithms, such as the quasi-Newton algorithm (Wright & Nocedal, 1999).

Estimating higher order gradients accurately and efficiently in such applications is challenging because such estimators are complex to construct and often have extremely high variance. It is therefore critical to develop methods that produce low variance estimates of higher order gradients while also being easy to use in practice through automatic differentiation.

Given an objective function with random variables, the naive approach is to derive the corresponding higher order gradient estimators analytically. Al-Shedivat et al. (2017) apply the score function trick repeatedly to obtain estimators for higher order gradients in a meta-learning setting. Foerster et al. (2018a) combine this approach with a Taylor expansion to generate higher order gradient estimators in a multi-agent setting. However, this approach is incompatible with automatic differentiation and is thus error prone and difficult to generalise to arbitrary objective functions.

Schulman et al. (2015) argue that the gradient estimate produced by differentiating an SL objective may be treated as an objective itself, and a new SL can be built to estimate higher order gradients. Unfortunately, this approach can produce incorrect higher order gradient estimates (Foerster et al., 2018b) because it treats cost terms as fixed samples in the SL and thus does not maintain all necessary dependencies in the SCG after differentiation. By contrast, the *infinitely differentiable Monte-Carlo estimator* (DiCE) (Foerster et al., 2018b) estimates higher order gradients correctly while maintaining ease of use by leveraging automatic differentiation. DiCE consists of a single objective that can be differentiated an arbitrary number of times in order to generate higher order gradient estimators.

One strategy that DiCE uses to reduce variance is to introduce *control variates* or *baselines* (Paisley et al., 2012; Tucker et al., 2017; Grathwohl et al., 2017). While DiCE's baseline is effective for first order gradients, its higher order gradient estimates still have high variance.

In this paper, we propose a novel baseline term for any-order gradient estimation in DiCE. This term can be easily included in the original objective without changing the estimate of the objective itself or of the first order gradients. Nonetheless, repeated differentiation of the modified objective automatically produces appropriate control variates for the higher order gradients.

Furthermore, the new baseline term includes an additional dependence that properly captures the causal relationship between stochastic nodes in an SCG. However, this baseline term can be constructed using the same baseline function already used for the first order baseline. Using DiCE, we can preserve these dependencies through differentiation.

We prove that adding our new baseline term to the DiCE objective leaves all gradient estimators of DiCE unbiased, while reducing variance. We also explicitly derive the second order gradients of the DiCE objective with our baseline term in order to demonstrate how it introduces control variates for higher order gradient estimates.

Finally, we conduct a series of numerical evaluations to verify the correctness of our baseline term and empirically show its effectiveness for variance reduction of gradient estimates of up to fourth order. By evaluating on the tasks of the original DiCE paper, we show that we can reduce sample complexity by two orders of magnitude compared to the original DiCE formulation.

We believe that this new baseline term is a key component that can make DiCE a powerful and practical tool for easily obtaining low-variance higher order gradient estimators for reinforcement learning and meta-learning applications.

2. Background

Suppose x is a random variable distributed as $x \sim p(x;\theta)$, and $f(x)$ is a deterministic function of x. We assume that $f(x)$ is independent of θ so that $\nabla_\theta^n f = 0$ for $n \in \{1, 2, \dots\}$. Suppose we have the objective function $\mathcal{L}(\theta) = \mathbb{E}_x[f(x)]$ and we need to compute the gradients of the expectation, $\nabla_\theta \mathbb{E}_x[f(x)]$, in order to use gradient-based optimisation methods. An unbiased gradient estimator g is a random variable such that $\mathbb{E}[g(f)] = \nabla_\theta \mathbb{E}[f(x)]$. The *score function estimator* (Fu, 2006) is an unbiased estimator given by

$$\mathbb{E}_x[f(x)\nabla_\theta \log p(x;\theta)] = \nabla_\theta \mathbb{E}_x[f(x)]. \tag{1}$$

The *control variates* method is a variance reduction technique for improving the efficiency of Monte Carlo estimators. A *control variate* is a function $c(x)$ whose value can be easily obtained, and whose expectation $\mathbb{E}[c(x)]$ is known. Using the control variate, we can construct a new estimator

$$z(f) = g(f) - \alpha\Big[c(x) - \mathbb{E}_x[c(x)]\Big], \tag{2}$$

where $\alpha \in \mathbb{R}$ is a parameter. The expectation of $z(f)$ is

$$\mathbb{E}[z(f)] = \mathbb{E}_x[g(f)] - \alpha\,\mathbb{E}_x[c(x)] + \alpha\,\mathbb{E}_x[c(x)] = \mathbb{E}_x[f(x)].$$

Consequently, if $g(f)$ is an unbiased estimator of the gradient, so is $z(f)$. However, if $g(f)$ and $c(x)$ are sufficiently correlated, then there exists a setting of α such that the variance of $z(f)$ is lower than that of $g(f)$ (Grathwohl et al., 2017). The optimal α is typically difficult to estimate so in practice it is common to set $\alpha = 1$ and design $c(x)$ to be positively correlated with $g(f)$. In the case of the score function estimator it is straight forward to see that we can construct a positively correlated $c(x)$ by considering the average cost, $V = \mathbb{E}_x[f(x)]$:

$$c(x) = V\nabla_\theta \log p(x;\theta) \tag{3}$$

The variance reduced gradient estimator is then:

$$z(f) = \big(f(x) - V\big)\nabla_\theta \log p(x;\theta). \tag{4}$$

Note that this $c(x)$ is zero-mean by construction, so the adjustment term $-\,\mathbb{E}_x[c(x)]$ is missing. Due to its practical importance and simplicity we will be using this general form of variance reduction throughout the paper.

2.1. Stochastic Computation Graphs

A *stochastic computation graph* (SCG) (Schulman et al., 2015) is a directed and acyclic graph that consists of three types of nodes:

1. **Input nodes Θ:** an input node $\theta \in \Theta$ contains parameters set externally. We may be interested in the dependence of an objective function on these nodes, and will differentiate the objective with respect to their parameters.
2. **Deterministic nodes $\mathcal{D}$:** a deterministic node $d \in \mathcal{D}$ is a deterministic function of its parent nodes.
3. **Stochastic nodes $\mathcal{S}$:** a stochastic node $w \in \mathcal{S}$ is a random variable whose distribution is conditioned on its parent nodes.

Additionally, cost nodes $\mathcal{C}$ can be added into the formalism of an SCG without loss of generality. A cost node $c \in \mathcal{C}$ is a deterministic function of its parent nodes that produces a scalar value. The set of cost nodes $\mathcal{C}$ are those associated with an objective function $\mathcal{L} = \mathbb{E}[\sum_{c\in\mathcal{C}} c]$. In an SCG, (v, w) is a directed edge that connects node v and non-input node w, where v is a parent node of w. The notation $v \prec w$ means there is a path from node v to node w, i.e., node v influences node w.

2.2. DiCE

In order to estimate higher order gradients correctly and efficiently within the formalism of SCGs, the *infinitely differentiable Monte-Carlo estimator* (DiCE) (Foerster et al., 2018b) uses the *magic box* $\boxdot$ operator. The input to $\boxdot$ is a set of stochastic nodes $\mathcal{W}$, and $\boxdot$ is designed to have the following properties:

1. $\boxdot(\mathcal{W}) \rightarrowtail 1$,
2. $\nabla_\theta \boxdot(\mathcal{W}) = \boxdot(\mathcal{W}) \sum_{w\in\mathcal{W}} \nabla_\theta \log p(w;\theta)$.

Here $\rightarrowtail$ means "evaluates to", which is different from "equals to", $=$, i.e., full equality including equality of all derivatives. In the context of a computation graph, $\rightarrowtail$ denotes a forward pass evaluation. By contrast, the second property describes the behaviour of $\boxdot$ under differentiation. The righthand side of this equality can in turn be evaluated to estimate gradients as described below, or differentiated further. These properties are easily satisfied with the following implementation:

$$\boxdot(\mathcal{W}) = \exp(\tau - \bot(\tau)),$$

where $\tau = \sum_{w\in\mathcal{W}} \nabla_\theta \log p(w;\theta)$ and $\bot$ is a 'stop-grad' operator that sets the derivative to zero, $\nabla_x \bot(x) = 0$, and is commonly available in auto-differentiation libraries.

For a node w in an SCG, we use $\mathcal{S}_w$ to denote the set of stochastic nodes that influence the node w and are influenced by θ, i.e., $\mathcal{S}_w = \{s | s \in \mathcal{S}, s \prec w, \theta \prec s\}$. Using $\boxdot$, the DiCE objective is

$$\mathcal{L}_{\boxdot} = \sum_{c\in\mathcal{C}} \boxdot(\mathcal{S}_c) c. \tag{5}$$

Under repeated differentiation, the DiCE objective generates arbitrary order gradient estimators (Foerster et al., 2018b, Theorem 1): $\mathbb{E}[\nabla^n_\theta \mathcal{L}_{\boxdot}] \rightarrowtail \nabla^n_\theta \mathcal{L}$, for $n \in \{0, 1, 2, \dots\}$.

2.3. Variance Reduction with DiCE

$\mathcal{L}_{\boxdot}$ by itself already implements a simple form of variance reduction by respecting causality. In gradient estimates, each cost node c is multiplied by the sum of gradients of log-probabilities of only upstream nodes $\mathcal{S}_c$ that can influence c. This reduces variance compared to using the log joint probability of *all* stochastic nodes, which would still create an unbiased gradient estimate.

However, DiCE further reduces variance for first-order gradient estimation by including a baseline

$$\mathcal{B}^{(1)}_{\boxdot} = \sum_{w\in\mathcal{S}} (1 - \boxdot(\{w\})) b_w, \tag{6}$$

where b_w is a function of the set $\text{NONINFLUENCED}(w) = \{v | w \not\prec v\}$, i.e., the set of nodes that does not influence w.[1] Here b_w may be chosen to reduce variance, and a common choice for b_w is the average cost-to-go, i.e., $\mathbb{E}[R_w | \text{NONINFLUENCED}(w))]$, where $R_w = \sum_{c\in\mathcal{C}_w} c$, where $\mathcal{C}_w = \{c | c \in \mathcal{C}, w \prec c\}$, i.e., the set of cost nodes that depend on node w.[2] In order to maintain unbiased gradient estimates, the baseline factor b_w should be a function that is independent of the stochastic node w. Greensmith et al. (2004) provide an overview of variance reduction techniques for gradient estimators, including the use of this type of baseline. The baseline term $\mathcal{B}^{(1)}_{\boxdot}$ can be added to the DiCE objective to obtain

$$\mathcal{L}^{b_1}_{\boxdot} = \mathcal{L}_{\boxdot} + \mathcal{B}^{(1)}_{\boxdot}. \tag{7}$$

$\mathcal{L}^{b_1}_{\boxdot}$ evaluates to the same value as $\mathcal{L}_{\boxdot}$ because $(1 - \boxdot(\mathcal{W})) \rightarrowtail 0$.

3. Method

To provide intuition for our new baseline, we first consider explicitly the effect of the first order baseline on the first or-

[1] b(NONINFLUENCED(w)) in (Schulman et al., 2015).

[2] We use the R_w notation to correspond with a *return* for readers familiar with reinforcement learning, a key use case. The cost notation for nodes is kept to maintain consistency with Schulman et al. (2015).

der gradient estimates. The estimates without and with baseline are as follows (derivations are given in Appendix A):

$$\nabla_\theta \mathcal{L}_{\boxdot} \rightarrowtail \sum_{w \in \mathcal{S}} R_w \nabla_\theta \log p(w;\theta),$$
$$\nabla_\theta \mathcal{L}^{b_1}_{\boxdot} \rightarrowtail \sum_{w \in \mathcal{S}} (R_w - b_w) \nabla_\theta \log p(w;\theta). \tag{8}$$

For each stochastic node $w \in \mathcal{S}$, the term $b_w \nabla_\theta \log p(w;\theta)$ works as a control variate to reduce the variance of the term $R_w \nabla_\theta \log p(w;\theta)$. To ensure the appropriate correlations, b_w may be a function trained to estimate $\mathbb{E}[R_w | \text{NONINFLUENCED}(w))]$. As a result, $\nabla_\theta \mathcal{L}^{b_1}_{\boxdot}$ is a first order gradient estimator with lower variance than $\nabla_\theta \mathcal{L}_{\boxdot}$. Additionally, $\nabla_\theta \mathcal{L}^{b_1}_{\boxdot}$ is unbiased because $\mathbb{E}[\nabla_\theta \mathcal{B}^{(1)}_{\boxdot}] \rightarrowtail 0$ (see Appendix A).

In (8) we have omitted a term, $\sum_{c \in \mathcal{C}} \nabla_\theta c$, that arises when the cost nodes depend directly on θ. In most use cases this term does not appear, as the costs are sampled from an unparameterised process, such as the unknown environment in reinforcement learning. This straight-through contribution to the gradient estimate is also typically much lower variance than the contribution estimated using the score function trick. Due to these considerations, we assume that the costs are independent of θ in the remainder of this work, although the remaining terms for both first and second order are derived in the appendix.

Next, we consider second-order gradient estimation using the objective $\mathcal{L}^{b_1}_{\boxdot}$, including the first order baseline. The second order gradient of the DiCE objective can be evaluated as follows (derivations are given in Appendix B):

$$\nabla^2_\theta \mathcal{L}_{\boxdot} \rightarrowtail \sum_{w \in \mathcal{S}} R_w \frac{\nabla^2_\theta p(w;\theta)}{p(w;\theta)} + \text{SUM}, \tag{9}$$
$$\nabla^2_\theta \mathcal{L}^{b_1}_{\boxdot} \rightarrowtail \sum_{w \in \mathcal{S}} (R_w - b_w) \frac{\nabla^2_\theta p(w;\theta)}{p(w;\theta)} + \text{SUM}, \tag{10}$$

where

$$\text{SUM} = 2 \sum_{w \in \mathcal{S}} R_w \nabla_\theta \log p(w;\theta) \Big[\sum_{v \in \mathcal{S}, v \prec w} \nabla_\theta \log p(v;\theta) \Big].$$

The baseline objective function $\mathcal{L}^{b_1}_{\boxdot}$ still implements a partial variance reduction of the second order gradient estimates, by providing a control variate for the first term in (10). However, the R_v in the second term are not paired with suitable control variates. As a result, the variance of this term could be extremely high. In fact, due to the nested summations over the high-variance R_v, this term can dominate the variance of the total gradient estimate. We explore this empirically in Section 4, where we observe that $\mathcal{B}^{(1)}_{\boxdot}$ is of little use for reducing the overall variance of the second order gradient estimates.

3.1. A Second Order Baseline

Before proposing an any order baseline (Section 3.2), we first introduce a second-order baseline with the goal of reducing variance for the following term:

$$2 \sum_{v \in \mathcal{S}} R_v \nabla_\theta \log p(v;\theta) \Big[\sum_{w \in \mathcal{S}, w \prec v} \nabla_\theta \log p(w;\theta) \Big].$$

To this end, we want to pair the R_v term with a baseline $-b_v$, similar to the first order gradients variance reduction. At the same time, the zeroth and first order gradients must remain unaffected by the second order baseline.

We thus have the following requirements for our second order baseline, $\mathcal{B}^{(2)}_{\boxdot}$:

1. $\mathcal{B}^{(2)}_{\boxdot}$ evaluates to 0: $\mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0$
2. The gradient of $\mathcal{B}^{(2)}_{\boxdot}$ evaluates to 0: $\nabla_\theta \mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0$
3. The second derivative of $\mathcal{B}^{(2)}_{\boxdot}$ evaluates to the variance reduction for the second order gradient.

The third requirement imposes the following condition on our second order baseline:

$$\nabla^2_\theta \mathcal{B}^{(2)}_{\boxdot} \rightarrowtail$$
$$-2 \sum_{w \in \mathcal{S}} b_w \nabla_\theta \log p(w;\theta) \Big[\sum_{v \in \mathcal{S}, v \prec w} \nabla_\theta \log p(v;\theta) \Big].$$

As with the first order baseline, we can satisfy the first requirement by constructing the baseline of the form $(1 - \boxdot)$, for an arbitrary input X:

$$\mathcal{B}^{(2)}_{\boxdot} = (1 - \boxdot(X)) \rightarrowtail (1 - 1) = 0.$$

Similarly, we can satisfy the second requirement by constructing the baseline as a product of two terms of the form $(1 - \boxdot)$, for arbitrary inputs X and Y:

$$\mathcal{B}^{(2)}_{\boxdot} = \big(1 - \boxdot(X)\big)\big(1 - \boxdot(Y)\big)$$
$$\nabla_\theta \mathcal{B}^{(2)}_{\boxdot} = \big(\boxdot(X) - 1\big)\nabla_\theta \boxdot(Y) - \nabla_\theta \boxdot(X)\big(1 - \boxdot(Y)\big)$$
$$\rightarrowtail 0 + 0 = 0$$

The second order derivative contains both terms of the form $(1 - \boxdot)\nabla^2_\theta \boxdot$, which evaluate to zero, and one term of the form:

$$2 \nabla_\theta \boxdot(X) \nabla_\theta \boxdot(Y). \tag{11}$$

From the third requirement it is clear that $\mathcal{B}^{(2)}_{\boxdot}$ has to be a sum of terms involving the different b_w, each of which is of the form given above:

$$\mathcal{B}^{(2)}_{\boxdot} = -\sum_{w\in\mathcal{S}} b_w\big(1-\boxdot(X_w)\big)\big(1-\boxdot(Y_w)\big) \tag{12}$$

Using the definition of DiCE:

$$\nabla_\theta \boxdot(X) \to \sum_{x\in X} \nabla_\theta \log p(x;\theta), \tag{13}$$

we can now expand the second derivative:

$$\nabla_\theta \mathcal{B}^{(2)}_{\boxdot} \to -\sum_{w\in\mathcal{S}} b_w 2\nabla_\theta\boxdot(X_w)\nabla_\theta\boxdot(Y_w) \tag{14}$$

$$= -\sum_{w\in\mathcal{S}} b_w \sum_{x\in X_w} \nabla_\theta \log p(x;\theta)\big(\sum_{y\in Y_w} \nabla_\theta \log p(y;\theta)\big). \tag{15}$$

Finally, by inspecting the second derivative term above and our third requirement, we set $X_w = \{w\}$ and $Y_w = \mathcal{S}_w$ to obtain:

$$\mathcal{B}^{(2)}_{\boxdot} = -\sum_{w\in\mathcal{S}} \big(1-\boxdot(\{w\})\big)\big(1-\boxdot(\mathcal{S}_w)\big)b_w. \tag{16}$$

As we will show below, this baseline term produces the appropriate variance reduction required for the second order gradient terms mentioned above, when differentiated twice. Note that $\mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0$ because $(1-\boxdot(\{w\})) \rightarrowtail 0$ and $(1-\boxdot(\mathcal{S}_w)) \rightarrowtail 0$, and b_w is the same as that used in $\mathcal{B}^{(1)}_{\boxdot}$.

3.2. Any Order Baseline

We can now produce our final baseline by adding $\mathcal{B}^{(1)}_{\boxdot}$ and $\mathcal{B}^{(2)}_{\boxdot}$:

$$\mathcal{B}_{\boxdot} = \mathcal{B}^{(1)}_{\boxdot} + \mathcal{B}^{(2)}_{\boxdot} \tag{17}$$

$$= \sum_{w\in\mathcal{S}} b_w\big(1-\boxdot(\{w\})\big)\Big(1-\big(1-\boxdot(\mathcal{S}_w)\big)\Big) \tag{18}$$

$$\mathcal{B}_{\boxdot} = \sum_{w\in\mathcal{S}} b_w\big(1-\boxdot(\{w\})\big)\cdot\boxdot(\mathcal{S}_w). \tag{19}$$

This baseline has a very intuitive interpretation: It is the original $\mathcal{B}^{(1)}_{\boxdot}$ which now includes the $\boxdot(\mathcal{S}_w)$ term which captures the causes of each of the variance reduction terms. Using the DiCE mechanism, this small change produces the correct variance reduction for higher order derivatives under automatic differentiation. Below we show this analytically for the first and second order gradients, in the supplementary material we also show a partial proof for variance reduction of some of the terms for any order gradients. In the experimental section we empirically verify variance reduction for up to fourth order gradients.

The new DiCE objective function becomes: $\mathcal{L}^{b_2}_{\boxdot} = \mathcal{L}_{\boxdot} + \mathcal{B}_{\boxdot}$. Since $\mathcal{B}^{(1)}_{\boxdot} \rightarrowtail 0$ and $\mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0$, $\mathcal{L}_{\boxdot}$, $\mathcal{L}^{b_1}_{\boxdot}$, and $\mathcal{L}^{b_2}_{\boxdot}$ all evaluate to the same estimate of the original objective. Furthermore, all derivatives of our modified objective $\mathcal{L}^{b_2}_{\boxdot}$ are unbiased estimators of the derivatives of the original objective, but now containing suitable control variates for variance reduction. We now show how this baseline term indeed reduces variance while remaining unbiased for the higher order derivatives of our objective.

3.3. Bias and Variance Analysis

In the DiCE objective $\mathcal{L}_{\boxdot}$, for each cost node $c \in \mathcal{C}$, the corresponding $\boxdot(\mathcal{S}_c)$ reflects the dependency of c on all stochastic nodes that influence it (and depend on θ). By contrast, the baseline term $\mathcal{B}^{(1)}_{\boxdot}$ only includes $\boxdot(\{w\})$, considering each stochastic node w separately. This simple approach results in variance reduction for first order gradients, as shown in (8). However, the failure to capture the dependence of stochastic nodes on each other in the simple baseline prevents it from reducing the variance of the cross terms that arise in second order derivatives (the final term in (10)). To capture these relationships properly, we include $\boxdot(\mathcal{S}_w)$ in the definition of $\mathcal{B}^{(2)}_{\boxdot}$, i.e., an additional dependence on the stochastic nodes that influence w. Use of the $\boxdot$ operator ensures that these dependencies are preserved through differentiation.

To verify that our proposed baseline indeed captures these dependencies appropriately, we now consider its impact on the gradient estimates. The first and second order gradients of the baseline term $\mathcal{B}^{(2)}_{\boxdot}$ can be evaluated as follows (derivations given in Appendix B):

$$\begin{aligned}&\nabla_\theta \mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0,\\ &\nabla^2_\theta \mathcal{B}^{(2)}_{\boxdot} \rightarrowtail -2\sum_{w\in\mathcal{S}} \nabla_\theta \log p(w;\theta)\mathrm{SUM}^1_w,\end{aligned} \tag{20}$$

where $\mathrm{SUM}^1_w = \sum_{v\in\mathcal{S}, w\prec v} b_v \nabla_\theta \log p(v;\theta)$.

The first order gradient estimates remain unchanged: as $\nabla_\theta \mathcal{B}^{(2)}_{\boxdot} \rightarrowtail 0$, $\nabla_\theta \mathcal{L}^{b_2}_{\boxdot}$ evaluates to the same value as $\nabla_\theta \mathcal{L}^{b_1}_{\boxdot}$. The second order gradient estimate of our full objective, $\nabla^2_\theta \mathcal{L}^{b_2}_{\boxdot}$, is as follows:

$$\begin{aligned}\nabla^2_\theta \mathcal{L}^{b_2}_{\boxdot} \rightarrowtail &\sum_{w\in\mathcal{S}} (R_w - b_w)\frac{\nabla^2_\theta p(w;\theta)}{p(w;\theta)}\\ &+ 2\sum_{w\in\mathcal{S}} \nabla_\theta \log p(w;\theta)\mathrm{SUM}^2_w,\end{aligned} \tag{21}$$

where $\mathrm{SUM}^2_w = \sum_{v\in\mathcal{S}, w\prec v}(R_v - b_v)\nabla_\theta \log p(v;\theta)$.

Control variates have been introduced for the terms in the second part of (21), when using our new baseline term $\mathcal{B}^{(2)}_{\boxdot}$. Here we assume that R_w and b_w are correlated by design, as they should be for variance reduction of the first order gradients. As a result, the estimator $\nabla^2_\theta \mathcal{L}^{b_2}_{\boxdot}$ could have significantly lower variance compared with $\nabla^2_\theta \mathcal{L}^{b_1}_{\boxdot}$ and $\nabla^2_\theta \mathcal{L}_{\boxdot}$, as we verify empirically in Section 4.

Furthermore, we verify that our baseline does not change the expected estimate of second order derivatives.

Theorem 1. $\mathbb{E}\left[\nabla^2_\theta \mathcal{L}^{b_2}_{\boxdot}\right] \rightarrowtail \nabla^2_\theta \mathbb{E}\left[\mathcal{L}\right]$.

Proof. First, we can prove that $\mathbb{E}[\nabla^2_\theta \mathcal{B}^{(1)}_{\boxdot}] \rightarrowtail 0$ and $\mathbb{E}[\nabla^2_\theta \mathcal{B}^{(2)}_{\boxdot}] \rightarrowtail 0$ (see Appendix). Since $\nabla^2_\theta \mathcal{L}_{\boxdot}$ is an unbiased estimator of $\nabla^2_\theta \mathbb{E}\left[\mathcal{L}\right]$, i.e., $\nabla^2_\theta \mathcal{L}_{\boxdot} \rightarrowtail \nabla^2_\theta \mathbb{E}\left[\mathcal{L}\right]$, then:

$$\mathbb{E}\left[\nabla^2_\theta \mathcal{L}^{b_2}_{\boxdot}\right] = \mathbb{E}\left[\nabla^2_\theta \mathcal{L}_{\boxdot}\right] + \mathbb{E}\left[\nabla^2_\theta \mathcal{B}^{(1)}_{\boxdot}\right] + \mathbb{E}\left[\nabla^2_\theta \mathcal{B}^{(2)}_{\boxdot}\right] \rightarrowtail \nabla^2_\theta \mathbb{E}\left[\mathcal{L}\right].$$

□

Thus, $\nabla^2_\theta \mathcal{L}^{b}_{\boxdot}$ is an unbiased second order gradient estimator of the original objective $\mathbb{E}\left[\mathcal{L}\right]$.

For simplicity we only carry out this analysis analytically for the second order gradients, which happen to be the most common use case. However, in Section 4 and Appendix C we show that our baseline also substantially reduces the variance of third and fourth order gradients, which converge to the true derivatives with increasing numbers of samples.

3.4. Reinforcement Learning

We now consider the particular case of reinforcement learning. Given a policy π, we can generate an episode of horizon T:

$$\tau = (s_0, a_0, r_0, \ldots, s_T, a_T, r_T).$$

The discounted return at time step t is the discounted sum of future rewards, $R_t(\tau) = \sum_{k=t}^{T} \gamma^{k-t} r_t$, where $\gamma \in [0, 1]$ is a discount factor. When the reinforcement learning problem is formalised as an SCG, the cost nodes are the discounted rewards and the objective function is $\mathcal{L} = \mathbb{E}[\sum_{t=0}^{T} \gamma^t r_t]$. The corresponding DiCE objective function is:

$$\mathcal{L}_{\boxdot} = \sum_{t=0}^{T} \boxdot(a_{t' \leq t}) \cdot \gamma^t r_t, \tag{22}$$

where $a_{t' \leq t}$ is the set of all previous actions at time step t, i.e., $a_{t' \leq t} = \{a_0, a_1, \ldots, a_t\}$. Clearly, these are the stochastic nodes that influence the reward at time t. We choose the baseline $b(s_t)$ to be a function of state s_t; it must be independent of the action a_t. In particular, we choose $b(s_t) = \gamma^t \hat{V}(s_t)$, where $\hat{V}(s_t)$ is an estimate of the state value function $V^\pi(s_t) = \mathbb{E}[R_t | s_t]$. First order variance reduction may now be achieved with the baseline term:

$$\mathcal{B}^{(1)}_{\boxdot} = \sum_{t=0}^{T} (1 - \boxdot(a_t)) b(s_t). \tag{23}$$

To reduce the variance of any-order gradient estimators, we can use our novel baseline term:

$$\mathcal{B}^{(2)}_{\boxdot} = -\sum_{t=0}^{T} \left(1 - \boxdot(a_t)\right) \cdot \boxdot(a_{t' < t}) b(s_t). \tag{24}$$

These baseline terms can be added to our original objective. As in the general case, the corresponding DiCE objectives with baselines are $\mathcal{L}^{b_1}_{\boxdot} = \mathcal{L}_{\boxdot} + \mathcal{B}^{(1)}_{\boxdot}$ and $\mathcal{L}^{b}_{\boxdot} = \mathcal{L}_{\boxdot} + \mathcal{B}_{\boxdot}$.

In $\boxdot(a_{t' < t})$, we need to have strict inequality $t' < t$, which captures the causality from all previous actions. The agent is able to look backward at its past actions but excludes its current action. Since there is no previous action at $t = 0$, at this time step we are evaluating $\boxdot(a_{t' < t})$ on the empty set, which simply equals 1.

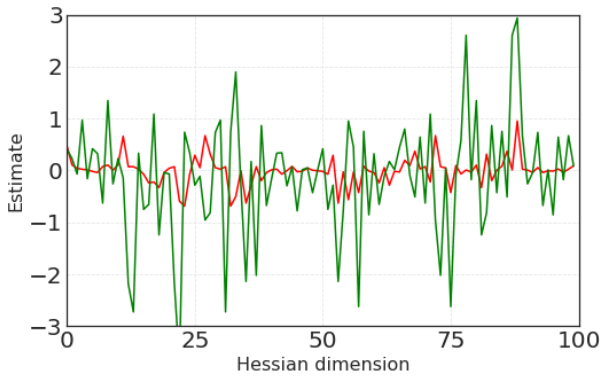
In our formulation, the second order baseline reuses the same state-value function that is also used for the first order baseline in the original formulation of the DiCE objective. The main benefit is that state-value function baselines do not change the expected gradients of the objective. This value function baseline has a strong empirical track record and is used in virtually all policy gradient methods, in particular in large scale applications such as A3C and PPO (Mnih et al., 2016; Schulman et al., 2017). However, action-dependent baselines are currently an active area of investigation (Tucker et al., 2018). While action-dependent baselines in general introduce bias, in some settings that bias can be exactly or approximately removed. In addition, Tucker et al. (2018) show that variance decomposition can be used to improve performance, which could be combined with our approach in future work.

	C	D
C	-1 -1	0 -3
D	-3 0	-2 -2

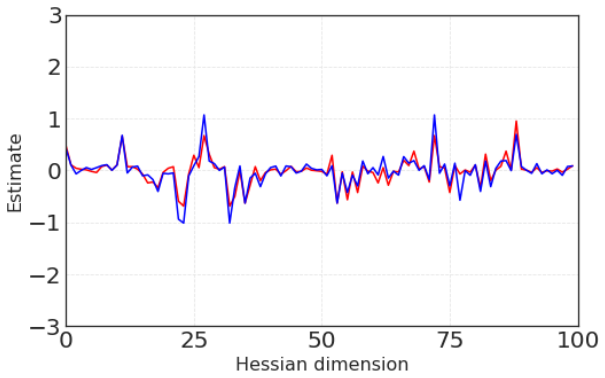
Table 1. The payoff matrix of prisoner's dilemma. Numbers in a cell correspond to the utilities of the player with the same colour.

4. Experiments

First, we numerically verify that DiCE with our new baseline term $\mathcal{L}^{b_2}_{\boxdot}$ can generate correct estimators of the Hessians in an SCG using a set of randomly initialised fixed policies in the *iterated prisoner's dilemma*.



(a) Flattened Hessian (green) of $\mathcal{L}_{\boxdot}^{b_1,1}$. Correlation coeff.: 0.29.



(b) Flattened Hessian (blue) of $\mathcal{L}_{\boxdot}^{b_2,1}$. Correlation coeff.: 0.97.

Figure 1. Flattened true (Red) and estimated Hessian of agent 1 for the iterated prisoner's dilemma. The sample size is 1000.

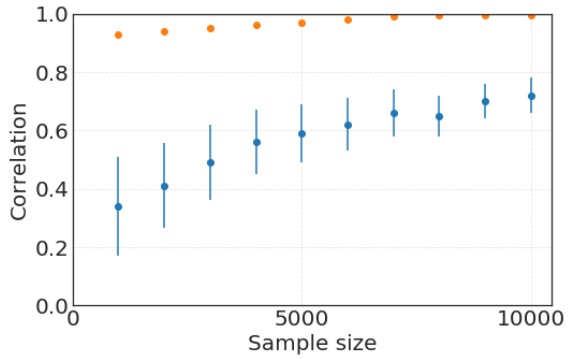


Figure 2. Correlation coefficients of the exact Hessian and the estimated Hessian generated from the multi-agent DiCE objective function with (orange) and without (blue) the second order baseline term $\mathcal{B}_{\boxdot}^{(2),i}$. The error bars show the standard deviation.

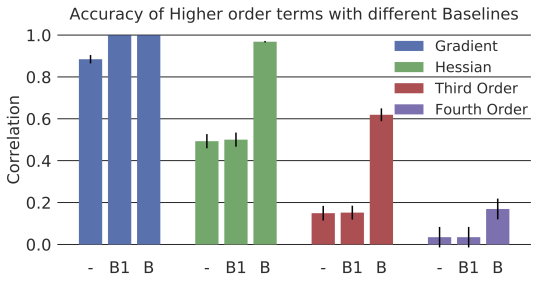


Figure 3. Mean correlation of first to fourth order true gradients with gradient estimators using no baseline (-), a first order baseline (B1), or our new combined baseline (B), given 1000 samples. The black error bars indicate the standard deviation.

To demonstrate the improvement of our baseline on the original DiCE, we use the same setting as Foerster et al. (2018b) to reproduce their experimental results for comparison.

In this setting, two agents play the game of the prisoner's dilemma iteratively. At each round, there are two possible actions for each agent, which are Cooperate (C) and Defect (D). As a result, there are four possible outcomes, CC, CD, DC, and DD at each round, which are the observation at the next time step. The payoff matrix is given in Table 1.

4.1. Multi-agent DiCE

The objective function for agent i is $\mathcal{L}^i = \mathbb{E}[\sum_{t=0}^{T} \gamma^t r_t^i]$. The per-agent DiCE objective $\mathcal{L}_{\boxdot}^i$ is a simple extension of (22), replacing r_t by the per-agent reward r_t^i, and a_t by the joint action $a_t^{j\in\{1,2\}}$. For correct higher order gradients it is essential to consider the dependence of the reward on the actions of both agents in this way. We require per-agent baseline factors $b^i(s_t)$ to then form per-agent baseline terms $\mathcal{B}_{\boxdot}^{i,(1)}$ and $\mathcal{B}_{\boxdot}^{i,(2)}$ in analogy with (23, 24). Again, the single-agent action at each timestep is replaced by the joint action.

Using DiCE, the dependencies between the returns and parameters of the two agents are accounted for, and first and second order gradients can be estimated efficiently using automatic differentiation.

We empirically test and compare the performance of $\mathcal{L}_{\boxdot}^{b_1,i}$ and $\mathcal{L}_{\boxdot}^{b,i}$ in gradient estimation up to fourth order. Foerster et al. (2018a) derive the value function of the IPD analytically, which we use as ground truth to verify the correctness of our estimator. Figures 1(a) and 1(b) show the flattened Hessians, i.e., reshaped into a 100 dimensional vector, of $\mathcal{L}_{\boxdot}^{b_1,1}$ and $\mathcal{L}_{\boxdot}^{b_2,1}$, compared to the true flattened Hessian. These results show that our novel baseline term dramatically improves the estimation of second order gradients.

Foerster et al. (2018b) needed a sample size of 100k to obtain second order gradient estimates with a correlation coefficient 0.97. Our baseline term reduces the required

sample size by two orders of magnitude, to 1k[3]. Figure 2 shows the correlation coefficients of the exact Hessian and the estimated Hessian using different sample sizes. These results demonstrate that our baseline term is important for estimating second order gradients accurately and efficiently when using DiCE.

Figure 3 extends the empirical analysis to higher order gradients by showing the correlation of first to fourth order gradient estimators with the true gradients given the different baselines. For higher order gradients, the first order baseline does not substantially improve the correlation with the true gradients. By contrast, our new combined baseline substantially improves correlation for all higher order gradients shown. In particular, without the new baseline, the fourth order gradient estimator is not significantly correlated with the true gradient.

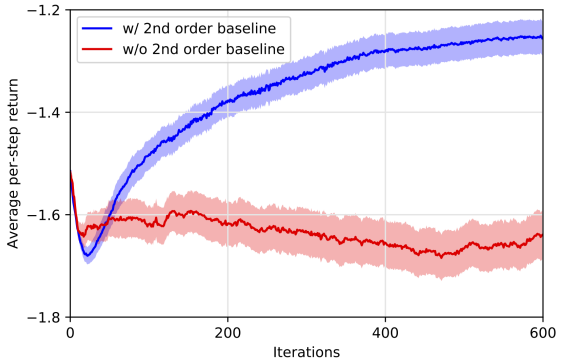


Figure 4. The performance of the LOLA-DiCE algorithm on the IPD with (blue) and without (red) the new second order baseline. Shading indicates the error of the mean.

4.2. LOLA-DiCE

In LOLA-DiCE (Foerster et al., 2018b) agents differentiate through the learning step of other agents, using the DiCE objective to calculate higher order derivatives:

$$\mathcal{L}^1(\theta_1, \theta_2)_{\text{LOLA}} = \mathbb{E}_{\pi_{\theta_1}, \pi_{\theta_2} + \Delta\theta_2(\theta_1,\theta_2)}[\sum_{t=0}^{T} \gamma^t r_t^1],$$

where $\Delta\theta_2(\theta_1, \theta_2) = \alpha_2 \nabla_{\theta_2} \mathbb{E}_{\pi_{\theta_1}, \pi_{\theta_2}}[\sum_{t=0}^{T} \gamma^t r_t^2]$ and α_2 is a step size.

We use a tabular policy and value function, initialised from a normal distribution with unit variance and zero mean. The discount factor, $\gamma = 0.96$, and the episodes are truncated after 150 steps. Our experiments use a batch size of 32, a learning rate of 0.05 for the policy, and a lookahead step size, $\alpha = 0.3$. To allow for proper variance reduction, we train two value functions, one for the inner loop and one for the outer loop, which are pre-trained over 200 training steps. To ensure that the value function closely tracks the value under the changing policy, we carry out 10 training steps of the value functions with a learning rate of 0.1 for each policy update.

Figure 4 shows the performance of LOLA-DiCE with and without our second order baseline. Without the second order baseline, agents fail to learn, yielding an average per-step return of around -1.6, close to that of a random policy, which achieves -1.5. By halving the batch size (32 vs 64), performance is comparable to that of the original work. However, our results using the first order baseline are much worse than what is reported in the original work, albeit at a smaller batch size. In communication with the authors we established that those results were produced by making the rewards at each timestep zero-mean within each batch, rather than relying on the first order baseline.

In settings where the value function is mostly independent of the state, which happens to be the case in the IPD with a large γ, this simple trick can produce variance reduction similar to what we achieve with our second order baseline. However, this ad-hoc normalisation would fail in settings with sparser rewards or in any setting where the value function strongly depends on the current state.

5. Conclusion

Recent progress in multi-agent reinforcement learning and meta-learning has lead to a variety of approaches that employ second order gradient estimators. While these are easy to construct with the DiCE objective (Foerster et al., 2018b), the high variance of higher order gradient estimators has prevented their widespread application in practice. By reusing the DiCE formalism, we introduce a baseline for second order gradient estimators in stochastic computation graphs. Similar to DiCE, this baseline is automatically constructed from user-defined objectives using automatic differentiation frameworks, making it straightforward to use in practice. Our baseline does not change the expected value of any derivatives. We demonstrate empirically that our new baseline dramatically improves higher order gradient estimation in a multi-agent task, reducing the required sample size by two orders of magnitude. We believe that low-variance higher order gradient estimators will unlock a variety of reinforcement learning and meta-learning applications in the future. Furthermore, we would like to extend the approach to deal with settings where the costs depend directly on the parameters. Lastly, we are interested in extending our framework to a baseline-generating term for any-order gradient estimators.

[3]The computation associated with applying and differentiating the magic box operator is negligible compared to the other computations.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement #637713).

The experiments were made possible by a generous equipment grant from NVIDIA.

References

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. *CoRR*, abs/1710.03641, 2017.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, pp. 1126–1135, 2017.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130. International Foundation for Autonomous Agents and Multiagent Systems, 2018a.

Foerster, J., Farquhar, G., Al-Shedivat, M., Rocktaeschel, T., Xing, E., and Whiteson, S. Dice: The infinitely differentiable monte carlo estimator. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1524–1533, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018b. PMLR. URL http://proceedings.mlr.press/v80/foerster18a.html.

Fu, M. C. Gradient estimation. *Handbooks in operations research and management science*, 13:575–616, 2006.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. K. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *CoRR*, abs/1711.00123, 2017.

Greensmith, E., Bartlett, P. L., and Baxter, J. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5: 1471–1530, 2004.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Paisley, J., Blei, D., and Jordan, M. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pp. 3528–3536, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Tucker, G., Mnih, A., Maddison, C. J., Lawson, J., and Sohl-Dickstein, J. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pp. 2627–2636, 2017.

Tucker, G., Bhupatiraju, S., Gu, S., Turner, R. E., Ghahramani, Z., and Levine, S. The mirage of action-dependent baselines in reinforcement learning. *arXiv preprint arXiv:1802.10031*, 2018.

Wright, S. and Nocedal, J. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.