Improving Gradient Directions for Episodic Policy Search

Verbesserung von Gradienten-Richtungen für episodische Policy-Suche
Master thesis by Pengfei Zhao
Date of submission: 20.11.2021

1. Review: João Carvalho, M.Sc.
2. Review: Prof. Dr. Jan Peters
Darmstadt

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Erklärung zur Abschlussarbeit gemäß §22 Abs. 7 und §23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Pengfei Zhao, die vorliegende Masterarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Fall eines Plagiats (§38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung gemäß §23 Abs. 7 APB überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 20.11.2021

Pengfei Zhao

Contents

Abstract

In this thesis, we analyze Monte-Carlo gradient estimators for episodic policy search. The widely-used score function estimator has a high variance and low computational complexity, whereas the lesser-known measure-valued derivative estimator shows low variance at the cost of linear scaling of computational complexity with parameter dimensions.

Therefore, we explore combinations of both estimators to obtain a gradient estimator with low variance and low computational complexity. We first investigate a convex combination of score function and measure-valued estimator at selected sub-dimensions and then examine partially correcting score function gradients from previous iterations with measure-valued gradient estimates. To obtain an initial understanding of both estimators, we first conduct experiments on black-box test functions and a simplified episodic task. We then compare approaches of combining estimators on the continuous control Mujoco tasks in terms of their average rewards, variances, and deviations from reference gradients.

Our experiments reveal comparable or improved average rewards for a convex combination compared to a score function baseline. The combination increases the variance of the overall estimator but decreases variance for a subset of chosen dimensions. However, the level of variance decrease is strongly affected by the parameters chosen for the combined estimator, such as the number of selected dimensions. Our results also demonstrate a significant impact of the dimension selection strategy and the combination weights on the combined estimators. We find that a convex combination of the measure-valued derivative can achieve high variance reduction at chosen dimensions. However, with growing number of chosen dimensions, the advantage of utilizing the measure-valued derivative diminishes.

Zusammenfassung

In dieser Arbeit analysieren wir Monte-Carlo-Gradientenschätzer für die episodische Policy-Suche. Der weit verbreitete Score-Funktion-Schätzer hat eine hohe Varianz und eine niedrige Rechenkomplexität, während der Measure-Valued-Derivative-Schätzer eine niedrige Varianz, aber eine lineare Skalierung der Rechenkomplexität mit der Parameterdimension aufweist.

Daher untersuchen wir Kombinationen beider Schätzer, um einen Gradientenschätzer mit geringer Varianz und geringer Rechenkomplexität zu erhalten. Wir untersuchen zunächst eine konvexe Kombination aus Score-Funktion- und Measure-Valued-Derivative-Schätzer bei ausgewählten Dimensionen und untersuchen dann die Teil-Korrektur von Score-Funktions-Gradienten aus früheren Iterationen mit dem Measure-Valued-Derivative-Gradienten. Um ein erstes Verständnis für beide Schätzer zu erlangen, führen wir Experimente mit Black-Box-Testfunktionen und vereinfachten episodischen Aufgaben durch. Anschließend vergleichen wir die Ansätze zur Kombination der Schätzer auf den Mujoco-Aufgaben.

Unsere Experimente zeigen vergleichbare oder verbesserte Rewards für eine konvexe Kombination im Vergleich zu einem Score-Funktion-Schätzer. Die Kombination erhöht die Varianz des Gesamtschätzers, verringert aber die Varianz für eine Teilmenge der ausgewählten Dimensionen. Das Ausmaß der Varianzverringerung wird jedoch stark von den Parametern für den kombinierten Schätzer beeinflusst, wie etwa die Anzahl der ausgewählten Dimensionen. Unsere Ergebnisse zeigen auch einen signifikanten Einfluss der Dimensionsauswahlstrategie und der Kombinationsgewichte auf die kombinierten Schätzer.

Wir stellen fest, dass der Measure-Valued-Derivative-Schätzer eine hohe Varianzreduktion erreicht. Mit zunehmenden Dimensionen verringert sich jedoch der Vorteil der Verwendung des Measure-Valued Derivative-Schätzer. Da die Berechnungskomplexität mit den Parameterdimensionen skaliert, erzielt ein Score-Funktion-Schätzer mit der gleichen Komplexität oft vergleichbare Ergebnisse.

Preface

List of Figures

List of Tables

Notation

SF	Score function
MVD	Measure valued derivative
$\boldsymbol{\eta}_N^{SF}$	A score function estimator using N samples.
$\boldsymbol{\eta}_M^{MVD}$	A measure valued derivative estimator using M samples.
$\hat{\boldsymbol{g}}^{SF}$	Single score function gradient estimate or sample.
$\hat{\boldsymbol{g}}^{MVD}$	Single measure valued derivative estimate or sample.
x	Plain script denotes scalars
$\boldsymbol{x}$	Bold script denotes vectors.
$\boldsymbol{X}$	Uppercase bold script denotes matrices.
$\boldsymbol{I}_n$	Identity matrix of dimension n
$\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$	Nabla operator describes $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \left[\frac{\partial f(\boldsymbol{x})}{\partial x_i} ... \frac{\partial f(\boldsymbol{x})}{\partial x_N}\right]$
$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$	Describes the scalar product between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$.
$p(\boldsymbol{x}; \boldsymbol{\omega})$	Probability distribution of random variable $\boldsymbol{x}$ parametrized by $\boldsymbol{\omega}$.
$\boldsymbol{x} \sim p(\boldsymbol{x}; \boldsymbol{\omega})$	$\boldsymbol{x}$ is sampled according to $p(\boldsymbol{x}; \boldsymbol{\omega})$.
$\mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\omega})}[f(\boldsymbol{x})]$	Expected value of $f(\boldsymbol{x})$, where x is distributed by $p(\boldsymbol{x}; \boldsymbol{\omega})$.
$\mathbb{V}_{p(\boldsymbol{x};\boldsymbol{\omega})}[f(\boldsymbol{x})]$	Variance of $f(\boldsymbol{x})$, where $\boldsymbol{x}$ is distributed by $p(\boldsymbol{x}; \boldsymbol{\omega})$.
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Normal distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.

1. Introduction

In Monte-Carlo gradient estimation, we are concerned with finding the gradient of the distributional parameters of an expectation. This seemingly specific problem is ubiquitous in applications ranging from operational research [21, 40] to derivative pricing [15, 10] and generative modelling [20].

Another field where Monte-Carlo gradient estimation plays an important role is reinforcement learning. In particular, the score function estimator, which was originally introduced through the REINFORCE algorithm [44], has been widely used for policy search and actor-critic methods. However, the score function estimator is known to show high variances without a proper variance reduction method [17]. This high variance constitutes the biggest drawback for the score function since less precise gradients lead to slower convergence. Therefore, reducing the variance of gradient estimators translates directly to an improvement in many reinforcement learning methods.

At the same time, episodic policy search methods have achieved remarkable success. Algorithms like evolutionary strategies [35] use the score function estimator to optimize policy parameters in a black-box optimization setting. Evolutionary strategies combine the score function with additional heuristics to stabilize the algorithm and have shown performance on par with more complex policy gradient methods. With the additional advantage of large-scale parallelization, reduced variance for long-horizon tasks, and without the requirement to backpropagate a gradient, algorithms in the class of evolutionary strategies offer an attractive alternative to other reinforcement learning methods.

One viable way to improve such algorithms is to use a gradient estimator with less variance. A lesser-known gradient estimator using the measure-valued derivative (MVD) [30] provides such an alternative. Despite its low variance properties, the MVD estimator has not enjoyed wide attention in the reinforcement learning community. For this reason, we will explore in this thesis ways of applying the MVD to improve gradient estimation for the setting of episodic policy search.

The thesis is outlined as follows. In the remainder of this chapter, we will define the problem and introduce related works. We explain the three most widely used gradient estimators and their advantages and disadvantages in Chapter 2. Chapter 3 introduces the setting of episodic policy search and how the introduced estimators can be applied in this setting. Chapter 4 details the approaches we will use to extend the score function estimator with the MVD. In Chapter 5, we conduct experiments analyzing the different estimators and the extensions utilizing the MVD. Finally, Chapter 6 discusses the potential use cases for the MVD in episodic policy search and gives an outlook on possible future research directions.

1.1. Problem Statement

Our goal for this thesis is to improve gradient estimation for the objective below.

$$J(\boldsymbol{\omega}) = \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f(\boldsymbol{\theta})\right] = \int_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) p(\boldsymbol{\theta};\boldsymbol{\omega}) \,\mathrm{d}\boldsymbol{\theta}, \tag{1.1}$$

where we aim to find the parameters $\boldsymbol{\omega}$ of distribution $p(\boldsymbol{\theta};\boldsymbol{\omega})$ that maximizes the objective $J(\boldsymbol{\omega})$.

$$\arg\max_{\boldsymbol{\omega}} J(\boldsymbol{\omega})$$

Here $f(\boldsymbol{\theta})$ is assumed to be an unknown function, i.e. we are dealing with a black-box optimization problem. We can solve this problem via Gradient Ascent [28]. For this purpose an estimate of the gradient is needed. By assuming interchangeability of integration and derivation, we can rewrite (1.1). For the validity of the assumption, we refer to [27].

$$\nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega}) = \nabla_{\boldsymbol{\omega}} \int_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) p(\boldsymbol{\theta};\boldsymbol{\omega}) \,\mathrm{d}\boldsymbol{\theta} = \int_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \nabla_{\boldsymbol{\omega}} p(\boldsymbol{\theta};\boldsymbol{\omega}) \,\mathrm{d}\boldsymbol{\theta} \tag{1.2}$$

The integral in (1.2) is in most cases intractable and cannot be solved analytically. Therefore, we have to estimate the gradient with Monte Carlo estimation. However, $\nabla_{\boldsymbol{\omega}} p(\boldsymbol{\theta};\boldsymbol{\omega})$ is not guaranteed to be a probability distribution itself. This means that we cannot rewrite equation (1.2) as an expectation, which is a requirement for Monte Carlo estimation. This thesis will investigate two gradient estimators, which circumvent this problem and explore strategies to combine both estimators.

1.2. Related Works

Gradient Estimation. The topic of Monte-Carlo gradient estimation spans a wide array of research areas and extensive reviews have been provided in [13, 27]. More recently, variance reduction for gradient estimation has included learned components. Grathwohl et al. construct a neural control variate for the score function estimator [16]. Ren et al. learn a parameterized distribution for antithetic sampling [31]. In contrast, our approach focus on variance reduction through the MVD estimator. Similar to our approach, Parmas et al. combine the pathwise with the score function estimator for model-based policy search [29]. We instead examine a combination between the MVD and score function estimator for the model-free episodic policy search setting.

Measure-Valued Derivatives. Measure-valued derivatives have been first introduced by Pflug [30] with subsequent development of MVD for Markov processes by Heidergott et al. [26]. In machine learning, MVD has been explored for approximate Bayesian inference. In [34], Rosca et al. demonstrate the superior variance characteristics of the MVD over the score function in the use case of Bayesian logistic regression. Furthermore, Buesing et al. compare the MVD to the finite-difference estimator and the score function estimator [5]. In reinforcement learning, MVD has been studied in the context of step-based policy gradient methods [2]. Carvalho et al. give an extensive analysis of the MVD in the setting of actor-critic policy gradient algorithms [6].

Episodic Policy Search. Policy search encompasses a large family of algorithms, and a comprehensive survey is given in [9]. In our work, we set the focus on gradient-based episodic policy search methods. One such method is PEPG, which uses a score function gradient for policy optimization [36]. Similarly, Wiestra et al. [43] extend the score function gradient with the natural gradient, where the distance metric is with respect to the distribution instead of the parameters. Recently, policy search algorithms relying on the score function such as ES [35] and ARS [25] have shown remarkable success and sparked a wide range of extensions. For example, variance reduction for ES has been explored by Choromanski et al. through the use of orthogonal search directions [8]. Tang et al. propose a novel control variate specifically for ES [38]. Other approaches identify relevant sub-spaces of the parameters to reduce the dimensionality of the problem [7, 37, 24]. Our approach also aims to identify relevant sub-dimensions while reducing variance along those dimensions using the MVD gradient estimator.

2. Gradient Estimation

The theory of Monte-Carlo estimation states that one can approximate the integral

$$\mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\omega})}\left[f\left(\boldsymbol{x}\right)\right] = \int f(\boldsymbol{x}) p\left(\boldsymbol{x};\boldsymbol{\omega}\right) d\boldsymbol{x}$$

as

$$\frac{1}{N}\sum_{i=1}^{N} f\left(\boldsymbol{x}_i\right) \quad \boldsymbol{x} \sim p\left(\boldsymbol{x};\boldsymbol{\omega}\right).$$

To maximize the objective $J\left(\boldsymbol{\omega}\right)$ in equation 1.1 with an estimation of the gradient $\nabla_{\boldsymbol{\omega}} J\left(\boldsymbol{\omega}\right)$, we need to reformulate the following equation as an expectation like above

$$\nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega}) = \int_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \nabla_{\boldsymbol{\omega}} p(\boldsymbol{\theta};\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{\theta}.$$

Like has been mentioned in section 1.1, a straightforward reformulation is not possible. The estimators introduced in this chapter aim to rewrite gradient $\nabla_{\boldsymbol{\omega}} J\left(\boldsymbol{\omega}\right)$ as an expectation and give rise to unbiased Monte-Carlo estimators of the gradient. We describe such an estimator as

$$\begin{aligned}
&\boldsymbol{\eta}_N(\boldsymbol{\omega}) \approx \nabla_{\boldsymbol{\omega}} J\left(\boldsymbol{\omega}\right) \\
&\boldsymbol{\eta}_N(\boldsymbol{\omega}) = \frac{1}{N}\sum_{i=1}^{N} \hat{\boldsymbol{g}}\left(\boldsymbol{\theta}_i^*\right) \quad \boldsymbol{\theta}^* \sim p^*\left(\boldsymbol{\theta}^*;\boldsymbol{\omega}^*\right).
\end{aligned}$$

It is noted that through the reformulation, the distribution $p^*\left(\boldsymbol{\theta}^*;\boldsymbol{\omega}^*\right)$ can be in general different from $p(\boldsymbol{\theta};\boldsymbol{\omega})$ as is the case for the pathwise estimator or the measure-valued derivative estimator. Here $\boldsymbol{\eta}_N(\boldsymbol{\omega})$ is a gradient estimator that uses N Monte-Carlo samples of the gradient estimate $\hat{\boldsymbol{g}}$. Each sample itself can be used as a valid Monte-Carlo estimator, as $\hat{\boldsymbol{g}} = \boldsymbol{\eta}_{N=1}$. However, in chapter 4 where we make use of the empirical variance of the Monte-Carlo samples, this distinction between samples $\hat{\boldsymbol{g}}$ and estimator $\boldsymbol{\eta}_N$ proves to be very useful. In general, we desire three properties for Monte-Carlo estimators

Unbiasedness. If the gradient estimator is unbiased, then in expectation, the estimated gradient will be the same as the true gradient

$$\mathbb{E}\left[\boldsymbol{\eta}_N(\boldsymbol{\omega})\right] = \nabla_{\boldsymbol{\omega}} J\left(\boldsymbol{\omega}\right).$$

An unbiased gradient estimator allows convergence guarantees to be made [32]. However, as we will see in chapter 5 under certain assumptions, an unbiased estimator can also yield good convergence properties.

Low Variance. Having a low variance gradient estimator, in general, gives us faster convergence as the gradients are more accurate and the learning rate can be set to higher values. In principle, one can always use more samples to reduce variance as the variance decreases as $\mathcal{O}\left(1/N\right)$ with the number of samples N. However, each introduced gradient estimator will have its intrinsic variance properties along with its variance reduction techniques. Making use of these intrinsic variance properties and combining them to reduce variance will be the core theme of this thesis.

Low computational cost. As has been noted above, we can always get more accurate estimators by using more samples. However, each of the introduced estimators requires a different number of samples to yield the same variance. On top, each sample may require a different number of evaluations of the black-box function $f\left(\boldsymbol{\theta}\right)$. The number of function evaluations especially is an important metric, as in the policy search case, each evaluation equates to an expensive rollout of the robotic system or simulation environment. Therefore we focus on the number function evaluations instead of number of samples and prefer an estimator that gives lower variance with the same number of evaluations.

This chapter introduces three Monte-Carlo gradient estimators: the pathwise estimator, the score function estimator, and the measure-valued derivative estimator. Each of them is unbiased, but they differ in their variance and computational aspects and their application domain.

2.1. Pathwise Estimator

The pathwise estimator also known as the reparametrization trick or push-in estimator makes use of a known sampling path $h(\epsilon, \boldsymbol{\omega})$ of the original distribution to move the

distributional parameters $\boldsymbol{\omega}$ into the cost function. For an univariate Gaussian $\mathcal{N}(\theta; \mu, \sigma^2)$, this sampling path could for example be $h(\epsilon, \boldsymbol{\omega}) = \mu + \sigma\epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$. Using the sampling path, the base distribution $p(\epsilon)$ becomes independent of distributional parameters $\boldsymbol{\omega}$ and we can propagate the gradient straight through the cost function:

$$\begin{aligned}\nabla_{\boldsymbol{\omega}} \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f(\boldsymbol{\theta})\right] &= \nabla_{\boldsymbol{\omega}} \int p(\boldsymbol{\theta};\boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} \\ &= \nabla_{\boldsymbol{\omega}} \int p(\boldsymbol{\epsilon}) f(h(\boldsymbol{\epsilon}, \boldsymbol{\omega})) d\boldsymbol{\epsilon}.\end{aligned}$$

The pathwise estimator is therefore given by

$$\boldsymbol{\eta}_N(\boldsymbol{\omega}) = \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\omega}} f(h(\boldsymbol{\epsilon}_i, \boldsymbol{\omega})) \quad \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}).$$

2.1.1. Variance

In general the pathwise estimator has favourable variance characteristics [27]. However, in certain use cases, this estimator can exhibit large variances. This is for example in the case of noisy cost evaluations [6] or when samples have to be tracked over a long path until the cost evaluation [29].

2.1.2. Computational Aspects

Another important advantage is the low cost of computation. In theory, each sample of the estimator only requires $\mathcal{O}(1)$ function evaluation. Despite its advantages, the pathwise estimator is limited in its application domain. First of all, a sampling path needs to be known a priori, secondly it is restricted to continuous distributions, even though relaxations exist in the form of the Gumbel-Softmax [19]. Lastly and most importantly, the pathwise estimator assumes differentiability of the cost function $f(\boldsymbol{\theta})$. In black-box optimization, however, the cost function is assumed to be unknown and the gradient inaccessible. For this reason, the pathwise estimator is not suited for our concern. Nevertheless, in model-based approaches, the pathwise estimator has been used successfully for policy search [29, 22].

2.2. Score Function

The score function (SF) in statistics is the derivative of the log-likelihood function with respect to the distributional parameters and finds usage for example in maximum likelihood estimation. We can derive a Monte-Carlo gradient estimator using the following identity of the SF

$$\nabla_{\boldsymbol{\omega}} p(\boldsymbol{\theta};\boldsymbol{\omega}) = \nabla_{\boldsymbol{\omega}} \log p(\boldsymbol{\theta};\boldsymbol{\omega}) p(\boldsymbol{\theta};\boldsymbol{\omega}).$$

Using the above identity, we can formulate the expectation as

$$\begin{aligned}
\nabla_{\boldsymbol{\omega}} \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f(\boldsymbol{\theta})\right] &= \nabla_{\boldsymbol{\omega}} \int p(\boldsymbol{\theta};\boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} \\
&= \int \nabla_{\boldsymbol{\omega}} p(\boldsymbol{\theta};\boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} \\
&= \int p(\boldsymbol{\theta};\boldsymbol{\omega}) \nabla_{\boldsymbol{\omega}} \log p(\boldsymbol{\theta};\boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta}.
\end{aligned}$$

The SF estimator is then given through

$$\boldsymbol{\eta}_N(\boldsymbol{\omega}) = \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\omega}} \log p(\boldsymbol{\theta}_i;\boldsymbol{\omega}) f(\boldsymbol{\theta}_i) \quad \boldsymbol{\theta} \sim p(\boldsymbol{\theta};\boldsymbol{\omega}).$$

2.2.1. Variance

In general the SF estimator exhibits the largest variances among the three estimators. The variance is dependent on the form of the cost function and the dimensionality of the parameters [27]. For this reason variance reduction methods are crucial for the SF estimator. In the following we will introduce two of them.

Baseline We can reduce the variance of the SF estimator by subtracting a baseline b:

$$\nabla_{\boldsymbol{\omega}} \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f\left(\boldsymbol{\theta}\right) - b\right] = \int p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right) \nabla_{\boldsymbol{\omega}} \log p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right) \left(f\left(\boldsymbol{\theta}\right) - b\right) d\boldsymbol{\theta}$$

Subtracting the baseline does not change the unbiasedness property of the estimator as

$$\int p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right)\nabla_{\boldsymbol{\omega}}\log\ p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right)bd\boldsymbol{\theta}=b\int\nabla_{\boldsymbol{\omega}}p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right)d\boldsymbol{\theta}=b\nabla_{\boldsymbol{\omega}}\int p\left(\boldsymbol{\theta};\boldsymbol{\omega}\right)d\boldsymbol{\theta}=0.$$

Here the log-likelihood identity is used again as $\nabla_{\boldsymbol{\omega}}\log p(\boldsymbol{\theta};\boldsymbol{\omega})p(\boldsymbol{\theta};\boldsymbol{\omega})=\nabla_{\boldsymbol{\omega}}p(\boldsymbol{\theta};\boldsymbol{\omega})$ for the above reformulation. This gives us the SF estimator with baseline variance reduction as

$$\boldsymbol{\eta}_N(\boldsymbol{\omega})=\frac{1}{N}\sum_{i=1}^{N}\nabla_{\boldsymbol{\omega}}\log p(\boldsymbol{\theta}_i;\boldsymbol{\omega})\left(f(\boldsymbol{\theta})-b\right)\quad\boldsymbol{\theta}\sim p(\boldsymbol{\theta};\boldsymbol{\omega}).$$

In practice, often a moving average of the cost function evaluations is used as the baseline. However, for certain classes of algorithms optimal baselines exist. In our case, using the score function for episodic policy search is equal to the PGPE algorithm [36]. For the PGPE, the optimal baseline [9] can be formulated as

$$\boldsymbol{b}=\frac{\sum_{i=1}^{N}(\nabla_{\boldsymbol{\omega}}\log p(\boldsymbol{\theta}_i;\boldsymbol{\omega}))^2 f(\boldsymbol{\theta}_i)}{\sum_{i=1}^{N}(\nabla_{\boldsymbol{\omega}}\log p(\boldsymbol{\theta}_i;\boldsymbol{\omega}))^2}.$$

Antithetic Sampling The idea of antithetic sampling is to draw negatively correlated samples for the Monte Carlo estimation instead of independent samples [14]. Intuitively if $f(\boldsymbol{x})$ is high for a sample $\boldsymbol{x}$, then for its antithetic sample $\tilde{\boldsymbol{x}}$, $f(\tilde{\boldsymbol{x}})$ should yield a low value. A Monte-Carlo estimator with antithetic sampling is then given by

$$\frac{1}{2N}\sum_{i=1}^{N}\frac{f(\boldsymbol{x}_i)+f(\tilde{\boldsymbol{x}}_i)}{2}\tag{2.1}$$

Through the negative correlation structure, we can achieve error cancellation and variance reduction. However, ensuring negative correlation is difficult, since it depends on the cost function and the underlying distribution [31]. Several works have focused on learning such an antithetic sampler instead [31, 45]. Following Ren et al [31], for the simple case of a zero-mean Gaussian $\mathcal{N}\left(\boldsymbol{x};\mathbf{0},\boldsymbol{\Sigma}\right)$, antithetic variates can be given as $\tilde{\boldsymbol{x}}=-\boldsymbol{x}$ and $\boldsymbol{x}$. This fact together with a reparametrization of the Gaussian is used for the antithetic sampling in Evolutionary Strategies [35] that we will introduce in section 3.1.

2.2.2. Computational Aspects

Like the pathwise estimator, the SF estimator only require $\mathcal{O}(1)$ function evaluations for each sample. However in practice due to its high variance, a higher number of samples is

required to get a comparable variance to the other estimators. The score function estimator is more broadly applicable than the pathwise estimator. It can be used for continuous as well as discrete distributions. Moreover, the cost functions can be non-smooth or non-differentiable, which allows for its use in black-box optimization

2.3. Measure-Valued Derivative

We can also compute an unbiased gradient estimate using the measure-valued derivative [30]. The estimator is based on the Hahn-Jordan decomposition that allows us decompose the derivative $\nabla_{\omega_d} p(\boldsymbol{\theta}; \boldsymbol{\omega})$ into a difference of two distributions scaled by a constant factor.

$$\nabla_{\omega_d} p(\boldsymbol{\theta}; \boldsymbol{\omega}) = c_{\omega_d}(p^+(\boldsymbol{\theta}; \boldsymbol{\omega}) - p^-(\boldsymbol{\theta}; \boldsymbol{\omega}))$$

Here p^+ and p^- are the positive and negative distributions for the MVD decomposition. Together with the constant factor c_{ω_d} they form a triplet (c_{ω_d}, p^+, p^-) that is different for each distribution and parameter. The MVD triplets for commonly encountered distributions are listed in table 2.1. Using the above decomposition the gradient can be formulated as

$$\begin{aligned}
\nabla_{\omega_d} \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f(\boldsymbol{\theta})\right] &= \nabla_{\omega_d} \int p(\boldsymbol{\theta}; \boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} && (2.2)\\
&= \int \nabla_{\omega_d} p(\boldsymbol{\theta}; \boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} && (2.3)\\
&= c_{\omega_d} \left(\int p^+(\boldsymbol{\theta}; \boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} - \int p^-(\boldsymbol{\theta}; \boldsymbol{\omega}) f(\boldsymbol{\theta}) d\boldsymbol{\theta} \right). && (2.4)
\end{aligned}$$

Now the MVD estimator can be written as

$$\boldsymbol{\eta}_N(\omega_d) = \frac{c_{\omega_d}}{N} \left(\sum_{i=1}^{N} f(\boldsymbol{\theta}_i^+) - \sum_{i=1}^{N} f(\boldsymbol{\theta}_i^-) \right); \quad \boldsymbol{\theta}^+ \sim p^+(\boldsymbol{\theta}; \boldsymbol{\omega}), \quad \boldsymbol{\theta}^- \sim p^-(\boldsymbol{\theta}; \boldsymbol{\omega})$$

Note here that the derivative is only computed with respect to a single parameter dimension d. As a result, estimating the gradient involves computing the MVD along each dimension.

2.3.1. Variance

The MVD estimator in general shows low variance as compared to the SF estimator. The variance of the MVD is given in [27] as:

$$\mathbb{V}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[\hat{\boldsymbol{g}}\right] = \mathbb{V}_{p^+(\boldsymbol{\theta}^+;\boldsymbol{\omega})}\left[f(\boldsymbol{\theta}^+)\right] + \mathbb{V}_{p^-(\boldsymbol{\theta}^-;\boldsymbol{\omega})}\left[f(\boldsymbol{\theta}^-)\right] - 2\mathrm{Cov}_{p^+(\boldsymbol{\theta}^+;\boldsymbol{\omega})p^-(\boldsymbol{\theta}^-;\boldsymbol{\omega})}[f(\boldsymbol{\theta}^+)f(\boldsymbol{\theta}^-)],$$

where $\hat{\boldsymbol{g}}$ denotes single samples of the MVD estimator. From the above expression we can observe that we can reduce the variance when $f(\boldsymbol{\theta}^+)$ and $f(\boldsymbol{\theta}^-)$ are positively correlated. The variance reduction method coupling makes use of this fact and achieves variance reduction through reusing common random numbers. In practice, it means we generate the positive sample $\boldsymbol{\theta}^+$ via a transformation of the negative sample $\boldsymbol{\theta}^-$ or vice versa. Unless otherwise stated, we will always use the MVD in combination with coupling.

2.3.2. Computational Aspects

For a single sample of the MVD estimator $\mathcal{O}(2D)$ function evaluations are required, where D specifies the dimensionality of the parameters $\boldsymbol{\theta}$. As such it scales linearly with the number of dimensions. Due to the low variance of the MVD however, it can be observed that the SF estimator requires a similar number of function evaluations as the MVD to yield comparable variance levels [6].

To compute the MVD for multivariate distributions, we consider a fully factorized distribution $p(\boldsymbol{\theta};\boldsymbol{\omega}) = \prod_{i=1}^{D} p(\theta_i;\omega_i)$. In this case, positive samples for the d-th dimension can be constructed by sampling $\boldsymbol{\theta}^+$ from $p^+(\boldsymbol{\theta}^+;\boldsymbol{\omega}) = p^+(\theta_d^+;\omega_d)\prod_{k=1,k\neq d}^{D} p(\theta_k;\omega_k)$. In practice, this means we can simply sample from the original multivariate distribution $p(\boldsymbol{\theta};\boldsymbol{\omega})$ and replace the d-th dimension with samples from the positive part of the univariate decomposition. The same goes for the negative samples.

The MVD can be used for any type of cost function and allows for discrete and continuous distributions alike [27]. Like the SF estimator, the MVD estimator does not require a differentiable cost function. For this reason, both make the only two viable estimators of the three introduced that can be applied for black-box optimization.

Distribution $p(\theta;\omega)$	Constant c_ω	Positive distribution p^+	Negative distribution p p^-
Bernoulli(ω)	1	δ_1	δ_0
Poisson(ω)	1	$\mathcal{P}(\omega)+1$	$\mathcal{P}(\omega)$
Normal(ω,σ^2)	$\frac{1}{\sigma\sqrt{2\pi}}$	$\omega+\sigma\mathcal{W}(2,0.5)$	$\omega-\sigma\mathcal{W}(2,0.5)$
Normal(μ,ω^2)	$\frac{1}{\omega}$	$\mathcal{M}(\mu,\omega^2)$	$\mathcal{N}(\mu,\omega^2)$
Exponential(ω)	$\frac{1}{\omega}$	$\mathcal{E}(\omega)$	$\omega^{-1}\mathcal{E}r(2)$
Gamma(a,ω)	$\frac{a}{\omega}$	$\mathcal{G}(a,\omega)$	$\mathcal{G}(a+1,\omega)$
Weibull(α,ω)	$1/\omega$	$\mathcal{W}(\alpha,\omega)$	$\mathcal{G}(2,\omega)^{1/\alpha}$

Table 2.1.: MVD Triplets (c_ω, p^+, p^-) for common distributions. The derivative is computed with respect to the distributional parameters ω. Expressions for the distributions are provided in table 2.2.

Name	Domain	Notation	Probability Density / Mass Functions
Gaussian	$\mathbb{R}$	$\mathcal{N}(x;\mu,\sigma^2)$	$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)\right)^2$
Double-sided Maxwell	$\mathbb{R}$	$\mathcal{M}(x;\mu,\sigma^2)$	$\frac{1}{\sqrt{2\pi\sigma^3}}(x-\mu)^2\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
Weibull	$\mathbb{R}^+$	$\mathcal{W}(x;\alpha,\beta,\mu)$	$\alpha\beta(x-\mu)^{\alpha-1}\exp\left(-\beta(x-\mu)^\alpha\right)\mathbb{1}_{\{x\geq 0\}}$
Poisson	$\mathbb{N}_0$	$\mathcal{P}(x;\theta)$	$\exp(-\theta)\sum_{j=0}^{\infty}\frac{\theta^j}{j!}\delta_j$
Erlang	$\mathbb{R}^+$	$\mathcal{E}r(x;\theta,\lambda)$	$\frac{\lambda^\theta x^{\theta-1}\exp(-\lambda x)}{(\theta-1)!}$
Gamma	$\mathbb{R}^+$	$\mathcal{G}(x;\alpha,\beta)$	$\frac{\beta^\alpha}{\Gamma(\alpha)}x^{\alpha-1}\exp(-x\beta)\mathbb{1}_{\{x\geq 0\}}$
Exponential	$\mathbb{R}^+$	$\mathcal{E}(x;\gamma)$	$\mathcal{G}(1,\lambda)$

Table 2.2.: Table of commonly encountered distributions along with their densities or mass functions.

3. Episodic Policy Search

In policy search, we aim to change the parameters of a policy such that a reinforcement learning agent using the policy optimizes its objective. Within policy search algorithms, several distinctions can be made. One can utilize a model of the environment dynamics to sample trajectories, which results in the class of model-based approaches. This class of algorithms typically benefits from reduced sample complexity. Model-free methods, on the other hand, learn policy parameters without an environment model. Such approaches can be helpful when a model is unknown or difficult to estimate [33]. A further distinction can be made in terms of how the agent explores the environment. One option is to utilize a stochastic policy and sample actions at each time step. However, such an approach typically leads to jerky trajectories and exhibits high variance for long-horizon tasks [9, 41]. An alternative is episodic policy search, where we sample policy parameters from an upper-level policy at the start of the episode. This option leads to exploration in the parameter space. It has the advantage that we can obtain smooth trajectories, and its variance is insensitive to the task horizon [35, 41].

This chapter will consider the model-free episodic case, which can be alternatively regarded as a black-box optimization problem. Here the black-box is an unknown environment modeled as $f(\boldsymbol{\theta})$. It returns the total accumulated rewards given $\boldsymbol{\theta}$ that parametrizes a lower-level deterministic policy $\pi_{\boldsymbol{\theta}}$ and is sampled from an upper-level policy distribution $p(\boldsymbol{\theta};\boldsymbol{\omega})$. Our goal is then to optimize with respect to the distributional parameters $\boldsymbol{\omega}$. This problem set-up is described by the equation

$$J(\boldsymbol{\omega}) = \mathbb{E}_{p(\boldsymbol{\theta};\boldsymbol{\omega})}\left[f(\boldsymbol{\theta})\right].$$

In order to maximize the above objective, an approach is to estimate the gradient as $\boldsymbol{\eta}_N(\boldsymbol{\omega}) \approx \nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega})$ via Monte-Carlo. In essence, most algorithms in this class can be summarized through the pattern given in algorithm 1.

Algorithm 1: Episodic Policy Search Algorithm Pattern

Hyperparameters : learning rate α, Monte-Carlo samples N
Input : initial parameters $\boldsymbol{\omega}_0$, distribution $p(\boldsymbol{\theta}; \boldsymbol{\omega})$, policy $\pi_{\boldsymbol{\theta}}$, black-box function $f(\boldsymbol{\theta})$

1 **for** *i in* $0, 1 \cdots$ **do**
2 $\quad \boldsymbol{\eta}_N(\boldsymbol{\omega}_i) \leftarrow$ `monte-carlo-gradient(`$p(\boldsymbol{\theta}; \boldsymbol{\omega}_i), f(\boldsymbol{\theta}), \pi_{\boldsymbol{\theta}}, N$`)`
3 $\quad \boldsymbol{\omega}_{i+1} \leftarrow \boldsymbol{\omega}_i + \alpha \boldsymbol{\eta}_N(\boldsymbol{\omega}_i)$

Most episodic policy search algorithm like Parameter Exploring Policy Gradients (PEPG) [36] or Evolutionary Strategies (ES) [35] follow the basic pattern 1. While they distinguish themselves in components such as the upper-level or lower-level policy, they all rely on the SF for gradient estimation in line 2. Nevertheless, the additional heuristics for state and reward normalization can have a dramatic impact on the performance of the algorithm like in the case of Augmented Random Search (ARS) [25].

In this chapter, we introduce three algorithms for episodic policy search based on the SF estimator: PEPG, ES and ARS. Since they all share the basic pattern for parameter optimization given in algorithm 1, we will only introduce their difference in gradient estimation and the additional heuristics used. Finally, we will also replace gradient estimation in pattern 1 with a MVD estimator, arriving at an episodic search algorithm using the MVD.

Following ES and ARS, we will also assume the upper-level policy distribution to be a Gaussian with isotropic covariance $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$, where D denotes the parameter dimensions. However, the derivations can be easily transferred to the case of a Gaussian with diagonal covariance. In order to avoid overly verbose pseudocode and to stay close to the experimental set-up in chapter 5, the code provided in this chapter will only estimate gradient with respect to the mean vector $\boldsymbol{\mu}$. Pseudocode for the estimation of the gradient with respect to the standard deviation follow equivalently.

3.1. Episodic Policy Search with SF

As stated, we assume the upper-level policy to be represented by a Gaussian with isotropic covariance $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$ with parameter dimensions D. In this case, we can arrive at

expression for the score function of d-th dimension of $\boldsymbol{\mu}$ through

$$\begin{aligned}\nabla_{\mu_d} \log(\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma I_D)) &= \nabla_{\mu_d} \log \prod_{d=0}^{D} \mathcal{N}(\theta_d; \mu_d, \sigma) \\ &= \nabla_{\mu_d} \sum_{d=0}^{D} \log \mathcal{N}(\theta_d; \mu_d, \sigma) \\ &= \frac{\theta_d - \mu_d}{\sigma^2}.\end{aligned}$$

Similarly, we obtain for the standard deviation σ

$$\nabla_\sigma \log \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma I_D) = \frac{(\theta_d - \mu_d)^2 - \sigma^2}{\sigma^3}.$$

Using the score function expressions, we arrive at gradient estimators for the parameters of the Gaussian

$$\begin{aligned}\boldsymbol{\eta}_N(\boldsymbol{\mu}) &= \frac{1}{N} \sum_{i=1}^{N} \frac{(\boldsymbol{\theta}_i - \boldsymbol{\mu})}{\sigma^2} f(\boldsymbol{\theta}_i) \quad \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma I_D) \\ \boldsymbol{\eta}_N(\sigma) &= \frac{1}{N} \sum_{i=1}^{N} \frac{(\boldsymbol{\theta}_i - \boldsymbol{\mu})^2 - \sigma^2)}{\sigma^3} f(\boldsymbol{\theta}_i) \quad \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma I_D).\end{aligned}$$

With a reparametrization of the Gaussian as $\boldsymbol{\theta} = \boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}_D, \boldsymbol{I}_D)$, we can get a new perspective on the gradient estimators:

$$\begin{aligned}\boldsymbol{\eta}_N(\boldsymbol{\mu}) &= \frac{1}{N} \sum_{i=1}^{N} \frac{\boldsymbol{\epsilon}_i}{\sigma} f(\boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}_i) \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon};, \mathbf{0}_D, \boldsymbol{I}_D) \\ \boldsymbol{\eta}_N(\sigma) &= \frac{1}{N} \sum_{i=1}^{N} \frac{(\boldsymbol{\epsilon}_i^2 - 1)}{\sigma} f(\boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}_i) \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}_D, \boldsymbol{I}_D).\end{aligned}$$

This new perspective tells us that, in fact, one can view the above equations as estimating the gradient with respect to a Gaussian-smoothed version of the objective [35, 8]. The smoothed version $f(\boldsymbol{\mu} + \sigma\boldsymbol{\epsilon})$ is differentiable, even if the original $f(\boldsymbol{\theta})$ is not [8]. Using the estimator described in algorithm 2 with pattern 1, gives us a basic episodic policy search algorithm using the SF estimator.

Algorithm 2: Basic SF Gradient Estimation

Hyperparameters : Monte-Carlo samples N

Input : initial parameters $\boldsymbol{\mu}_0$, σ_0; distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$; policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$

for *i in* $0, 1 \cdots N$ **do**

1 $\quad \boldsymbol{\theta}_i \leftarrow \boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}_i$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \boldsymbol{0}_D, \boldsymbol{I}_D)$

2 $\quad r_i \leftarrow$ evaluate $f(\boldsymbol{\theta}_i)$

end

3 $\boldsymbol{\eta}_N(\boldsymbol{\mu}) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \frac{\boldsymbol{\epsilon}_i}{\sigma} r_i$

4 **return** $\boldsymbol{\eta}_N(\boldsymbol{\mu})$

PEPG and ES. Algorithm 2 forms the basis for Evolutionary Strategies (ES) [35] and Parameter Exploring Policy Gradients (PEPG) [36]. Like in the shown algorithm, both approaches make use of the SF for gradient estimation. ES also models the upper-level policy as a Gaussian with isotropic covariance. PEPG, on the other hand, represents the upper-level policy as a Gaussian with diagonal covariance. While PEPG also optimizes the standard deviation, ES fixes the standard deviation as a hyperparameter. Overall, both ES and PEPG show close resemblance. Therefore we will introduce them both together.

As mentioned before, additional heuristics can have a major impact on the performance of black-box optimization algorithms. One class of such heuristics is to normalize the rewards. Since the rewards or evaluations of the black-box function directly scale the SF estimator, high magnitude rewards can lead to large variance of the gradient estimates. For this matter, the heuristic used in PEPG is to normalize rewards using the a priori known maximum reward or the maximal reward encountered so far. Another approach, followed by ES, is to transform rewards into ranks using a monotonously increasing function[35, 43].

One important extension to algorithm 2 introduced in PEPG and ES is that of antithetic sampling. Like mentioned in the previous chapter 2, antithetic sampling makes use of the negative correlation of two samples for variance reduction. For a zero-mean Gaussian $\mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{\Sigma})$ antithetic samples are given as $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}} = -\boldsymbol{x}$. Thus following the expression

2.1 for antithetic sampling, we obtain

$$\boldsymbol{\eta}_N(\boldsymbol{\mu}) = \frac{1}{N}\sum_{i=1}^{N} \frac{\boldsymbol{\epsilon}_i}{\sigma}\left(\frac{f(\boldsymbol{\mu}+\sigma\boldsymbol{\epsilon}_i) - f(\boldsymbol{\mu}-\sigma\boldsymbol{\epsilon}_i)}{2}\right) \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon};, \mathbf{0}_D, \boldsymbol{I}_D)$$

$$\boldsymbol{\eta}_N(\sigma) = \frac{1}{N}\sum_{i=1}^{N} \frac{(\boldsymbol{\epsilon}_i^2 - 1)}{\sigma}\frac{(f(\boldsymbol{\mu}+\sigma\boldsymbol{\epsilon}_i) - f(\boldsymbol{\mu}-\sigma\boldsymbol{\epsilon}_i))}{2} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon};, \mathbf{0}_D, \boldsymbol{I}_D).$$

Furthermore, PEPG also proposes to use a baseline for variance reduction. The antithetic SF estimator and the SF estimator with baseline are shown in algorithms 3 and 4.

Algorithm 3: SF Gradient Estimation with Antithetic Sampling

Hyperparameters : Monte-Carlo samples N
Input : initial parameters $\boldsymbol{\mu}_0$, σ_0, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{I}_D\boldsymbol{\sigma})$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$

for *i in* $1, 2 \cdots N$ **do**
1 $\quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}_D, \boldsymbol{I}_D)$
2 $\quad \boldsymbol{\theta}_i^+ \leftarrow \boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}_i$
3 $\quad \boldsymbol{\theta}_i^- \leftarrow \boldsymbol{\mu} - \sigma\boldsymbol{\epsilon}_i$
4 $\quad r_i^+ \leftarrow$ evaluate $f(\boldsymbol{\theta}_i^+)$
5 $\quad r_i^- \leftarrow$ evaluate $f(\boldsymbol{\theta}_i^-)$
end
6 $\boldsymbol{\eta}_N(\boldsymbol{\mu}) \leftarrow \frac{1}{2N}\sum_{i=1}^{N} \frac{\boldsymbol{\epsilon}_i}{\sigma}\frac{r_i^+ - r_i^-}{2}$
7 **return** $\boldsymbol{\eta}_N(\boldsymbol{\mu})$

Augmented Random Search. Like ES and PEPG, Augmented Random Search (ARS) [25] makes use of the antithetic SF for gradient estimation. However, one striking difference is the class of policies used. While ES models $\boldsymbol{\theta}$ as parameters of a large neural network, ARS uses $\boldsymbol{\theta}$ as a linear policy. Despite the simplicity of ARS, it achieves results that are state-of-the-art for black-box optimization.

Apart from the policy parametrization, the biggest difference between ARS and ES or PEPG is its heuristics for reward normalization and state normalization. In ARS, reward normalization is achieved by rescaling the rewards of an iteration with their standard deviation $r_i' = \frac{r_i}{\sigma_R} \quad i \in [1 \cdots N]$, where σ_R is the standard deviation of all rewards

Algorithm 4: SF Gradient Estimation with Baseline

Hyperparameters : Monte-Carlo samples N
Input : initial parameters $\boldsymbol{\mu}_0$, σ_0, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{I}_D\boldsymbol{\sigma})$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$

for *i in* $1, 2 \cdots N$ **do**
1 $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}_D, \boldsymbol{I}_D)$
2 $\quad \boldsymbol{\theta}_i \leftarrow \boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}_i$
3 $\quad r_i \leftarrow$ evaluate $f(\boldsymbol{\theta}_i)$
end
4 $\boldsymbol{\eta}_N(\boldsymbol{\mu}) \leftarrow \frac{1}{N}\sum_{i=1}^{N} \frac{\boldsymbol{\epsilon}_i}{\sigma}(r_i - b)$
5 $b \leftarrow$ update as moving average of $r_{i \cdots N}$ or according to optimal baseline given in section 2.2
6 **return** $\boldsymbol{\eta}_N(\boldsymbol{\mu})$

collected during one iteration. This approach is motivated by the observation that at high rewards, perturbations to the policy can cause large variations in rewards [25].

An additional heuristic used in ARS aims to normalize the states of the environment. Since the environment states can have arbitrary ranges, the states with large ranges result in larger action outputs in contrast to states with a small ranges. Therefore, normalization allows the linear policy to have equal influence on the environment states [25]. Practically this normalization is applied as:

$$\boldsymbol{x}_{normalized} = \mathrm{diag}(\boldsymbol{\Sigma}_{\boldsymbol{x}})^{-1/2}(\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{x}}),$$

where $\boldsymbol{x}$ denotes the environment states. The covariance $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ and mean $\boldsymbol{\mu}_{\boldsymbol{x}}$ of the environment states are computed as running statistics that are updated at each iteration. The modified antithetic SF estimator with the ARS heuristics is given in algorithm 5.

3.2. Episodic Policy Search with MVD

The gradient estimation in pattern 1 can be equally performed with the MVD. Let us recall from section 2.3 that the MVD specifies the derivative of a distribution using a triplet $(c_{\omega_d}, p^+(\boldsymbol{\theta}; \boldsymbol{\omega}), p^-(\boldsymbol{\theta}; \boldsymbol{\omega}))$ as

$$\nabla_{\omega_d} p(\boldsymbol{\theta}; \boldsymbol{\omega}) = c_{\omega_d}(p^+(\boldsymbol{\theta}; \boldsymbol{\omega}) - p^-(\boldsymbol{\theta}; \boldsymbol{\omega}))$$

Algorithm 5: Antithetic SF Gradient Estimation with ARS Heuristics [25]

Hyperparameters : Monte-Carlo samples N

Input : initial parameters $\boldsymbol{\mu}_0$, σ_0, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$

for *i in* $1 \cdots N$ **do**

$\boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{\epsilon}; \boldsymbol{0}_D, \boldsymbol{I}_D)$

$\boldsymbol{\theta}_i^+ \leftarrow \boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}_i$

$\boldsymbol{\theta}_i^- \leftarrow \boldsymbol{\mu} - \sigma \boldsymbol{\epsilon}_i$

$r_i^+ \leftarrow$ collect rewards $f(\boldsymbol{\theta}_i^+)$ using policy $\pi(\boldsymbol{x}) = \boldsymbol{\theta}^+ \operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{x}})^{-1/2}(\boldsymbol{x} - \mu_{\boldsymbol{x}})$

$r_i^- \leftarrow$ collect rewards $f(\boldsymbol{\theta}_i^-)$ using policy $\pi(\boldsymbol{x}) = \boldsymbol{\theta}^- \operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{x}})^{-1/2}(\boldsymbol{x} - \mu_{\boldsymbol{x}})$

end

$\sigma_R \leftarrow$ standard deviation of combined rewards from $r^+_{1\cdots N}$ and $r^-_{1,\cdots,N}$.

$\boldsymbol{\eta}_N(\boldsymbol{\mu}) \leftarrow \frac{1}{2N} \sum_{i=1}^N \frac{\boldsymbol{\epsilon}_i}{\sigma_R}(r_i^+ - r_i^-)$

$\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{x}} \leftarrow$ update mean and covariance using all the states encountered so far during training.

return $\boldsymbol{\eta}_N(\boldsymbol{\mu})$

As given in table 2.1, the triplet for the mean μ of an univariate Gaussian is given as $(1/\sigma\sqrt{2\pi}, \mu + \mathcal{W}(2, 0.5), \mu - \mathcal{W}(2, 0.5))$ and the Monte-Carlo estimator for μ is

$$\eta_N(\mu) = \frac{1}{N\sigma\sqrt{2\pi}} \sum_{i=1}^{N} f(\mu + \sigma\epsilon_i^+) - f(\mu - \sigma\epsilon_i^-) \quad , \epsilon^+ \sim \mathcal{W}(2, 0.5), \epsilon^- \sim \mathcal{W}(2, 0.5).$$

For the standard deviation σ, the triplet is $(1/\sigma, \mathcal{M}(\mu, \sigma^2), \mathcal{N}(\mu, \sigma^2))$ with the estimator been

$$\eta_N(\sigma) = \frac{1}{N\sigma} \sum_{i=1}^{N} f(\theta_i^+) - f(\theta_i^-) \quad , \theta^+ \sim \mathcal{M}(\theta; \mu, \sigma), \theta^- \sim \mathcal{N}(\theta; \mu, \sigma).$$

Note here that the gradient estimator for μ shows close resemblance to the SF estimator using antithetic sampling. As previously, we would like to estimate the gradient of the mean vector $\boldsymbol{\mu}$ for $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$. Like is described in section 2.3, for a fully factorized distribution $p(\boldsymbol{\theta}; \boldsymbol{\omega}) = \prod_{d=1}^D p(\theta_d; \omega_d)$, the positive component of the MVD for dimension d is given as

$$p_d^+(\boldsymbol{\theta}; \boldsymbol{\omega}) = p_d^+(\theta_d; \omega_d) \prod_{k \neq d}^{D} p(\theta_k; \omega_k),$$

with the same applying to the negative component. For estimating the gradient with respect to μ_d, a sample from the positive component $p^+(\cdot)$ is constructed by first sampling from $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$ and replacing the d-th component with a sample from the Weibull $\mathcal{W}(2, 0.5)$. We can now formulate gradient estimation using the MVD in algorithm 6, which in combination with pattern 1 forms a fully functional episodic policy search algorithm using MVD. Note that we can share samples ϵ in line 2 and 3 to use coupling as variance reduction.

Algorithm 6: MVD Gradient Estimation

Hyperparameters : Monte-Carlo samples N
Input : initial parameters $\boldsymbol{\mu}_0$, σ_0, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$

for *d in* $1 \cdots D$ **do**
  for *i in* $1 \cdots N$ **do**
1 $\boldsymbol{\theta}_i \sim \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$
2 $\theta_d^+ \leftarrow \mu_d + \sigma \epsilon_i^+$ with $\epsilon^+ \sim \mathcal{W}(\epsilon; 2, 0.5)$
  /* use ϵ^+ as ϵ^- for variance reduction with coupling */
3 $\theta_d^- \leftarrow \mu_d - \sigma \epsilon_i^-$ with $\epsilon^- \sim \mathcal{W}(\epsilon; 2, 0.5)$
4 $\boldsymbol{\theta}^+ \leftarrow [\theta_1, \cdots, \theta_d^+, \cdots, \theta_D]$
5 $\boldsymbol{\theta}^- \leftarrow [\theta_1, \cdots, \theta_d^-, \cdots, \theta_D]$
6 $r_i^+ \leftarrow$ evaluate $f(\boldsymbol{\theta}^+)$ using policy $\pi_{\boldsymbol{\theta}^+}$
7 $r_i^- \leftarrow$ evaluate $f(\boldsymbol{\theta}^-)$ using policy $\pi_{\boldsymbol{\theta}^-}$
  end
8 $\eta(\mu_d) \leftarrow \frac{1}{N\sigma\sqrt{2\pi}} \sum_{i=1}^{N} r_i^+ - r_i^-$
end
9 $\boldsymbol{\eta}_N(\boldsymbol{\mu}) \leftarrow [\eta(\mu_1), \eta(\mu_2), \cdots, \eta(\mu_D)]$
10 **return** $\boldsymbol{\eta}_N(\boldsymbol{\mu})$

4. Combining Estimators

The SF estimator generally shows high variance but requires only $\mathcal{O}(1)$ function evaluations per Monte-Carlo sample. At the same time, the MVD estimator has been shown to exhibit low variance, however requiring $\mathcal{O}(2D)$ evaluations, where D denotes the parameter dimensionality [27, 34]. Ideally, we would like to obtain a Monte-Carlo gradient estimator that has both properties of reduced variance and efficient computation. As can be seen, the strength and weaknesses of both estimators are complementary. Therefore, an intuitive step towards an estimator with the above-mentioned desirable properties is to combine the SF with the MVD estimator.

For this reason, we will introduce two approaches for such a combination in this chapter. First, we will formulate an unbiased estimator based on the convex combination between SF and MVD estimators. Secondly, we introduce a biased estimator that makes use of previous gradient directions and corrects them using MVD gradients at a subset of parameter dimensions.

4.1. Convex Combination

The idea behind using a convex combination of SF and MVD estimator is to use the gradient from SF estimation as a basis and improve variance along a subset of the parameter dimensions with a convex combination of SF and MVD. We assume that reduced variance along all dimensions is not necessary for improved performance in a high-dimensional space. As other works suggest, contribution of the parameters to overall performance is limited to a subset of the parameter dimensions [23, 42]. For this reason, we aim to selectively improve variance along a subset of dimensions using a convex combination of the SF and MVD estimator. Overall, there are two essential aspects of this approach. First, how the weights of the combination are determined, and second how the relevant dimensions are selected.

In general if both combined estimators are unbiased, the convex combination also yields an unbiased estimator. Let $\boldsymbol{\eta}_1(\boldsymbol{\omega})$ and $\boldsymbol{\eta}_2(\boldsymbol{\omega})$ be two unbiased estimators for gradient $\nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega})$, then

$$\begin{aligned}\mathbb{E}\left[c\boldsymbol{\eta}_1(\boldsymbol{\omega}) + (1-c)\boldsymbol{\eta}_2(\boldsymbol{\omega})\right] &= c\mathbb{E}\left[\boldsymbol{\eta}_1(\boldsymbol{\omega})\right] + (1-c)\mathbb{E}\left[\boldsymbol{\eta}_2(\boldsymbol{\omega})\right] \\ &= c\nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega}) + (1-c)\nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega}) = \nabla_{\boldsymbol{\omega}} J(\boldsymbol{\omega}).\end{aligned}$$

Therefore combining a SF estimator with a MVD estimator along selected dimensions will still yield an unbiased estimator. At the same time, this strategy allows us to reduce the variance along a selected subset of dimensions.

4.1.1. Combination Weights

A naive way to choose the combination weights is to replace the estimated SF gradients with the MVD gradients for certain dimensions. However, this approach assumes that the MVD estimator will always yield a lower variance, which is not guaranteed to be the case [27]. Additionally, a replacement effectively discards the information gained from the SF estimation. A more principled approach is given by the inverse-variance weighting [11, 29]. Let the estimator variance of the SF be $\hat{\sigma}_{SF} = \mathbb{V}\left[\boldsymbol{\eta}_N^{SF}\right]$ and the MVD estimator variance be $\hat{\sigma}_{MVD} = \mathbb{V}\left[\boldsymbol{\eta}_M^{MVD}\right]$, then the weighting is given by

$$c_{SF} = \frac{\hat{\sigma}_{SF}^{-2}}{\hat{\sigma}_{SF}^{-2} + \hat{\sigma}_{MVD}^{-2}} \qquad\qquad c_{MVD} = 1 - c_{SF}$$

Through this weighting, the convex combination yields an estimator that has the least variance among all weighted estimators [11]. However, computing the estimator variances poses a problem as a faithful computation requires repeated estimation of $\boldsymbol{\eta}_N^{SF}$ and $\boldsymbol{\eta}_M^{MVD}$. For each computation of the SF estimator $\boldsymbol{\eta}_N^{SF}$, $\mathcal{O}(N)$ function evaluations are needed and $\mathcal{O}(2MK)$ evaluations are needed for a computation of the MVD estimator $\boldsymbol{\eta}_M^{MVD}$, where K is the number of selected dimensions. However, this increase in the required number of evaluations defeats our original goal of computational efficiency, as the major computational burden is the evaluation of the black-box function.

We know that the variance of Monte-Carlo estimators decrease with the number of samples N as $\mathcal{O}(1/N)$. Since each estimator $\boldsymbol{\eta}_N$ is computed from averaging N samples $\hat{\boldsymbol{g}}$, we can

use the empirical variance of the samples $\mathbb{V}\left[\hat{\boldsymbol{g}}\right]$ to approximate the estimator variance $\mathbb{V}\left[\boldsymbol{\eta}_N\right]$ as

$$\mathbb{V}\left[\boldsymbol{\eta}_N\right] \approx \frac{\mathbb{V}\left[\hat{\boldsymbol{g}}\right]}{N}$$

Let $\bar{\sigma}_{SF}$ be the variance $\mathbb{V}\left[\hat{\boldsymbol{g}}^{SF}\right]$ of the SF estimator $\boldsymbol{\eta}_N^{SF}$ with N samples and let $\bar{\sigma}_{MVD}$ be the variance $\mathbb{V}\left[\hat{\boldsymbol{g}}^{MVD}\right]$ of the MVD estimator $\hat{\boldsymbol{g}}_M^{SF}$ using M samples, then we reformulate the weighting as

$$c_{SF} = \frac{\bar{\sigma}_{SF}^{-2}/N}{\bar{\sigma}_{SF}^{-2}/N + \bar{\sigma}_{MVD}^{-2}/M} \qquad\qquad c_{MVD} = 1 - c_{SF}.$$

Estimating $\bar{\sigma}_{SF}$ and $\bar{\sigma}_{MVD}$ however requires at least two samples for the respective estimator. In cases, where less than two samples are available, we choose the replacement strategy instead.

4.1.2. Selection Strategy

An important question is how we select the dimensions along which the MVD gradient should be estimated. A simple baseline is to select dimensions uniformly random. However, this does not take into account the particular structure of the optimization problem.

An alternative approach is to select the dimensions with the highest variance. We have two possibilities to compute this variance. The first option computes the empirical variance $\mathbb{V}\left[\hat{\boldsymbol{g}}^{SF}\right]$ from the SF samples. With the second option, we estimate the variance from an outer product of the estimated gradient, which follows the variance estimation method from Choromanski et al. [7]. This second version of the variance estimation is similar to the non-central second-moment approximation found in ADAM [20, 1]. These two variance computations schemes can be additionally extended by computing them on a moving average basis. The moving average version of the two variance estimation schemes is shown in algorithm 7, with the empirical variance formulation as V1 and the outer product version as V2.

Algorithm 7: Updating covariance matrix with version V1 in blue and version V2 in orange

Input : covariance matrix Cov, decay rate r, SF samples $\hat{\boldsymbol{g}}^{SF}_{1\cdots N}$, CCMVD estimator $\boldsymbol{\eta}^{CCMVD}$

1 $\mathrm{Cov} \leftarrow (1-r)\,\mathrm{Cov} + r\mathbb{V}\left[\hat{\boldsymbol{g}}^{SF}_{1\cdots N}\right]$

2 $\mathrm{Cov} \leftarrow (1-r)\,\mathrm{Cov} + r\boldsymbol{\eta}^{CCMVD}\left(\boldsymbol{\eta}^{CCMVD}\right)^{\intercal}$

4.1.3. Algorithm

With a selection strategy and the combination weights, we now have everything in place to formulate an algorithm for episodic policy search using a convexly combined estimator from the SF and the MVD, which we term CCMVD.

As has been mentioned in section 2.2, heuristics can have a large impact on the performance of policy search methods. For a fair comparison with the state-of-the-art ARS, we extend CCMVD with the same state and reward normalization as ARS. Similar to ARS the SF estimator in line 2 of algorithm 8 use antithetic sampling and in general follows algorithm 5. The MVD estimation in line 7 follows the previously specified algorithm 6 with the additional extension of the state normalization heuristic from ARS.

One difference between the CCMVD and ARS is that the standard deviation of return σ_R is computed from the combined returns collected from SF and MVD estimation. A further extension to be elaborated is a warm-up period in line 7 of algorithm 8, where we initially sample MVD dimensions uniformly random before selecting them using the covariance matrix. The warm-up period has the purpose of improving the running average estimate of the covariance matrix.

As we can see in algorithm 8, the CCMVD estimator has three parameters: the number of SF sample N, the number of MVD samples M and the number of dimensions K to estimate the MVD gradient. These terms determine the computational complexity, which results in $2(N + KM)$ function evaluations. At the same time, these terms also determine how much variance reduction can be achieved, as the reduction is directly determined by how many dimensions we select and how many MVD samples we use for each dimension. In general, we would like to find a CCMVD estimator that uses the same budget of function evaluations as a SF estimator while achieving less variance and faster convergence. As we will see in chapter 5, this choice of N, M and K proves to be complicated.

Algorithm 8: CCMVD Gradient Estimation

Hyperparameters : standard deviation σ, number of SF samples N, number of MVD samples M, number of MVD dimensions K, warm-up period w, decay rate r

Input : initial parameters $\boldsymbol{\omega}$, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$, current iteration t

/* 1) gradient estimation following ARS with antithetic SF and heuristics. */

$\boldsymbol{\eta}_N^{SF}, \hat{\boldsymbol{g}}_{1\cdots N}^{SF} \leftarrow$ `antithetic-sf-gradient`$(\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D), f(\boldsymbol{\theta}), \pi_{\boldsymbol{\theta}}, N)$

Cov $\leftarrow$ Update covariance according to V1 or V2 from algorithm 7

if $t < w$ **then**

  Sample dimensions $d_{1\cdots K}$ uniformly random

else

  $d_{1\cdots K} \leftarrow$ select K dimensions with the highest trace of Cov

/* 2) gradient estimation for selected dimensions $d_{1\cdots K}$ with MVD using ARS heuristics. */

$\boldsymbol{\eta}_M^{MVD}, \hat{\boldsymbol{g}}_{1\cdots M}^{MVD} \leftarrow$ `mvd-gradient`$(\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D), f(\boldsymbol{\theta}), \pi_{\boldsymbol{\theta}}, M, d_{1\cdots K})$

$\boldsymbol{\eta}^{CCMVD} \leftarrow$ use SF estimator $\boldsymbol{\eta}_N^{SF}$ as basis

/* 3) approximating inverse-variance weighting */

$\hat{\boldsymbol{\sigma}}_{SF} \leftarrow$ `trace`$(\mathbb{V}[\hat{\boldsymbol{g}}_{1\cdots N}^{SF}])/N$

$\hat{\boldsymbol{\sigma}}_{MVD} \leftarrow$ `trace`$(\mathbb{V}[\hat{\boldsymbol{g}}_{1\cdots M}^{MVD}])/M$

/* 4) combine MVD with base SF along selected dimensions. */

for d *in* $d_1 \cdots d_K$ **do**

  $c_{SF} \leftarrow \left(1 + \frac{\hat{\sigma}_{SF}[d]^2}{\hat{\sigma}_{MVD}[d]^2}\right)^{-1}$

  $\boldsymbol{\eta}^{CCMVD}[d] \leftarrow c_{SF}\, \boldsymbol{\eta}_N^{SF}[d] + (1 - c_{SF})\, \boldsymbol{\eta}_M^{MVD}[d]$

return $\boldsymbol{\eta}_N^{CCMVD}$

4.2. Locally Updated Gradient

If the gradient is smooth, we can expect that gradients from a past iteration to be close to the current gradient. We can then make use the gradient from previous iterations by correcting it along selected dimensions. Let us assume that the distributional parameters $\boldsymbol{\omega} \in \mathbb{R}^D$ and $\boldsymbol{e}_i$ with $i \in \mathcal{K}$ denote unit vectors describing the selected dimensions given by set $\mathcal{K} \subseteq \mathcal{D}$ with $\mathcal{D} = \{1 \cdots D\}$. Let also $\boldsymbol{\eta}(\boldsymbol{\omega}_t)$ describe the estimated gradient of $\boldsymbol{\omega}$ at iteration t, then

$$\boldsymbol{\eta}(\boldsymbol{\omega}_t) = \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1}) + \underbrace{\sum_{i \in \mathcal{K}} \boldsymbol{e}_i \left(\boldsymbol{\eta}(\boldsymbol{\omega}_t) - \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1})\right)}_{\text{correct gradient along dimensions of } \mathcal{K}}$$

This gradient estimate is biased as it derives from a gradient estimate with respect to a parameter from a previous iteration, even if $\boldsymbol{\eta}$ itself is an unbiased estimator.

$$\begin{aligned}
\mathbb{E}\left[\boldsymbol{\eta}(\boldsymbol{\omega}_t)\right] &= \mathbb{E}\left[\boldsymbol{\eta}(\boldsymbol{\omega}_{t-1})\right] + \mathbb{E}\left[\sum_{i \in \mathcal{K}} \boldsymbol{e}_i \left(\boldsymbol{\eta}(\boldsymbol{\omega}_t) - \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1})\right)\right] \\
&= \mathbb{E}\left[\sum_{j \in \mathcal{D}} \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1}) \boldsymbol{e}_j - \sum_{j \in \mathcal{K}} \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1}) \boldsymbol{e}_j\right] + \mathbb{E}\left[\sum_{i \in \mathcal{K}} \boldsymbol{\eta}(\boldsymbol{\omega}_t) \boldsymbol{e}_i\right] \\
&= \mathbb{E}\left[\sum_{j \in \mathcal{D} \setminus \mathcal{K}} \boldsymbol{\eta}(\boldsymbol{\omega}_{t-1}) \boldsymbol{e}_j\right] + \mathbb{E}\left[\sum_{i \in \mathcal{K}} \boldsymbol{\eta}(\boldsymbol{\omega}_t) \boldsymbol{e}_i\right] \\
&= \sum_{j \in \mathcal{D} \setminus \mathcal{K}} \mathbb{E}\left[\boldsymbol{\eta}(\boldsymbol{\omega}_{t-1})\right] \boldsymbol{e}_j + \sum_{i \in \mathcal{K}} \mathbb{E}\left[\boldsymbol{\eta}(\boldsymbol{\omega}_t)\right] \boldsymbol{e}_i \\
&= \sum_{j \in \mathcal{D} \setminus \mathcal{K}} \nabla_{\boldsymbol{\omega}_{t-1}} J(\boldsymbol{\omega}_{t-1}) \boldsymbol{e}_j + \sum_{i \in \mathcal{K}} \nabla_{\boldsymbol{\omega}_t} J(\boldsymbol{\omega}_t) \boldsymbol{e}_i
\end{aligned}$$

The first summand in the last line corresponds to the unbiased gradient estimate of the previous iteration $t-1$. At iteration t however this estimate becomes biased. For the second summand, we have a unbiased gradient estimate along the dimensions of $\mathcal{K}$. However, by our smoothness assumption the corrected gradient should still stay close to the true one. At the same time, as the correction term only involves computing gradients along a subset of dimensions, we can reduce the number of function evaluations.

4.2.1. Algorithm

In our case, we can compute the gradient with the SF estimator at every n-th iteration and correct at a subset of dimensions with the MVD at iterations in-between. This biased gradient estimator, which we call Local-Update (LOCU) estimator, is shown in algorithm 9. We make use of the same dimension selection strategies as for the CCMVD estimator detailed in algorithm 8. Furthermore, the same state normalization and reward normalization as for ARS and CCMVD are applied here. We chose to use a simple replacement strategy instead of combination weights from section 4.1. The reason is that the empirical variance of the SF gradient samples $\mathbb{V}\left[\hat{\boldsymbol{g}}^{SF}\right]$ is not available at the iteration when the MVD is used for correction.

Algorithm 9: Gradient Estimation with LOCU Estimator

Hyperparameters : standard deviation σ, number of SF samples N, number of MVD samples M, number of MVD dimensions K, warm-up period w, decay rate r, update interval n

Input : initial parameters $\boldsymbol{\omega}$, distribution $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D)$, policy $\pi_{\boldsymbol{\theta}}$; black-box function $f(\boldsymbol{\theta})$, current iteration t

if $t \bmod n = 0$ **then**
  /* 1) Full antithetic SF gradient following ARS in algorithm 5. */
  $\boldsymbol{\eta}_N^{SF}, \hat{\boldsymbol{g}}_{1\cdots N}^{SF} \leftarrow$ `estimate-sf-gradient`$(\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D), f(\boldsymbol{\theta}), \pi_{\boldsymbol{\theta}}, N)$
  Cov $\leftarrow$ Update covariance according to V1 or V2 from algorithm 7
  $\boldsymbol{\eta}^{LOCU} \leftarrow \boldsymbol{\eta}_N^{SF}$
else
  if $t < w$ **then**
    Sample dimensions $d_{1\cdots K}$ uniformly random
  else
    $d_{1\cdots K} \leftarrow$ select K dimensions with the highest trace of Cov
  /* 2) Partial gradient for correction at dimensions $d_{1\cdots K}$ using MVD with ARS heuristic. */
  $\boldsymbol{\eta}_M^{MVD}, \hat{\boldsymbol{g}}_{1\cdots M}^{MVD} \leftarrow$ `estimate-mvd-gradient`$(\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma \boldsymbol{I}_D), f(\boldsymbol{\theta}), \pi_{\boldsymbol{\theta}}, M, d_{1\cdots K})$
  for d *in* $d_1 \cdots d_K$ **do**
    $\boldsymbol{\eta}^{LOCU}[d] \leftarrow \eta_M^{MVD}[d]$
  return $\boldsymbol{\eta}^{LOCU}$

5. Experiments

This chapter presents the experiments that demonstrate the potential benefit of using the MVD for episodic policy search. At first, we will explain our experimental setup. Then in section 2, we analyze the properties of SF and MVD estimator without any extensional heuristics in a simplified setting. Lastly, in section 3, we present the results for combining the SF and MVD estimator.

5.1. Experimental Setup

General Setup. We follow the setup of ARS [25] and ES [35]. The upper-level policy distribution is chosen to be a Gaussian with fixed isotropic covariance $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma^2 \boldsymbol{I}_D)$, where σ is set to be a hyperparameter. We therefore only estimate the gradient $\nabla_{\boldsymbol{\mu}} J(\boldsymbol{\mu})$. Following ARS, we set the parameter for the lower-level policy $\boldsymbol{\theta}$ as a linear policy and initialize it with a zero matrix.

Throughout our analysis, we will utilize the antithetic version of the SF estimator and combine MVD with coupling as both extensions yield high variance reductions for the respective estimator. When analyzing the combined estimators, we compare to an antithetic SF estimator extended with state and reward normalization from ARS, which essentially results in the original ARS algorithm. The number of samples used for the baseline antithetic SF are transferred from Mania et al. [25]. Unless otherwise stated, the optimizer used will be SGD.

Environments. For comparing the estimators in chapter 2, we use the black-box test functions Quadratic ,Styblinski and Rosenbrock. Their analytical expressions with a 3D plot are shown in figure 5.1. The comparison of the estimators in the episodic setting is conducted on the Mujoco environment Swimmer-v3 [39]. All other experiments are

conducted using version 3 of the Mujoco locomotion tasks included in OpenAI Gym [4, 39]. The dimensions for the linear policies for each environment are listed in table 5.1.

Environment	Parameter Dimensions
Swimmer-v3	16
Hopper-v3	33
HalfCheetah-v3	102
Walker2d-v3	102
Ant-v3	888

Table 5.1.: Parameter dimensions for the Mujoco environments used. The size of the parameter dimension is determined from the size of the observation space multiplied by the size of the action space.

Evaluation Metrics. We evaluate the gradient estimator regarding the average reward achieved, the variance, and the deviation from a reference gradient. The average reward is computed as an average over 100 evaluations of the black-box function.

For the variance of the gradient estimator $\mathbb{V}\left[\boldsymbol{\eta}_N(\boldsymbol{\omega})\right]$, we compute 100 samples of $\boldsymbol{\eta}_N$ and calculate the variance from these samples. The shown variance is averaged over all parameter dimensions. The reduction in the variance of estimator $\boldsymbol{\eta}$ relative to $\boldsymbol{\eta}'$ is measured as

$$\frac{\mathbb{V}\left[\boldsymbol{\eta}\right] - \mathbb{V}\left[\boldsymbol{\eta}'\right]}{\mathbb{V}\left[\boldsymbol{\eta}'\right]}.$$

Here, a negative value indicates a reduction with respect to $\boldsymbol{\eta}'$. We measure variance reduction at 10 evaluation points across the optimization process and show the averaged variance reduction across those 10 evaluation points.

We measure the deviation of the estimated gradient $\boldsymbol{\eta}$ from a reference gradient $\boldsymbol{g}$ as the deviation in angle and in magnitude. The deviation in angle is given as the cosine distance

$$d_{\sphericalangle} = 1 - \frac{\langle \boldsymbol{\eta}, \boldsymbol{g} \rangle}{||\boldsymbol{\eta}||||\boldsymbol{g}||}$$

and the deviation in magnitude is expressed through the relative absolute distance

$$d_{||\cdot||} = \frac{||\boldsymbol{\eta} - \boldsymbol{g}||}{||\boldsymbol{g}||}.$$

Here, we want both distances to be close to 0. In cases where a true gradient is available, such as for the black-box test functions, we use the auto-differentiation library jax [3] to compute a reference gradient. Otherwise, the reference gradient is estimated from an antithetic SF estimator. The number of Monte-Carlo samples N for the reference gradient is determined through $\min(100D, 1000)$, where D denotes the dimensionality of the parameter space.

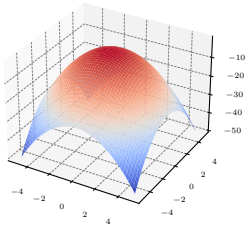
5.2. Comparing Estimators

Before analyzing the results for a combination of estimators, it makes sense to examine how each estimator performs individually. Doing so gives us an initial intuition about the expected improvement from adding the MVD estimator. Here, we are particularly interested in the case when both estimators use the same number of function evaluations, which are the most expensive computation.

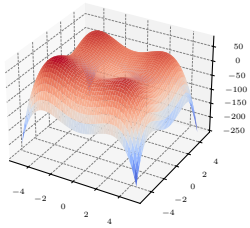
We first compare the SF and MVD estimator on a set of commonly used black-box test functions. We then examine both estimators in an episodic setting using the Mujoco environment Swimmer-v3. Lastly, we look at the influence of the ARS heuristics on the estimator variance. The comparison is made without using additional heuristics such as state or reward normalization. The influence of those heuristics is studied separately at the end of this section.

5.2.1. Test Functions

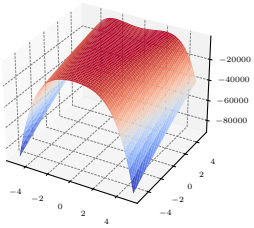
Zeroth-order optimization such as ES has been argued to be strongly affected by the dimensionality of the problem [35, 28]. Test functions used in the black-box optimization literature offer an excellent opportunity to probe the performance of our estimators on varying dimensions, as the dimensionality is only a parameter of the function. For this purpose, we analyze SF and MVD estimators on the Quadratic, Styblinski, and Rosenbrock, which are shown in figure 5.1.



(a) $f(\boldsymbol{\theta}) = \sum_{d=1}^{D} \theta_d^2$



(b) $f(\boldsymbol{\theta}) = \frac{1}{2} \sum_{d=1}^{D} \theta_d^4 - 16\theta_d^2 + 5\theta_d$



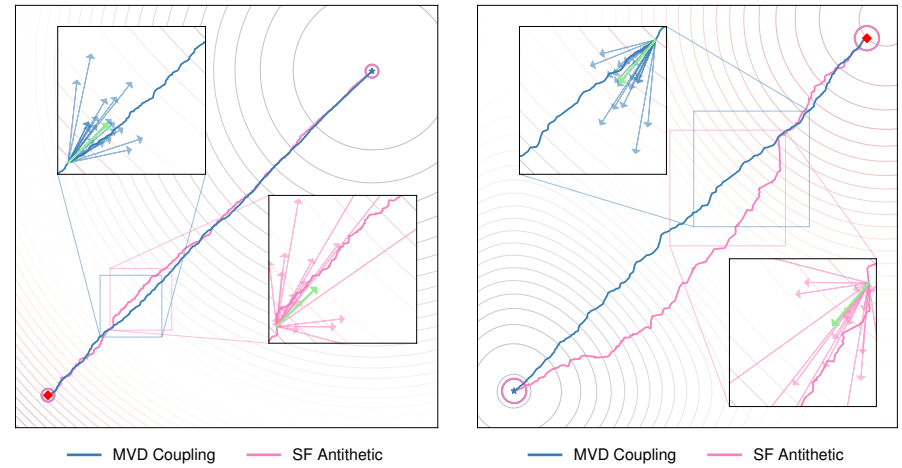
(c) $f(\boldsymbol{\theta}) = \sum_{d=1}^{D-1} 100(\theta_{d+1} - \theta_d^2)^2 + (\theta_d - 1)^2$

Figure 5.1.: Shown are the black-box test functions. Figure 5.2a depicts the Quadratic function, figure 5.2c shows the Rosenbrock and figure 5.2b displays the Styblinski function.

Before analyzing the estimators on higher-dimensions, we first gain an intuition for the two dimensional case. In figure 5.2, we see the trajectories of the optimized mean for both estimators along with the contour of the cost function. We also visualize the estimated gradients and the true gradient as arrows in the zoomed-in areas, where the true gradient is shown as the green arrow. Here it is obvious, that the gradients estimated with the MVD are more centered around the true gradient.

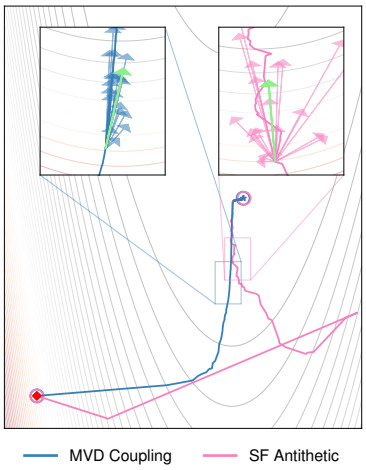
In higher dimensions, we can observe in figure 5.3 that all estimators perform similarly well in terms of average reward achieved. Despite the similarity, the variance and other properties for the corresponding reward curves show a qualitative difference between the estimators.

Variance. In figure 5.4, we plot the estimator variance $\mathbb{V}\left[\boldsymbol{\eta}_N(\boldsymbol{\omega})\right]$ that corresponds to the reward curves in figure 5.3. The variance is lower for the MVD in all test functions. For Rosenbrock, the improved variance of the MVD estimator is only observed at an early



(a) Quadratic

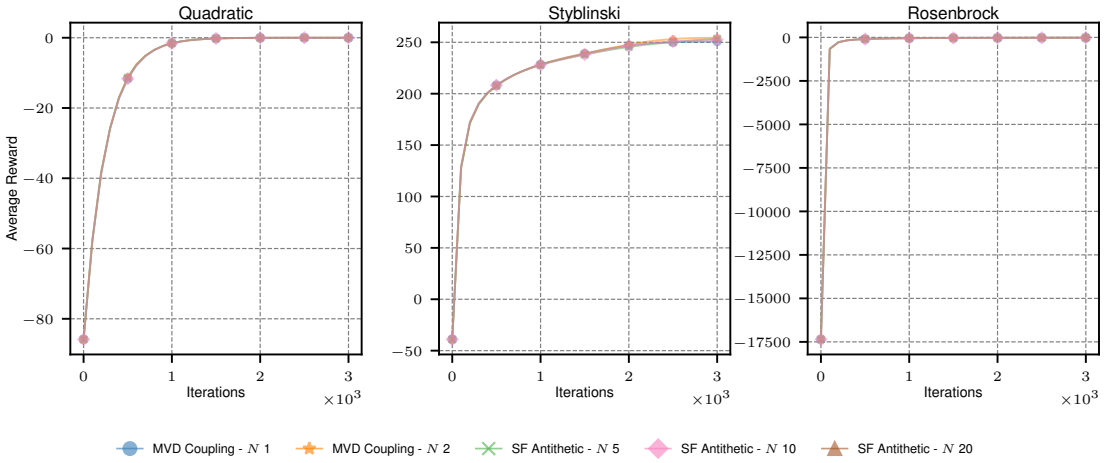
(b) Styblinski



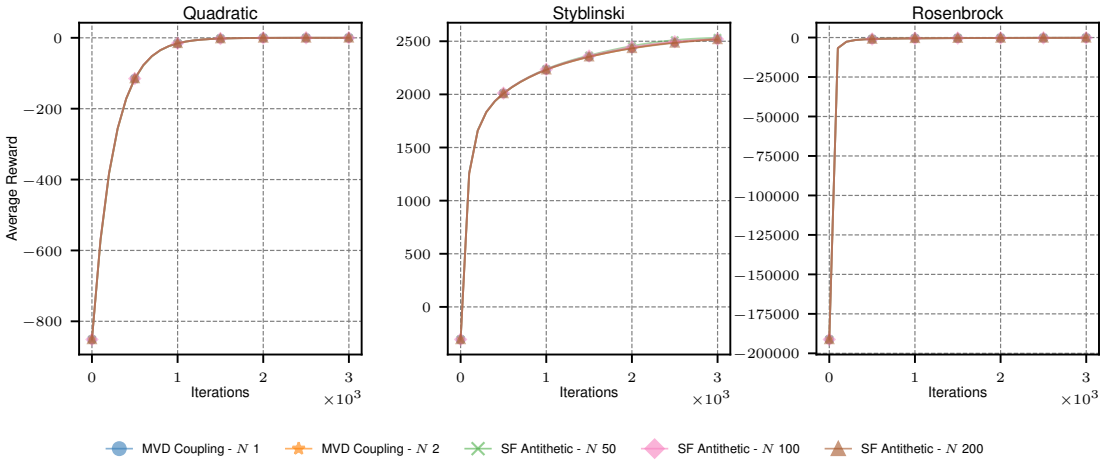
(c) Rosenbrock

Figure 5.2.: Shown are the trajectories of the optimized mean for the antithetic SF and MVD. The red diamond depicts the starting point. The arrows in the zoomed-in area shows samples of the estimated gradients. The green arrow corresponds to the true gradient.

stage of optimization. However, looking at the reward curves in figure 5.4 we can see that the Rosenbrock converges early. For the other functions, a clear advantage of the MVD estimator can be noticed. Variance is lower for the MVD estimator even when compared



(a) Average reward on the test functions with the parameter dimensions set to $D = 10$. The antithetic SF estimator with $N = 10$ and $N = 20$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.



(b) Average reward on the test functions with the parameter dimensions set to $D = 100$. The antithetic SF estimator with $N = 100$ and $N = 200$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

Figure 5.3.: Comparison of the average reward using parameter dimensions $D = 10$ and $D = 100$. The learning and standard deviation used is provided in appendix B.

to a SF estimator with a doubled function evaluation budget. Contrary to our initial assumption, transitioning from 10 dimensions to 100 dimensions does not significantly impact the variance of the estimators.

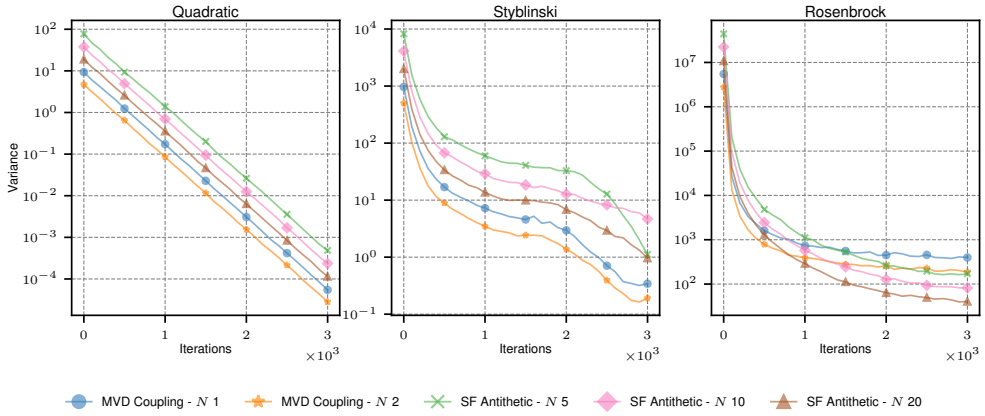
Gradient Distances. Looking at the deviation in magnitude and angle in figure 5.5 and figure 5.6, we observe similar improvements for the MVD as in the case of the variance. The cosine and the relative absolute distance of the MVD estimator are lower than the distances for SF estimator using the function evaluation budget. With a function evaluation budget of two times that of the MVD estimator, the SF estimator still exhibits higher distances. For Rosenbrock, the shown distances exhibit erratic behavior, which could be attributed to its early convergence. Similar to the variance case, the increase of dimensionality hardly affects the gradient distances of the estimators.

5.2.2. Episodic Setting

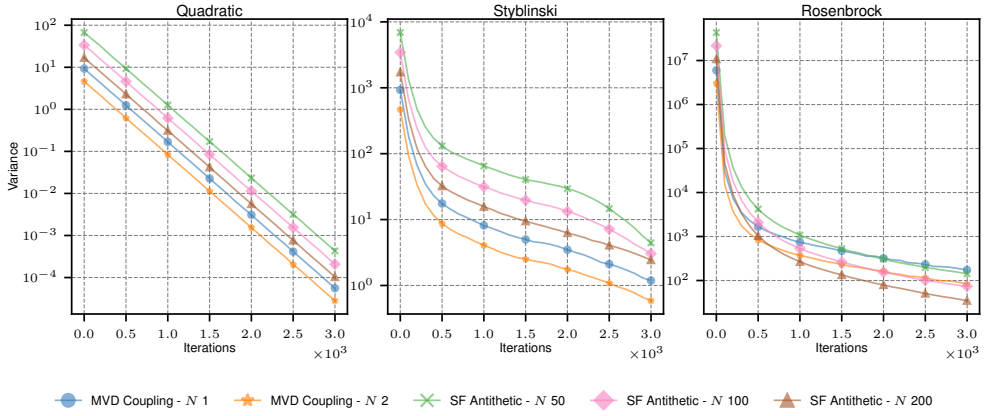
The previous section analyzed the estimator on test functions from the black-box optimization literature. These functions represent a simplified case, where the reward is immediately returned, and there is no notion of an environment or a reinforcement learning agent. In this section, we will examine the estimators in an episodic setting, where the agent has to step through the environment for a fixed horizon length until it receives the reward. As we will see, this change of setting has a significant impact on the qualitative difference of these estimators.

In figure 5.7, we plot the average reward for the Swimmer-v3 environment using a linear policy of dimension 16. The plot depicts the reward corresponding to the other metrics shown in this subsection. As we can see, the performance is similar for all estimators except the single-sample SF estimator. However, despite using a comparable number of function evaluations, the SF estimators slightly improve over the MVD estimators. Compared to the previous test function setting, this decrease in performance for the MVD is also reflected in the other metric we measure.

Variance. In contrast to the test function setting, the MVD estimator loses its advantage over the SF estimator. As can be seen in figure 5.8 the SF estimators using the same amount of evaluations show lower variance than their corresponding MVD counterparts. For example, the MVD using 2 samples and 32 function evaluations has a higher variance than a SF estimator using 16 samples and 16 function evaluations. When considering the

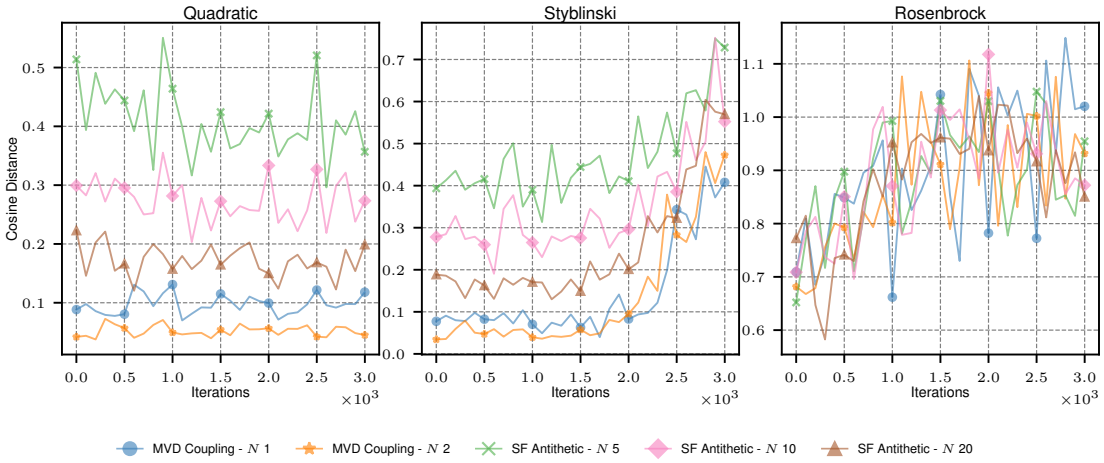


(a) Variance of the gradient estimators for parameter dimensions 10. As in figure 5.3, the antithetic SF estimator with $N = 10$ and $N = 20$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

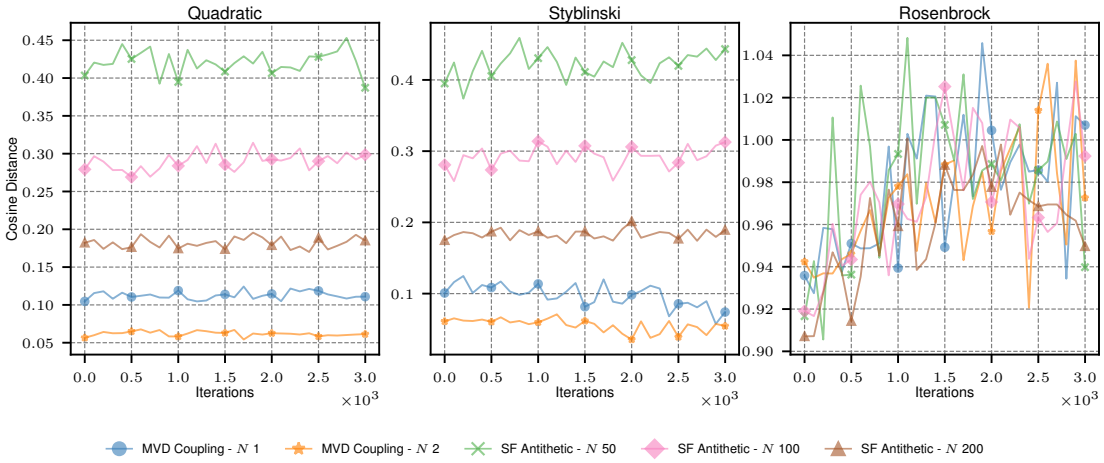


(b) Variance of the gradient estimators for parameter dimensions 100. As in figure 5.3, the antithetic SF estimator with $N = 100$ and $N = 200$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

Figure 5.4.: Comparison of the estimator variance $\mathbb{V}\left[\boldsymbol{\eta}_N(\boldsymbol{\omega})\right]$ for dimensions 10 and 100. The learning and standard deviation used is provided in appendix B.

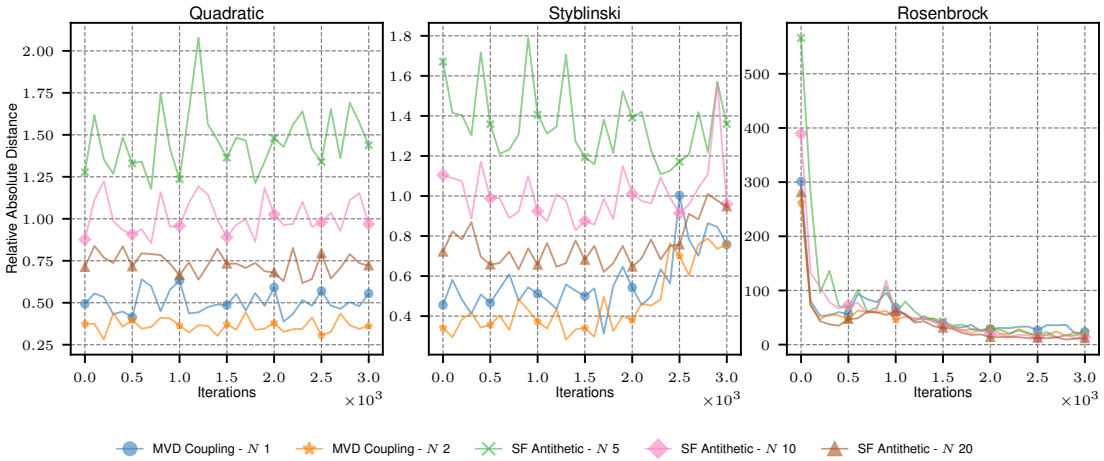


(a) Cosine distance of the gradient estimators for parameter dimensions 10. As in figure 5.3, the antithetic SF estimator with $N = 10$ and $N = 20$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

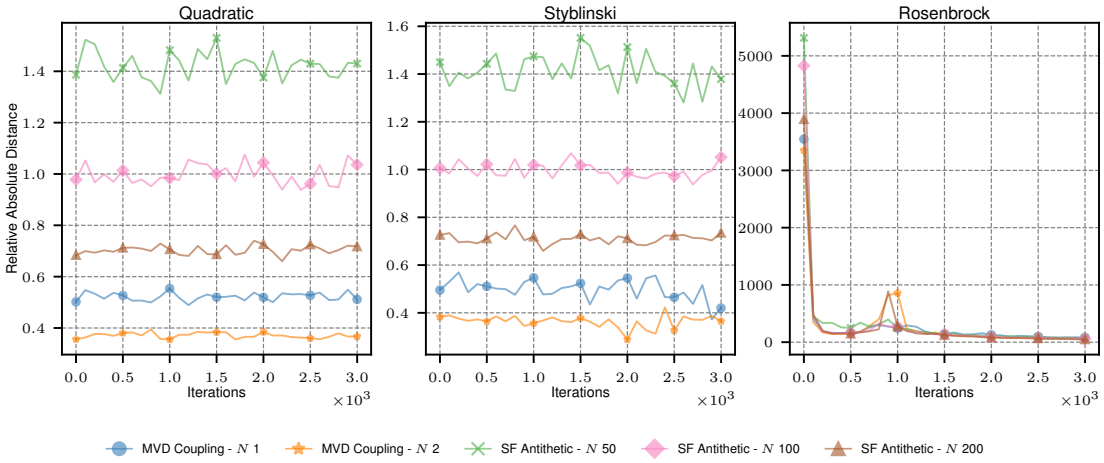


(b) Cosine distance of the gradient estimators for parameter dimensions 100. As in figure 5.3, the antithetic SF estimator with $N = 100$ and $N = 200$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

Figure 5.5.: Comparison of the cosine distance $d_{\triangleleft}$ for dimensions 10 and 100. The learning and standard deviation used is provided in appendix B.



(a) Relative absolute distance of the gradient estimators for parameter dimensions 10. As in figure 5.3, the antithetic SF estimator with $N = 10$ and $N = 20$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.



(b) Relative absolute distance of the gradient estimators for parameter dimensions 10. As in figure 5.3, the antithetic SF estimator with $N = 100$ and $N = 200$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

Figure 5.6.: Comparison of the relative distance absolute $d_{||\cdot||}$ for dimensions 10 and 100. The learning and standard deviation used is provided in appendix B.

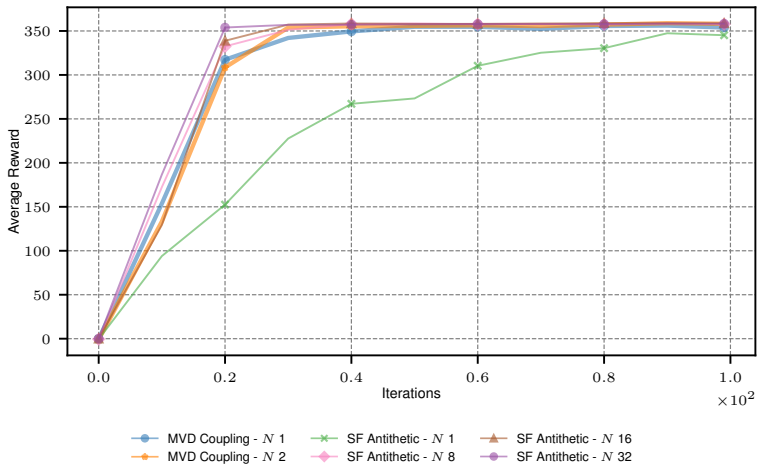


Figure 5.7.: Shown is the average reward for Swimmer-v3. The learning rate is $1\mathrm{e}{-4}$ and the standard deviation is set to 0.1. The antithetic SF estimator with $N = 16$ and $N = 32$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

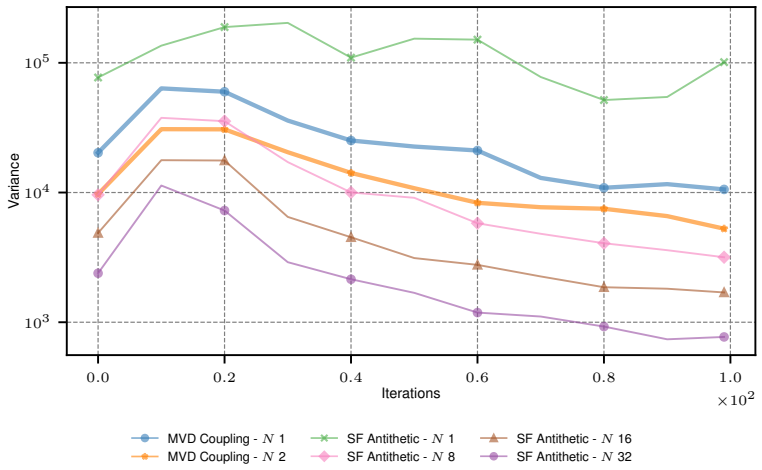


Figure 5.8.: Shown is estimator variance $\mathbb{V}\left[\boldsymbol{\eta}_N(\boldsymbol{\omega})\right]$ for Swimmer-v3. The learning rate is $1\mathrm{e}{-4}$ and the standard deviation is set to 0.1. As in figure 5.7, the antithetic SF estimator with $N = 16$ and $N = 32$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

same budget of function evaluations, the variance is increased for the MVD in the episodic setting. However, increasing the MVD samples to a small fraction of the SF samples still improves variance for the MVD. For example, the MVD using 2 samples shows a similar variance as the SF using 8 samples.

Gradient Distances. The same trend as observed for variance, can be seen in figure 5.9 and figure 5.10 for the cosine distance and relative absolute distance. Looking at the cosine distance in figure 5.9, the MVD using 2 samples and 32 evaluations is approximately on par with the SF estimator using 16 samples and 16 evaluations, thus requiring 16 evaluations more. Similar observations can be made for the relative absolute distance in figure 5.10. Here the single-sample MVD shows a similar deviation in magnitude to the reference gradient as the MVD with two samples. The relative absolute distances for both estimators are comparable to a SF estimator using 8 samples, showing again that the MVD achieves similar performance as the SF estimator using a small fraction of SF samples but more function evaluations.

Effect of ARS Heuristics. As mentioned in section 3.1, episodic policy search algorithms often benefit enormously from additional heuristics used. Therefore, we are interested in how these heuristics affect the estimated gradient. In particular, as our primary goal is the reduction of variance through the usage of the MVD estimator, investigating the effect of applied heuristics allows us to exclude a possible variance reduction through these heuristics. Since we extend the combined estimators with ARS heuristics, we will focus on the state normalization and reward normalization used in ARS.

In figure 5.11, we see the variance for four different cases differing between whether heuristics were used, which heuristic was used or when both heuristics were used. While the use of state normalization largely preserves the variance qualitatively and quantitatively, the application of reward normalization drastically decreases the variance. This decrease can be intuitively understood since both estimators compute the gradient estimate as a weighted difference of rewards. Normalization of the rewards and, in most cases, a reduction in absolute magnitude thus will reduce the variance of the gradient estimator. However, we can also observe that the reward normalization still preserves the qualitative difference between the estimators.

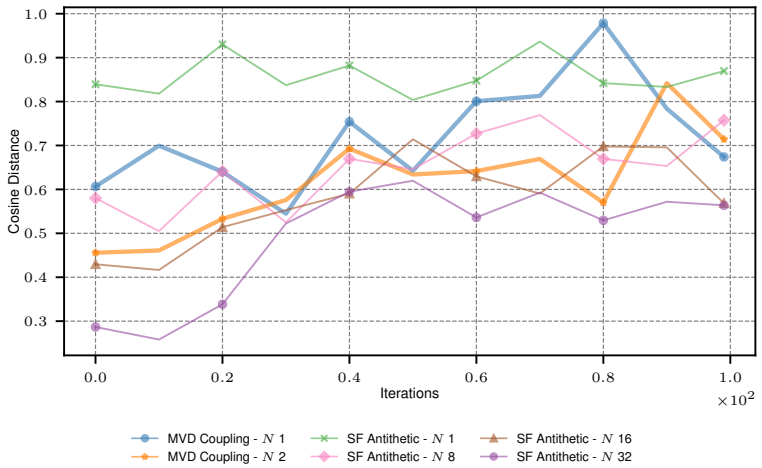


Figure 5.9.: Shown is the cosine distance $d_{\sphericalangle}$ for Swimmer-v3. The learning rate is $1\mathrm{e}{-4}$ and the standard deviation is set to 0.1. As in figure 5.7, the antithetic SF estimator with $N = 16$ and $N = 32$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

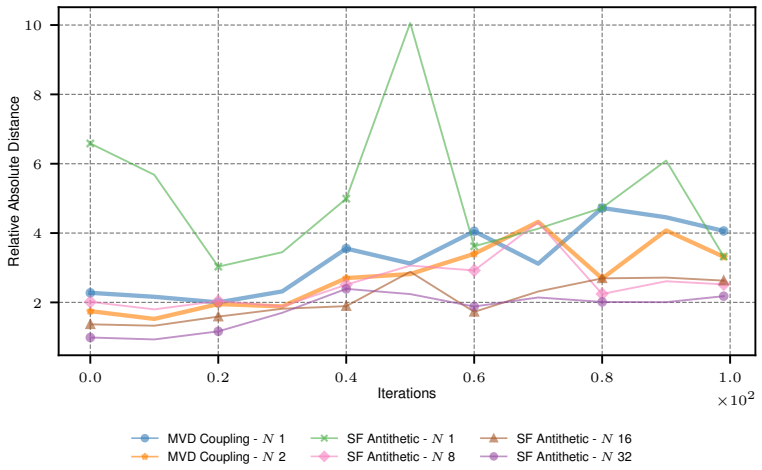


Figure 5.10.: Shown is the relative absolute distance $d_{||\cdot||}$ for Swimmer-v3. The learning rate is $1\mathrm{e}{-4}$ and the standard deviation is set to 0.1. As in figure 5.7, the antithetic SF estimator with $N = 16$ and $N = 32$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

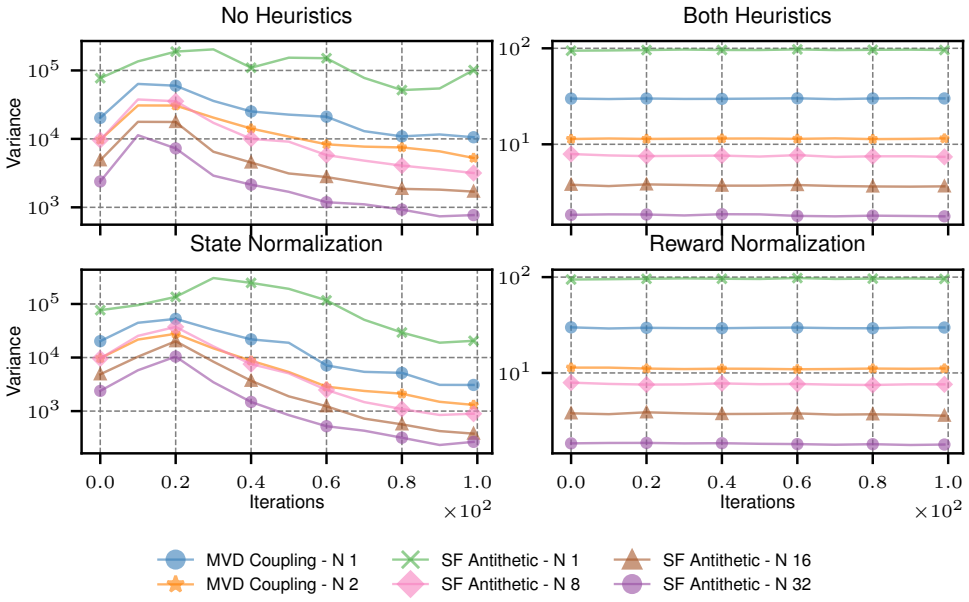


Figure 5.11.: Comparison of the variance between different scenarios of applying ARS heuristics. The learning rate is $1\mathrm{e}{-4}$ and the standard deviation is set to 0.1. As in figure 5.7, the antithetic SF estimator with $N = 16$ and $N = 32$ samples uses the same number of function evaluations as the MVD estimator with $N = 1$ and $N = 2$ samples, respectively.

5.3. Combining Estimators

By estimating MVD gradients along a few specifically chosen dimensions, we aim to mitigate the high computational complexity of the MVD estimator and reduce variance at the chosen dimensions. In this section, we present the results of the combination approaches introduced in chapter 4 that aim to achieve such a goal.

We compare the combined estimators to an antithetic SF estimator extended with the state and reward normalization from ARS. In essence, the extended antithetic SF estimator is equivalent to the original ARS algorithm. In the following, we will refer to it as the baseline SF estimator or simply SF estimator. The number of samples for the baseline SF estimator is adjusted to match the function evaluation budget of the compared combined estimator and are taken over from Mania et al. [25].

5.3.1. Convex Combination

The MVD yields lower variance than the SF for the same number of samples but requires function evaluations linear in the parameter dimensions. Therefore, the crucial component for the CCMVD is to reduce dimensionality by selectively choosing the dimensions for MVD computation.

In section 4.1.1, we have introduced three such selection strategies for the CCMVD. The first one selects dimensions uniformly random and acts as a baseline comparison. The second and third strategies select the dimensions of highest variance from an estimation of the covariance matrix and are described in algorithm 7. The first version V1 of the covariance estimation uses the empirical variance of the Monte-Carlo samples and the second version V2 estimates the covariance using an outer product of the estimated gradient.

Another important aspect is the parameters for the CCMVD. Namely, the number of samples for the base SF estimator N, the number of MVD dimensions K, and the number of samples for each MVD dimension M. Each tuple of parameters (N, K, M) shown in the figures equates to the same amount of function evaluations $2(N + KM)$.

To compare the baseline SF and the CCMVD estimator, we evaluated a grid search for selecting the best learning rate and standard deviation for both estimators. The results of the grid search can be found in appendix A and the chosen hyperparameters are shown in appendix B.

Even though a significant improvement can only be observed for the Hopper-v3, we can still see that the different selection strategies yield different qualitative results. This qualitative difference between dimension selection schemes is summarized in figure 5.12, where we show the best (N, K, M) configuration for each selection scheme together with the SF baseline. The complete results for all selection strategies are given in appendix C.1. Selection based on variance is either close to the SF baseline or improve it. In contrast, the average reward for the random selection is significantly lower than the SF baseline and the variance-based selections.

Variance. Variance reduction with the CCMVD is not always given and depends on the parameter configuration (N, K, M). For this reason, we provide an analysis of the estimator variance to investigate which configurations lead to an increase or decrease in variance. We show results for selection strategy V2 as it achieved the highest rewards previously. Additionally, we compare the CCMVD estimator with a counterpart that relies on the same selection strategy but replaces the MVD with the SF estimator and term this estimator as CCSF. This comparison aims to show the advantage of using the MVD for convex combination over an antithetic SF.

We can see in figure 5.13a the variance reduction of the CCMVD and CCSF relative to an antithetic SF estimator using the same function evaluations budget. Here, a positive percentage describes an increase of variance and a negative percentage a decrease. Firstly, we observe a variance increase for both CCMVD and CCSF irrespective of parameter configuration. Secondly, it can be seen that the variances of the estimators are determined mainly by the number of base SF samples N. Increasing N has a more significant impact on the variance reduction than increasing the number of dimensions K or samples per dimension M. Despite the variance increase for the estimators, a variance decrease is achieved for the selected dimension in the case of specific configurations of (N, K, M). In figure 5.13b, we see that variance at the chosen dimensions is reduced for all cases when the number of samples for the MVD M is larger one. In such cases, variance reduction with the CCMVD is mostly more effective than CCSF. However, as the number of selected dimensions K increases, the number of function evaluations increases with $2KM$. By taking more samples for the CCSF to adjust for the increase of function evaluations, an increased variance reduction for the CCSF is observed. For the configurations that achieved an improved reward, we notice a combination of low increase of variance for the overall estimator combined with a variance decrease at the chosen dimensions, e.g. configuration $(5, 2, 5)$ for Walker2d-v3.

A possible question is why the per-dimension variance increase is much larger for the

CCSF in the single-dimension and single-sample case compared to the other parameter configurations like for Swimmer-v3 or Hopper-v3. Here, a replacement strategy is applied since inverse-variance weighting is not applicable in these specific scenarios. The absence of the weighting scheme can act as a possible explanation for the increase. In the following, we will elaborate on the impact of the weighting scheme.

Influence of Weighting. In figure 5.14, the advantage of applying the inverse-variance weighting over a replacement strategy for CCMVD and CCSF is analyzed. The figures show the per-dimensions variance reduction relative to an antithetic SF estimator with the same function evaluation budget of $2(N + KM)$. In cases where more than one sample is taken, we can approximate the estimator's variance and utilize the inverse-variance weighting. In such cases, we observe for the CCMVD in figure 5.14a a clear advantage of using the weighting scheme over a replacement strategy. The effect of the inverse-variance weighting is most drastic for Walker2d-v3, where an replacement scheme leads to a variance increase of around 700% contrasted with a variance decrease achieved through the inverse-variance weighting. A similar trend is observed for CCSF, where estimators without weighting yield no improvement or an increase in variance. However, applying the weighting results in a variance reduction for all cases. Comparing the influence of the weighting on the CCSF and CCMVD, we can see that CCMVD benefits more from the inverse-variance weighting than the CCSF.

Gradient Distances. We show gradient distances as boxplots in figures 5.15a and 5.15b, where each box summarizes gradient distances collected from 10 evaluation points. We compare to gradient distances of an antithetic SF estimator with the same evaluation budget. Figure 5.15a indicates a lower cosine distance for the antithetic SF estimator than the CCMVD estimators for most environments. For the Walker2d-v3 and HalfCheetah-v3, cosine distances are in a comparable range to the antithetic SF. Except for Walker2d-v3, a lower relative absolute difference for the SF estimator can also be identified in figure 5.15b. Similar to the variance analysis, configurations with a higher number of dimensions K also show a higher relative absolute distance. For the configurations with an improved average reward, the gradient distances do not show a significant difference relative to the antithetic SF comparison.

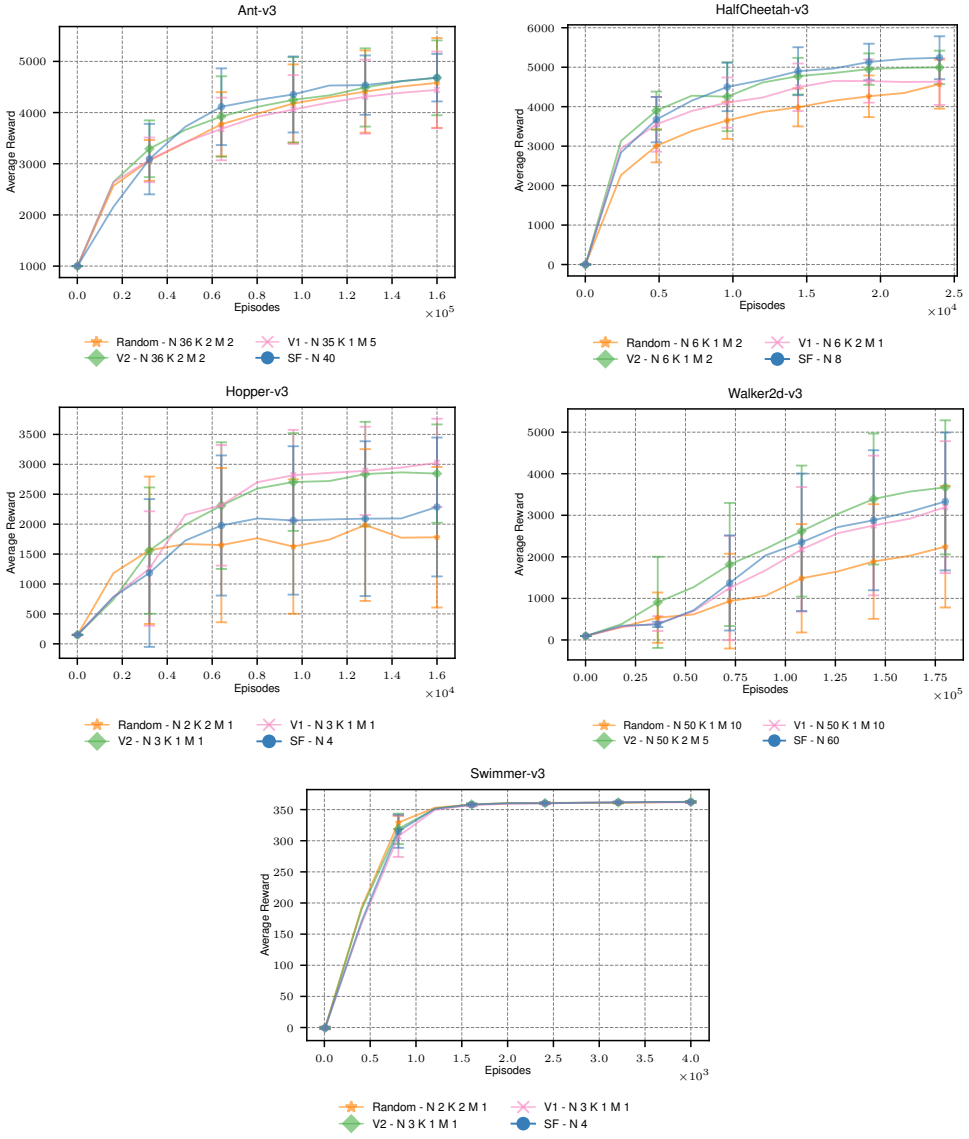
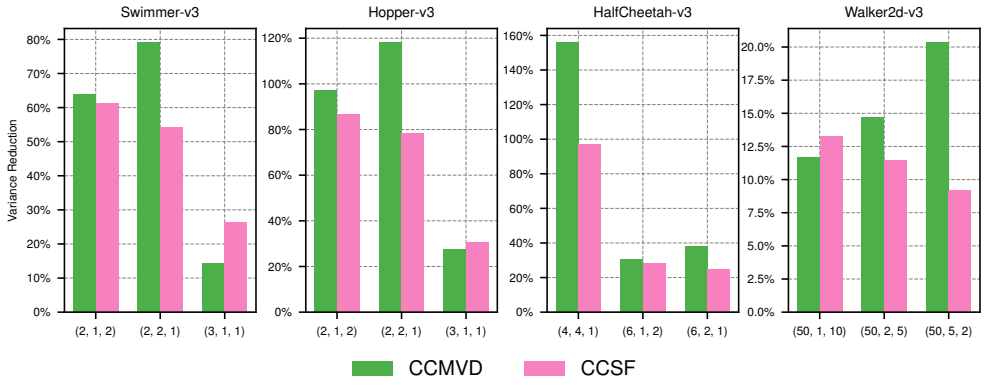
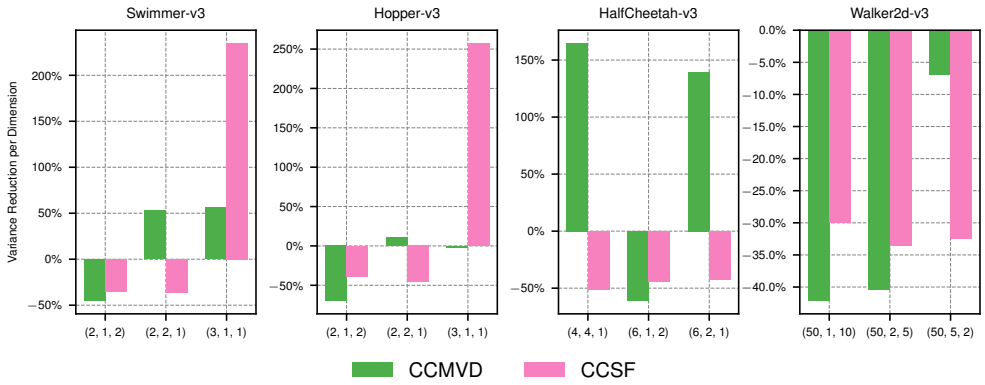


Figure 5.12.: Average reward for the Mujoco environments with selection based on uniform random sampling, variance estimation V1 and variance estimation V2. The tuple (N, K, M) consists of the number of antithetic SF samples N, the number of dimensions K and the number of MVD samples per dimension M. The rewards are averaged over 10 seeds.

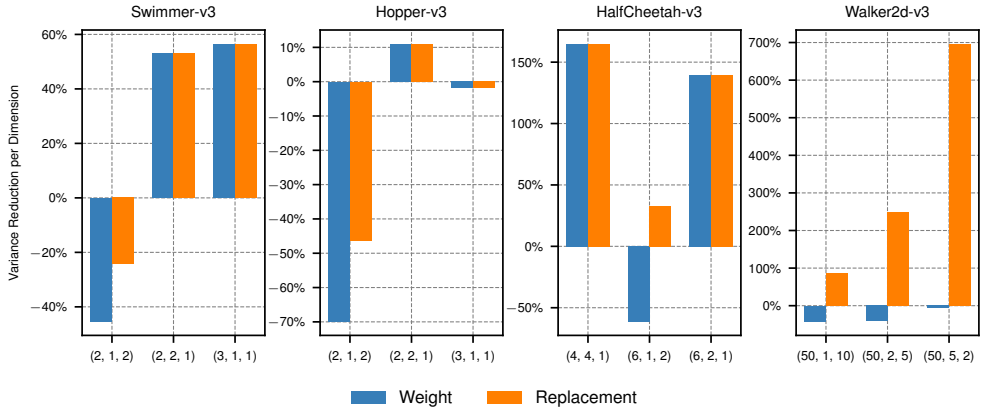


(a) Variance reduction with respect to antithetic SF estimator with the same function evaluation budget of $2(N + MK)$.

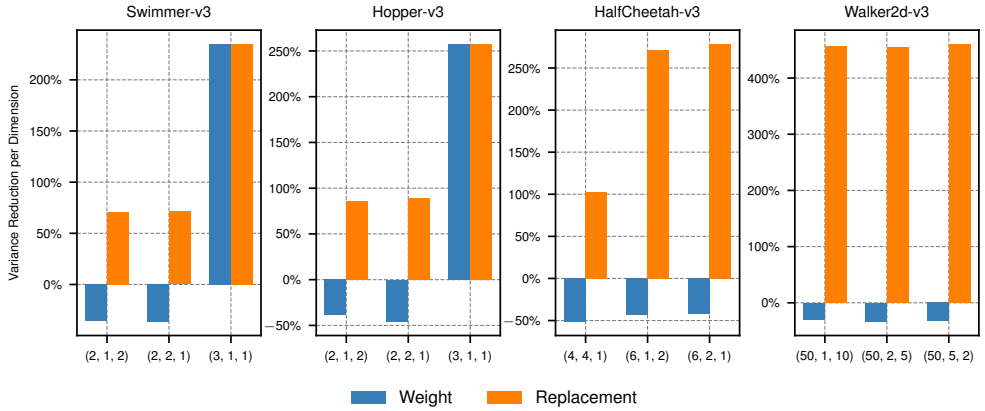


(b) Variance reduction at selected dimensions with respect to antithetic SF estimator with the same function evaluation budget of $2(N + MK)$.

Figure 5.13.: Comparison of variance reduction between CCMVD and CCSF. The upper row show variance reduction relative to an antithetic SF estimator using the same number of function evaluations. The bottom row show variance reduction relative to the base antithetic SF estimator of the convex combination with N samples. The dimensions were selected with selection strategy V2 from algorithm 7. The x-axis shows configurations of (N, K, M).

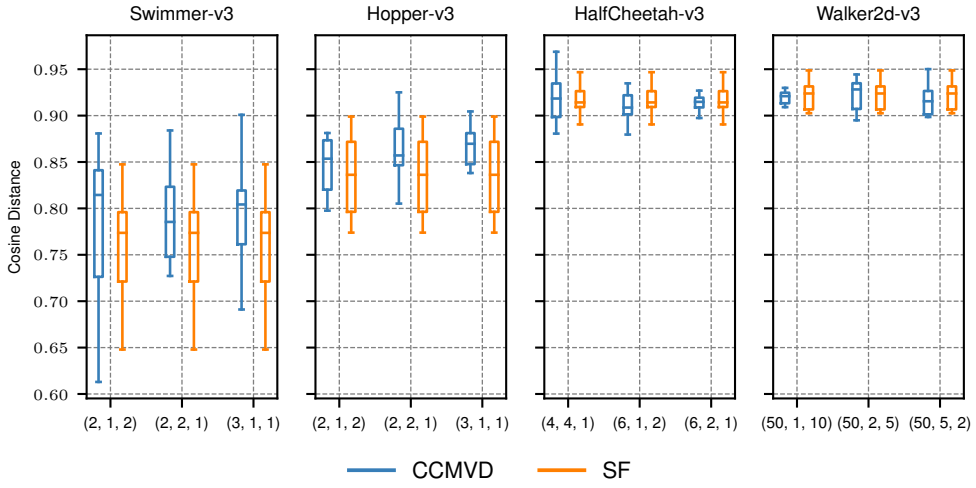


(a) Variance reduction at selected dimensions for CCMVD. The comparison is between estimators with inverse-variance weighting and estimators with replacement strategy.

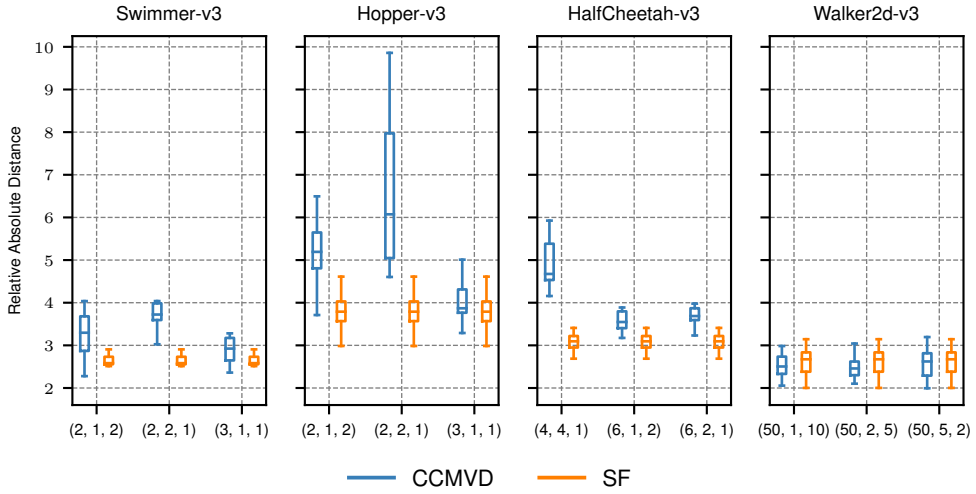


(b) Variance reduction at selected dimensions for CCSF. The comparison is between estimators with inverse-variance weighting and estimators with replacement strategy.

Figure 5.14.: Comparison of variance reduction relative to antithetic SF estimator with the same function evaluation budget of $2(N + KM)$. The selection strategy for the dimensions is given as V2 of algorithm 7. The x-axis shows configurations of (N, K, M).



(a) Comparison of cosine distances $d_{\measuredangle}$ relative a reference gradient.



(b) Comparison of relative absolute distances $d_{||\cdot||}$ relative a reference gradient.

Figure 5.15.: Boxplots of gradient distances comparing CCMVD with an antithetic SF estimator using same evaluation budget of $2(N + MK)$. Each box corresponds to statistics of gradient distances collected from 10 evaluation points at regular intervals during optimization. The x-axis shows configurations of (N, K, M).

5.3.2. Locally Updated Gradient

When the gradients are locally smooth, computational resources can be saved by reusing a previous gradient and correcting the previous gradient only at selected dimensions with an updated gradient. Since the MVD gradient is estimated per dimension, this use case is ideal for applying the MVD for updating the previous gradient.

The same tuple of parameters (N, K, M) as for the CCMVD estimator specifies the LOCU estimator. N denotes the number of samples used for estimating a full antithetic SF gradient, K specifies the number of dimensions to partially correct the full gradient with the MVD in the next iteration and M gives the number of MVD samples per dimension. Here, we only consider the case when the reused full gradient is one iteration apart from the partially corrected gradient. That is partial and full gradient follow each other in consecutive iterations.

In figure 5.16, the average reward for the LOCU estimator on the Mujoco environments is shown for different selection strategies. As for the CCMVD estimator, we show the best configuration for the random selection and selection strategies based on V1 and V2 of algorithm 7. Additionally, we utilize the best hyperparameters from the grid search conducted for the CCMVD and the baseline SF estimator. We reuse the hyperparameters, since the parameter tuple (N, K, M) is shared between the LOCU and CCMVD estimator. An other reason is that the combination process of CCMVD and LOCU only differ in their combination weights applied.

We can first observe that a significant improvement over the SF baseline with the same number of evaluations is not achieved. The second observation is that in some environments like HalfCheetah-v3 and Walker2d-v3, the decrease in performance for the variance-based selection strategies is considerable. In particular, for selection strategy V2, which obtained the best results for the CCMVD estimator, the decrease is notable. A possible explanation for this finding is given in chapter 6.

Variance. Figure 5.17 shows the variance reduction at the chosen dimensions for the partial gradient through applying the MVD. Reduction is relative to an antithetic SF estimator with the same evaluation budget as a full and partial gradient. The figure reveals that replacing MVD estimates at the selected dimensions increases variance for most configurations. Increasing the MVD dimensions K produces a lower variance, Nevertheless, the variance increase is still significant for those cases.

Gradient Distances. For the LOCU estimator, we distinguish between a full gradient computed from N samples of an antithetic SF and a partial gradient that results from correcting the full gradient at K dimensions with M MVD samples.

This part investigates how much the full gradient differs from the partial gradient and how much the partial gradient differs from a reference gradient. For this purpose, we plot the gradient distances collected from 10 evaluation points throughout the optimization process as box plots. In other words, each box shows the statistics for the gradient distances over those 10 evaluation points.

Figures 5.18a and 5.18b show the relative and cosine distance between the partial and the full gradient. These plots depict the deviation of a full gradient from a partial gradient compared to deviations of two reference gradients separated by the same interval. First of all, the difference between consecutive reference gradients is comparatively small. In most cases, the difference is smaller than the distance between the full and partial gradient, even though the partial gradient modifies the full gradient only at a small subset of dimensions. Despite a higher cosine and relative distance between full and partial gradients, the distances are still in a low range as compared to the gradient distances of the estimated gradients and the reference gradients shown in figures 5.18a and 5.18b.

We compare the relative and cosine distances of full and partial gradient to their respective reference gradients in figures 5.18a and 5.18b. We show how much the full and partial gradients deviate from approximated true gradients. It can be seen that, despite reusing the previous gradient, the deviation of the partial gradient from a reference gradient is similar to the deviation of the full gradient from a reference gradient.

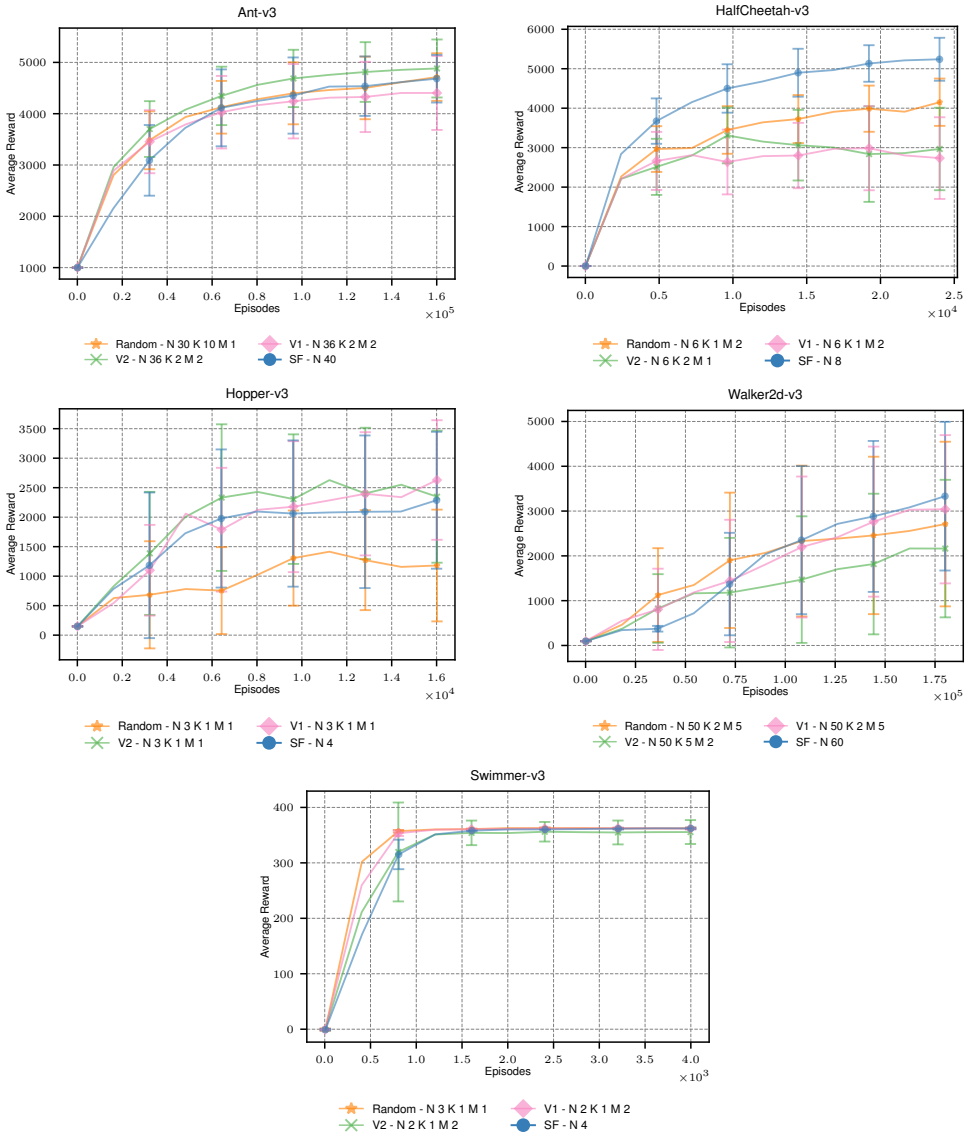


Figure 5.16.: Average reward for the Mujoco environments with dimension selection based on uniform random sampling, variance estimation V1 and variance estimation V2. The tuple (N, K, M) consists of the number of antithetic SF samples N for the full gradient, the number of MVD dimensions K used for the partial gradient, and the number of MVD samples per dimension M. The rewards are averaged over 10 seeds.

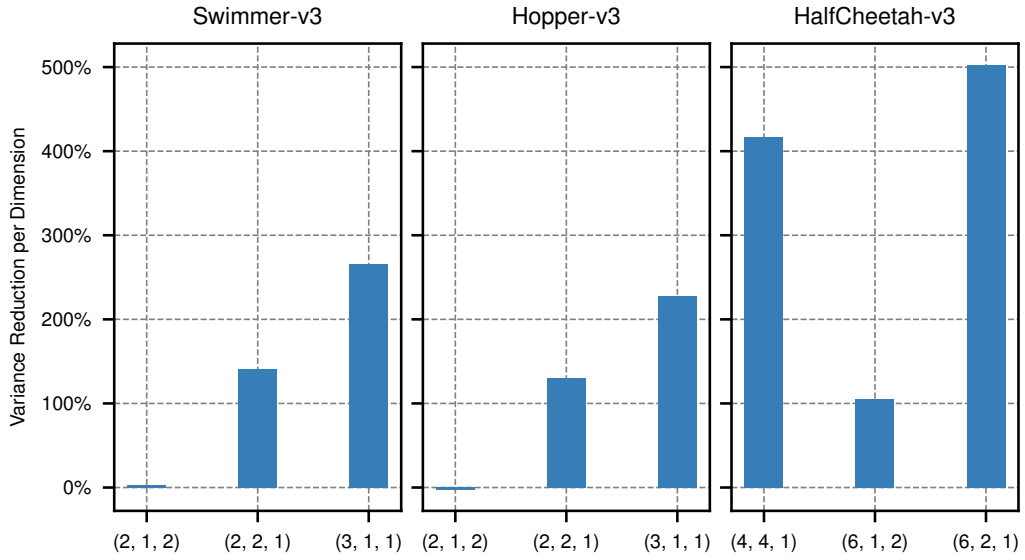
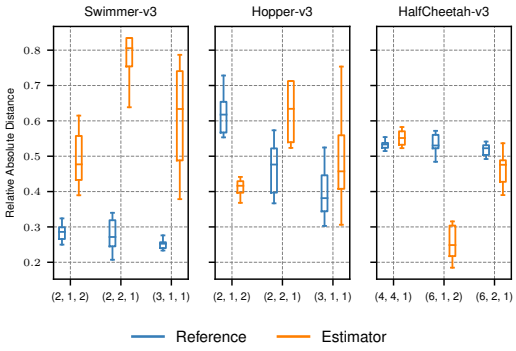
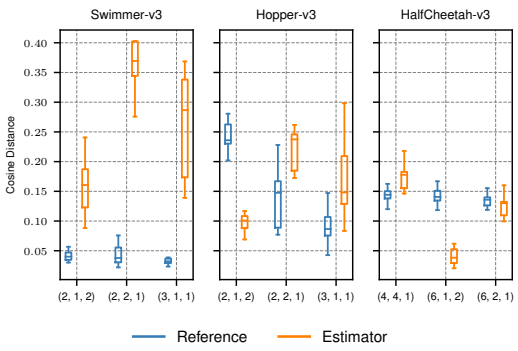


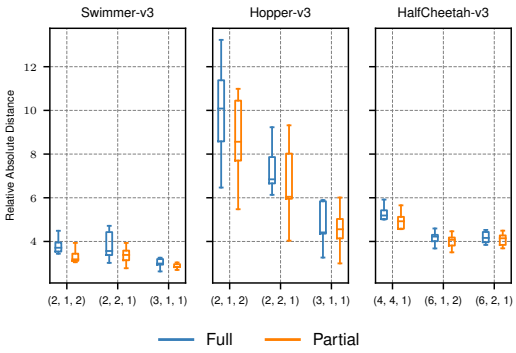
Figure 5.17.: Variance reduction of at the chosen dimensions for the partial gradient relative to an antithetic SF estimator with an evaluation budget of $2(N + MK)$. The tuple (N, K, M) consists of the number of antithetic SF samples N for the full gradient, the number of MVD dimensions K used for the partial gradient, and the number of MVD samples per dimension M.



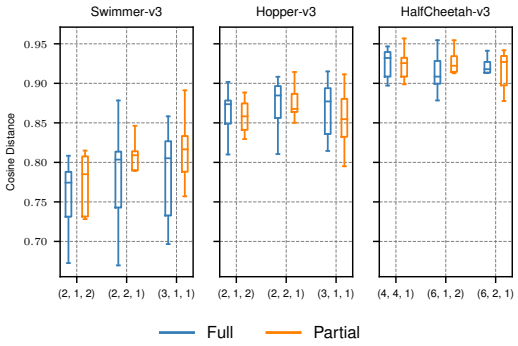
(a) Relative absolute distances $d_{||\cdot||}$ between two consecutive reference gradients and relative distances between estimated full and partial gradients.



(b) Cosine distances $d_{\sphericalangle}$ between two consecutive reference gradients and cosine distances between estimated full and partial gradients.



(c) Relative absolute distances $d_{||\cdot||}$ of full and partial to their respective reference gradients.



(d) Cosine distances $d_{\sphericalangle}$ of full and partial to their respective reference gradients

Figure 5.18.: Comparison of gradient distances shown as box plots. Each box depicts statistics of the distance metric obtained from 10 evaluation points. The upper row shows distances between consecutive gradients of the LOCU estimator and consecutive reference gradients. The bottom row shows distances of the full and partial gradient to their respective reference gradient. The x-axis shows configurations of (N, K, M).

6. Discussion

In this chapter, we interpret the results of the experiments from chapter 5 and highlight the implications for utilizing the MVD for episodic policy search.

6.1. Test Functions and Episodic Setting

For black-box test functions, the variance of the MVD estimator is lower than an antithetic SF estimator using the same budget of evaluations. However, experiments in the episodic setting have shown that the advantage of the MVD regarding its variance is dampened. Here, considering the same budget of evaluations, the antithetic SF estimator achieves lower variance. Nevertheless, the MVD estimator results in similar variance levels as the SF with a fraction of the SF samples.

The contrast between the test function and episodic setting could be explained in the horizon length and random initial conditions of the episodic task. Previously, Vemula et al. have indicated that both factors largely impact the performance of gradient-based episodic policy-search algorithms [41]. Their research, however, focused on the comparison between action and parameter space exploring policy-search methods. Therefore, additional research should be conducted to compare the influence of both factors on the variance of different gradient estimators.

Research comparing the SF to the MVD estimator in the use case of approximate Bayesian inference has also shown an improved variance of the SF estimator when the function evaluations are limited to be the same [34]. This finding is in line with our results and suggests a lower variance of the SF when using the same evaluation budget in certain use cases.

6.2. Convex Combination

When examining a convex combination of SF and MVD estimators, we have experienced an increase of variance for the CCMVD compared to the antithetic SF estimator with the same evaluation budget. However, variance is reduced at the subset of selected dimensions.

Even though the variance is reduced along selected dimensions, it depends on the configuration of base samples N, the number of dimensions K, and the number of additional samples per dimension M. A low N results in high variance for the whole estimator, while a low M yields high variance at the selected dimensions. Overall, it is challenging to select N, K, and M to achieve low variance for the whole estimator and at the selected dimensions while maintaining the fixed function evaluation budget of $2(N + KM)$. Using more than one MVD sample per dimension results in a variance reduction in most cases. However, when the number of selected dimensions is increased, the combined estimator CCSF, where the MVD is replaced with another SF estimator, often gains an advantage. Maintaining the same evaluation budget and increasing selected dimensions allows the CCSF estimator to take more samples and achieve lower variance. Given an increased variance of the combined estimators over an antithetic SF baseline and the higher variance reduction of the CCSF in the case of selecting multiple dimensions, we recommend using a plain antithetic SF estimator with increased samples in most cases.

Our results also indicate a significant influence of the inverse-variance weighting for the convex combination. Applying the replacement strategy instead of the weighting scheme, both CCSF, and CCMVD significantly lower variance. While inverse-variance weighting can improve variance reduction, the challenge of utilizing the correct configuration of (N, K, M) remains.

How the dimensions for MVD computation are selected also influences the performance of the convex combination estimator. Selection strategies based on an estimated variance outperform a random selection on all the tested environments. The improvement of variance-based selections over random selection implies a subset of particularly relevant dimensions for the optimization process that can be identified through the variance of the estimated gradient. A subspace where relevant gradient directions are located is also reported in [18, 7]. However, our selection scheme restricts this subspace to one spanned by the standard basis vectors. Overall, our results indicate that improved dimension selection also yields improved performance. However, the increase of estimator variance when applying the convex combination could explain the improvement in a limited number of environments.

6.3. Locally Updated Gradient

One interesting result we observed for the LOCU estimator is the decrease of performance for the variance-based selection schemes. A possible explanation for this finding is using a replacement strategy instead of the inverse-variance weighting for the LOCU estimator. As explained previously, the weighting scheme considerably influences variance reduction. As observed previously, utilizing the replacement strategy leads to higher variance at the chosen dimensions. Since the selection strategy aims to select the most relevant dimensions, we potentially increase the variance along those relevant dimensions, which is more detrimental than randomly selecting dimensions. Such a case is observed for the HalfCheetah-v3 and Walker2d-v3 environments, where random selection leads to better performance than other strategies.

Results on the gradient distances for the LOCU estimator indicate that consecutive gradients remain in proximity, and our initial assumption of a locally smooth gradient is justified for the environments we consider. However, such an assumption strongly depends on many factors, such as the learning rate and the environment used. Furthermore, we have shown that the full gradient and the partial gradient, which reuses previous gradient information, deviate from a reference gradient in a comparable range. This finding suggests that despite reusing incorrect gradient information from previous iterations, the partial gradient remains close to the true gradient due to the local smoothness of the gradients.

Even though the general idea of saving samples through correcting the previous gradient at specific dimensions seems to be validated, improvements are observed only for one environment. We hypothesize that this result is due to an increased variance due to using a replacement scheme instead of inverse-variance weighting. As mentioned, variance reduction with the inverse-variance weighting is more efficient than a replacement strategy. Therefore further improvement can be expected when the weighting scheme is extended to the LOCU estimator using a running average variance estimate similar to the variance-based selection strategies.

7. Conclusion

In this thesis, we aimed to explore applications of the measured-valued derivative for episodic policy search. For this purpose, we have compared the measure-valued estimator to the widely-used score function estimator and extended both with heuristics from state-of-the-art episodic policy-search algorithms. To mitigate the growth of computational complexity with parameter dimensions for the measure-valued derivative, we investigated two approaches to combine score function and measure-valued estimators.

In general, we have found that variance is more efficiently reduced with the measure-valued estimator when only the sample size is considered. With the combined estimator, variance reduces for the subset of chosen dimensions, but variance increases for the overall estimator. The increase of required evaluations with parameter dimensions represents a major problem. We notice an advantage for a combined estimator that replaces the measure-valued derivative with the score function with increasing selected dimensions, indicating a diminishing advantage with more selected dimensions. Furthermore, combining both estimators with a more informed choice of dimensions and combination weights demonstrates improvements or equal performance on the studied environments.

Reusing gradients from previous iterations presents another way to mitigate the growth of computational complexity of the measure-valued derivative. We have found this approach leads to improvements only on a limited set of environments. Despite of this, our results suggest that partial gradients that are locally corrected with the measure-valued estimator from previously estimated gradients do not show larger deviations than a fully estimated gradient.

7.1. Future Research

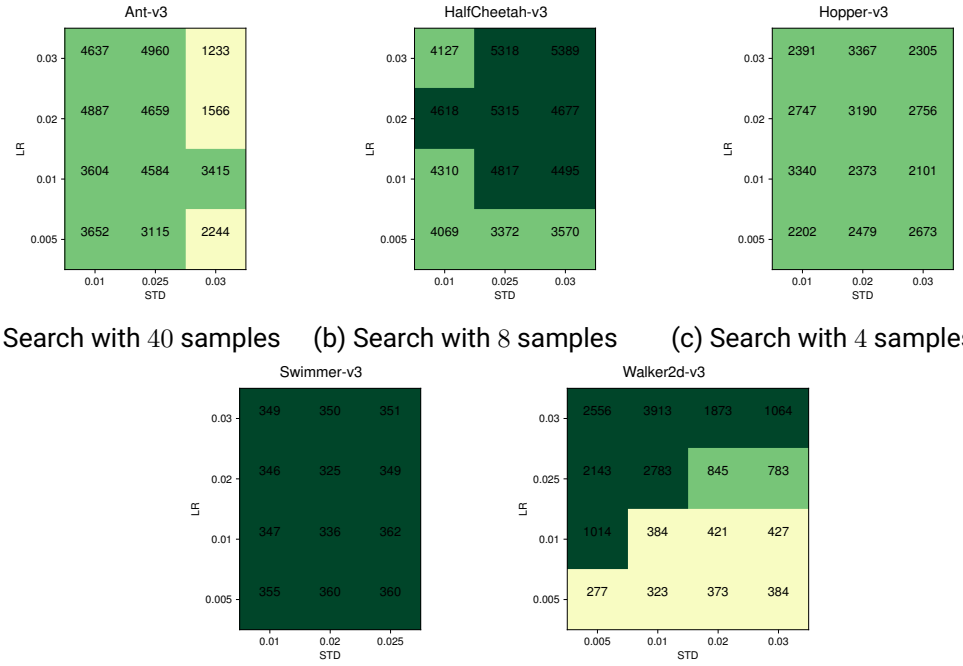
There is a variety of techniques for variance reduction for Monte-Carlo estimators. We have utilized coupling to reduce variance for the measure-valued derivative in our work.

Other reduction methods such as control variates or Rao-Blackwellization can be explored for future research.

During our comparison of the measure-valued derivative with the score function, we noticed a striking similarity between the antithetic score function estimator and the measure-valued derivative. The similarity is particularly evident for gradient estimation of the Gaussian mean. Therefore, an investigation into the source of this similarity promises further insight into the measure-valued derivative.

The main problem with the measure-valued derivative is the per-dimension gradient computation. In particular, the gradient estimation is restricted to gradients with respect to the standard basis. Transforming the basis into a new coordinate frame and estimating the gradients with respect to the transformed basis is an interesting future avenue. A similar idea is to compute a directional derivative with the measure-valued derivative along an arbitrary direction. This approach was introduced by Flynn et al. [12] for the measure-valued derivative and can be combined with methods to find relevant subspaces for black-box optimization [7, 37] to compute gradients along vectors spanning the relevant subspace.

A. Grid Search Results



(a) Search with 40 samples (b) Search with 8 samples (c) Search with 4 samples

(d) Search with 4 samples (e) Search with 60 samples

Figure A.1.: Grid search result for antithetic SF with state and reward normalization, which is equivalent to ARS [25]. Except for Swimmer-v3, the number of samples for each environment were transferred from Mania et al. [25]. Shown is the averaged end reward across 5 seeds.

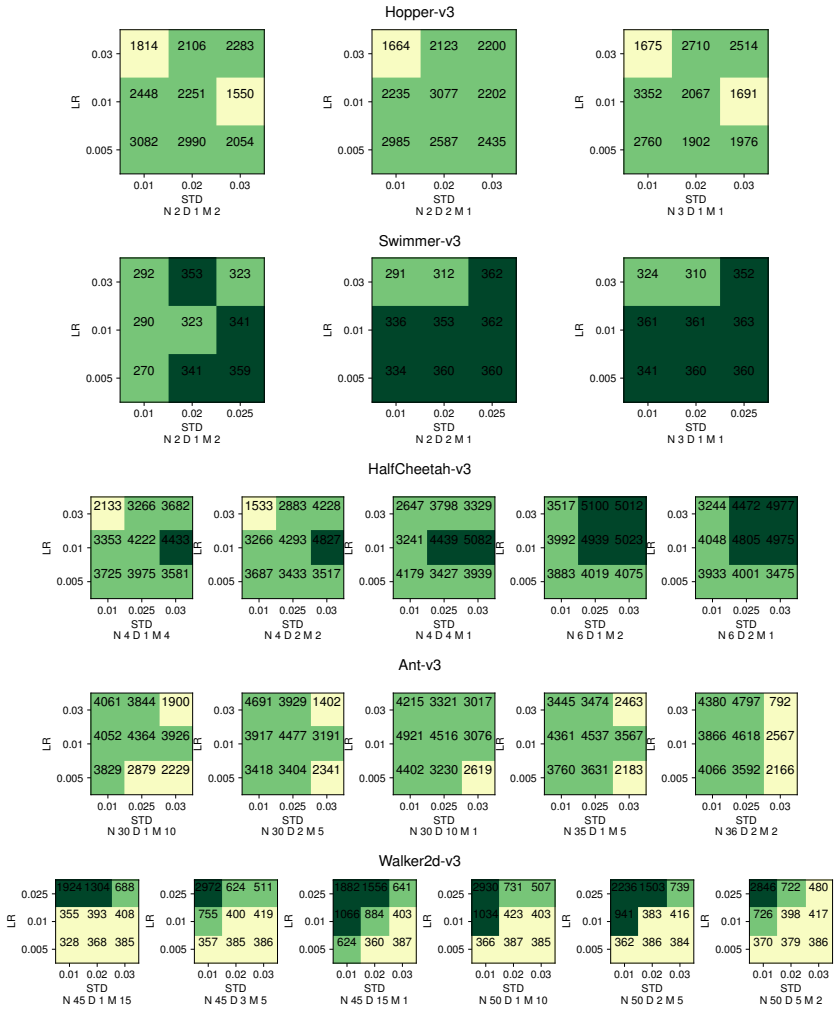


Figure A.2.: Grid-search result for CCMVD with selection strategy V2 and inverse-variance weighting. Shown is the averaged end reward across 5 seeds.

B. Hyperparameters

B.1. Black-box Test Functions

Test Function	Learning Rate	Standard Deviation
Quadratic	$1e{-3}$	0.1
Styblinski	$1e{-4}$	0.1
Rosenbrock	$1e{-5}$	0.1

Table B.1.: Table of hyperparameters used for the antithetic score-function/ARS baseline in chapter 5. Learning rate and standard deviation are obtained from the configuration of the grid-search in A.1.

B.2. Convex Combination

B.2.1. SF Hyperparameters

Environment	Learning Rate	Standard Deviation
Swimmer-v3	0.01	0.025
Hopper-v3	0.02	0.03
HalfCheetah-v3	0.03	0.03
Walker2d-v3	0.03	0.01
Ant-v3	0.03	0.025

Table B.2.: Table of hyperparameters used for the antithetic score-function/ARS baseline in chapter 5. Learning rate and standard deviation are obtained from the configuration of the grid-search in A.1.

B.2.2. CCMVD Hyperparameters

Environment	Learning Rate	Standard Deviation	Warm-Up	Covariance Decay
Swimmer-v3	0.01	0.025	1	0.9
Hopper-v3	0.01	0.01	5	0.9
HalfCheetah-v3	0.03	0.025	5	0.9
Walker2d-v3	0.025	0.01	5	0.9
Ant-v3	0.01	0.01	5	0.9

Table B.3.: Table of hyperparameters used for the CCMVD experiments in chapter 5. Learning rate and standard deviation are obtained from the configuration of the grid-search in A.2. LOCU estimator and CCMVD share the same hyperparameters.

C. Selection Strategy Results

C.1. CCMVD Results

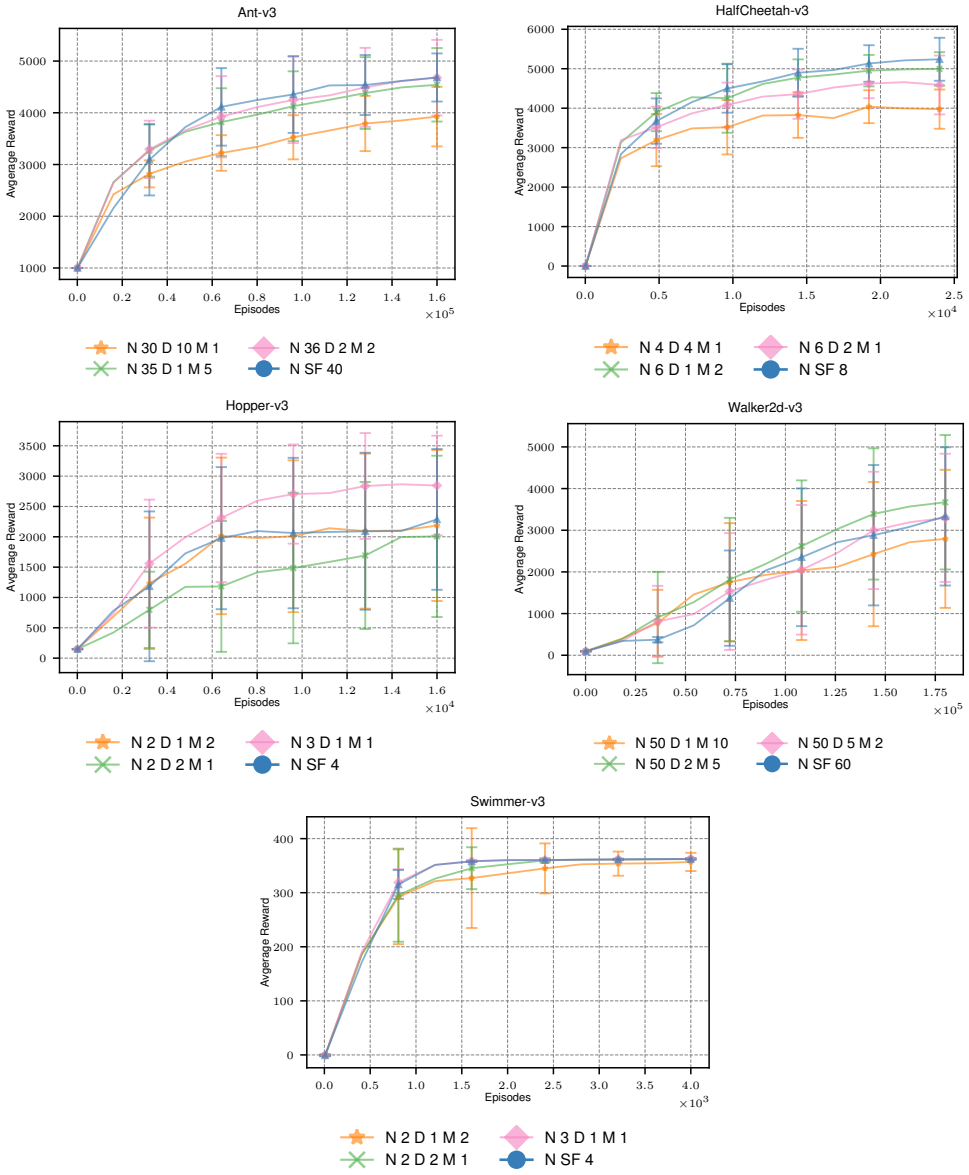


Figure C.1.: Average reward for the Mujoco environments selection based on variance estimation V2. The tuple (N, K, M) consists of the number of antithetic SF samples N, the number of dimensions K and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

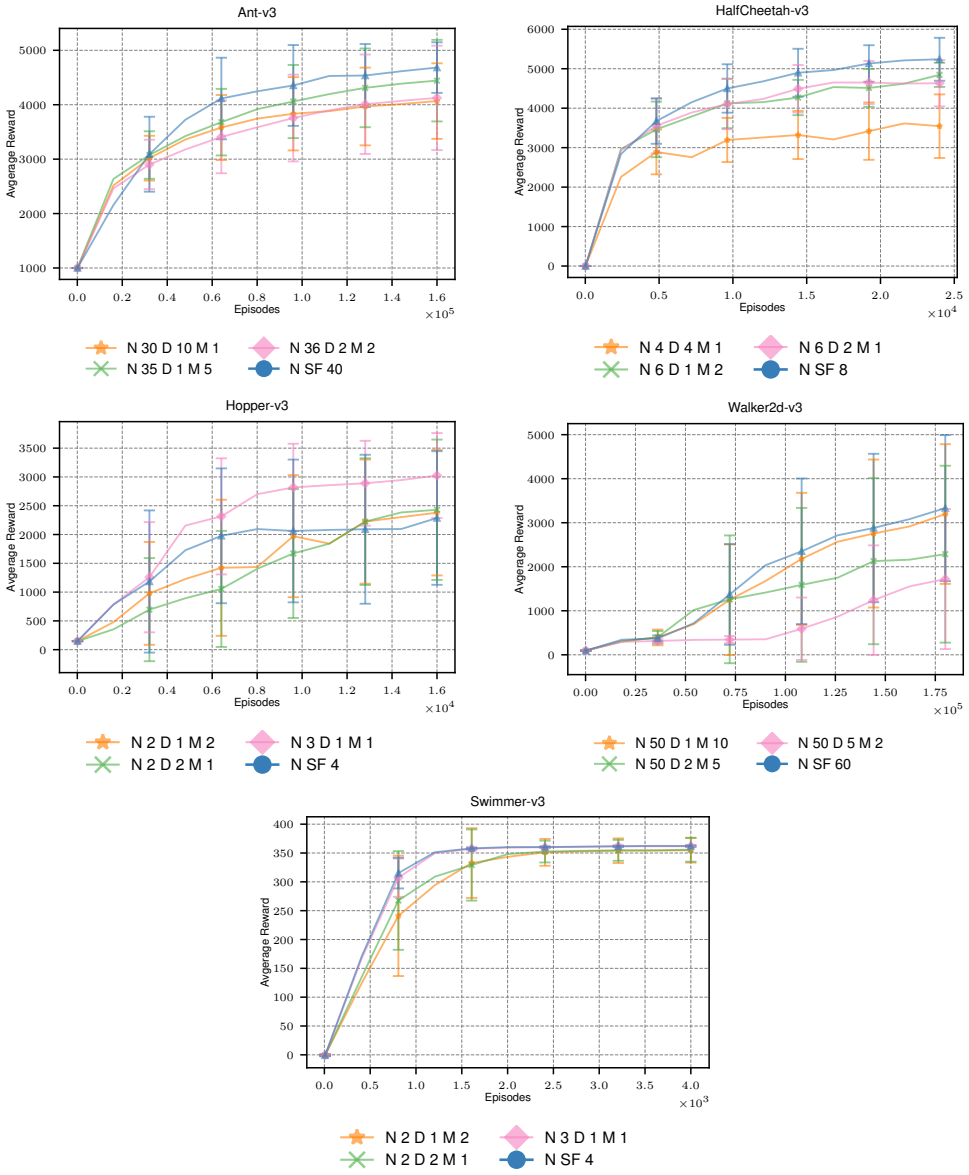


Figure C.2.: Average reward for the Mujoco environments selection based on variance estimation V1. The tuple (N, K, M) consists of the number of antithetic SF samples N, the number of dimensions K and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

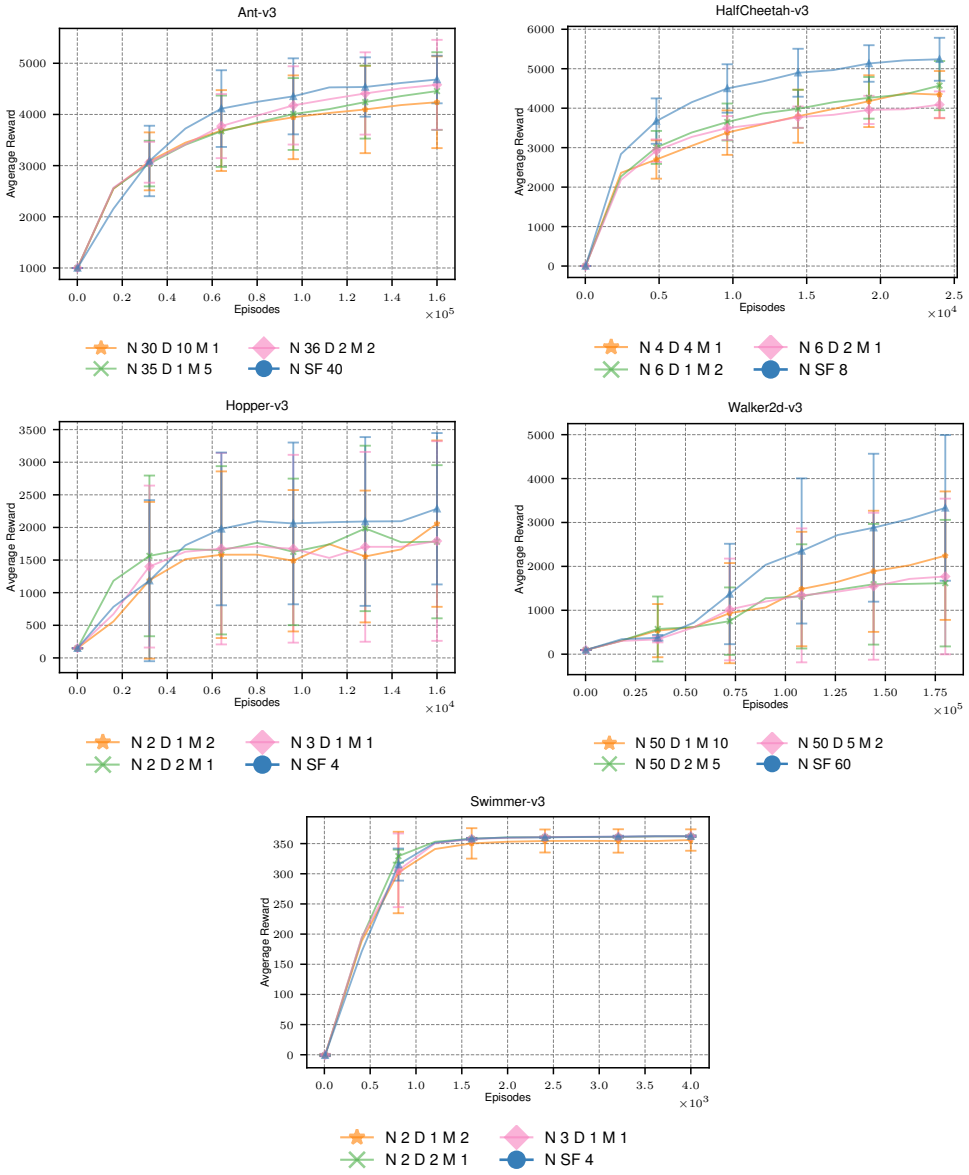


Figure C.3.: Average reward for the Mujoco environments selection based on uniform random sampling. The tuple (N, K, M) consists of the number of antithetic SF samples N, the number of dimensions K and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

C.2. LOCU Results

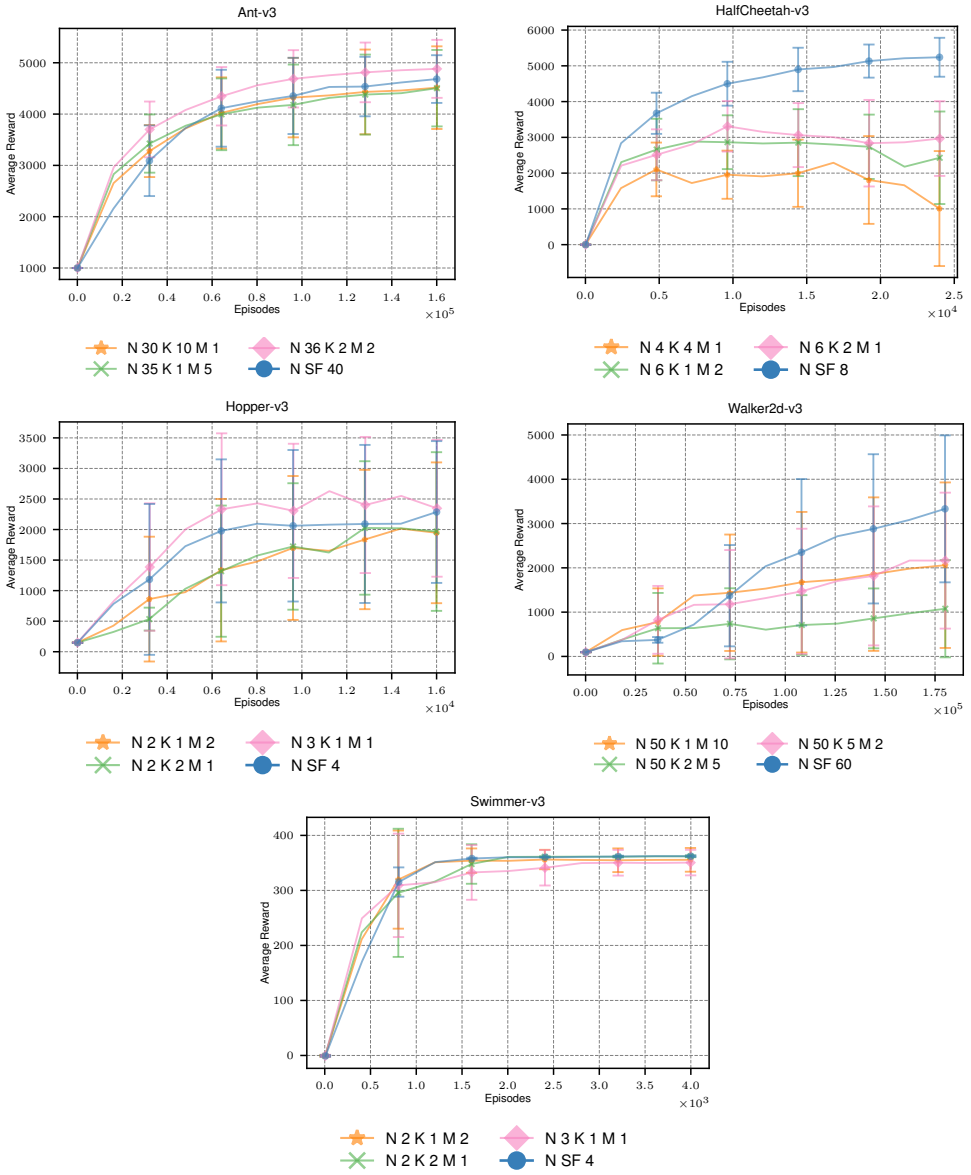


Figure C.4.: Average reward for the Mujoco environments with dimension selection based on uniform variance estimation V2. The tuple (N, K, M) consists of the number of antithetic SF samples N for the full gradient, the number of MVD dimensions K used for the partial gradient, and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

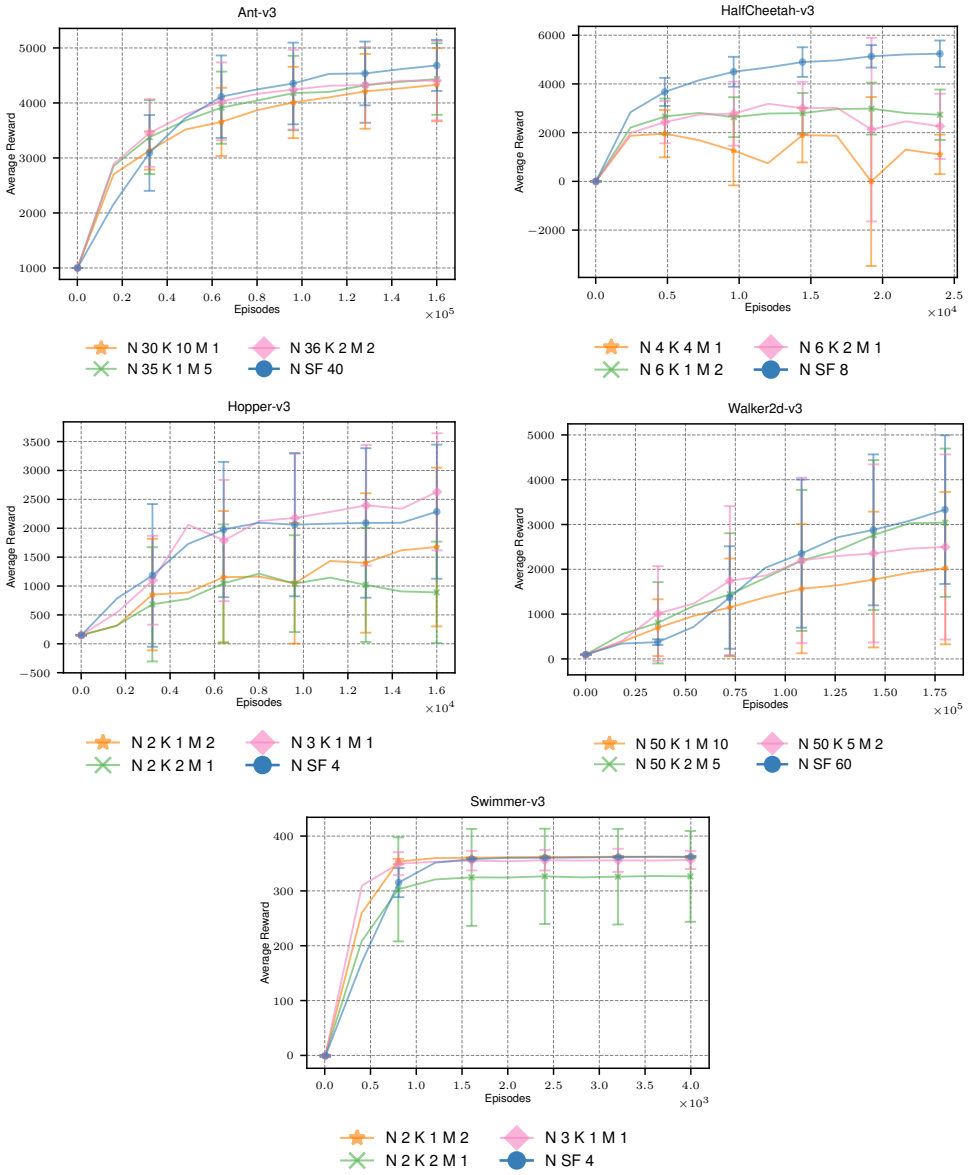


Figure C.5.: Average reward for the Mujoco environments with dimension selection based on variance estimation V1. The tuple (N, K, M) consists of the number of antithetic SF samples N for the full gradient, the number of MVD dimensions K used for the partial gradient, and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

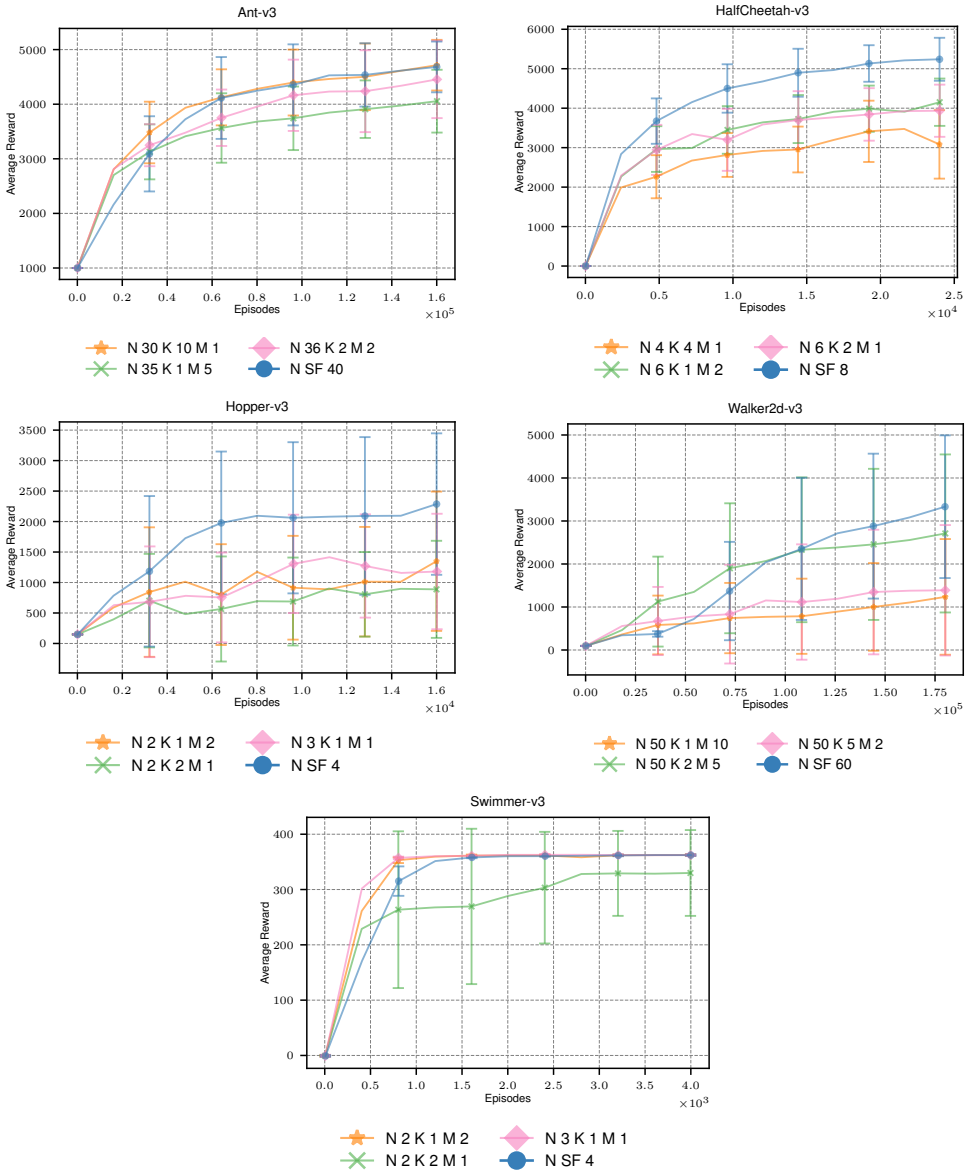


Figure C.6.: Average reward for the Mujoco environments with dimension selection based on uniform random sampling. The tuple (N, K, M) consists of the number of antithetic SF samples N for the full gradient, the number of MVD dimensions K used for the partial gradient, and the number of MVD samples per dimension M. The parameter tuples were chosen according to the grid search A. The rewards are averaged over 10 seeds.

Bibliography

[1] Lukas Balles and Philipp Hennig. “Dissecting Adam: The Sign, Magnitude and Variance of Stochastic Gradients”. In: *ICML*. 2018.

[2] Sujay Bhatt, Alec Koppel, and Vikram Krishnamurthy. “Policy Gradient using Weak Derivatives for Reinforcement Learning”. In: *2019 53rd Annual Conference on Information Sciences and Systems (CISS)* (2019), pp. 1–3.

[3] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.2.5. 2018. URL: http://github.com/google/jax.

[4] Greg Brockman et al. *OpenAI Gym*. 2016. eprint: arXiv:1606.01540.

[5] Lars Buesing, Théophane Weber, and Shakir Mohamed. “Stochastic Gradient Estimation With Finite Differences”. In: 2016.

[6] João Carvalho et al. “An Empirical Analysis of Measure-Valued Derivatives for Policy Gradients”. In: *2021 International Joint Conference on Neural Networks (IJCNN)* (2021), pp. 1–10.

[7] Krzysztof Choromanski et al. “From Complexity to Simplicity: Adaptive ES-Active Subspaces for Blackbox Optimization”. In: *NeurIPS*. 2019.

[8] Krzysztof Choromanski et al. “Structured Evolution with Compact Architectures for Scalable Policy Optimization”. In: *ArXiv* abs/1804.02395 (2018).

[9] Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. now, 2013. URL: https://ieeexplore.ieee.org/document/8186816.

[10] Jörn Dunkel and Stefan Weber. “Efficient Monte Carlo methods for convex risk measures in portfolio credit risk models”. In: *2007 Winter Simulation Conference* (2007), pp. 958–966.

[11] JL Fleiss. “Review papers : The statistical basis of meta-analysis”. In: *Statistical Methods in Medical Research* 2.2 (1993). PMID: 8261254, pp. 121–145. DOI: 10.1177/096228029300200202. eprint: https://doi.org/10.1177/096228029300200202. URL: https://doi.org/10.1177/096228029300200202.

[12] Thomas Flynn and Felisa V'azquez-Abad. "A simultaneous perturbation weak derivative estimator for stochastic neural networks". In: *Comput. Manag. Sci.* 16 (2019), pp. 715–738.

[13] Michael C. Fu. "Stochastic Gradient Estimation". In: 2005.

[14] John Geweke. "Antithetic acceleration of Monte Carlo integration in Bayesian inference". In: *Journal of Econometrics* 38.1 (1988), pp. 73–89. ISSN: 0304-4076. DOI: https://doi.org/10.1016/0304-4076(88)90027-9. URL: https://www.sciencedirect.com/science/article/pii/0304407688900279.

[15] Paul Glasserman. "Monte Carlo Methods in Financial Engineering". In: 2003.

[16] Will Grathwohl et al. "Backpropagation through the Void: Optimizing control variates for black-box gradient estimation". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=SyzKd1bCW.

[17] Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. "Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning". In: *J. Mach. Learn. Res.* 2004.

[18] Guy Gur-Ari, Daniel A. Roberts, and Ethan Dyer. "Gradient Descent Happens in a Tiny Subspace". In: *CoRR* abs/1812.04754 (2018). arXiv: 1812.04754. URL: http://arxiv.org/abs/1812.04754.

[19] Eric Jang, Shixiang Shane Gu, and Ben Poole. "Categorical Reparameterization with Gumbel-Softmax". In: *ArXiv* abs/1611.01144 (2017).

[20] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114 (2014).

[21] Charles R. Leake. "Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method". In: *Journal of the Operational Research Society* 45 (1994), pp. 960–961.

[22] Daniel Lévy and Stefano Ermon. "Deterministic Policy Optimization by Combining Pathwise and Score Function Estimators for Discrete Action Spaces". In: *ArXiv* abs/1711.08068 (2018).

[23] Chunyuan Li et al. "Measuring the Intrinsic Dimension of Objective Landscapes". In: *International Conference on Learning Representations*. 2018.

[24] Niru Maheswaranathan et al. “Guided evolutionary strategies: augmenting random search with surrogate gradients”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 4264–4273. URL: https://proceedings.mlr.press/v97/maheswaranathan19a.html.

[25] Horia Mania, Aurelia Guy, and Benjamin Recht. “Simple random search of static linear policies is competitive for reinforcement learning”. In: *NeurIPS*. 2018.

[26] “Measure-Valued Differentiation for Stationary Markov Chains”. In: *Mathematics of Operations Research* 31.1 (2006), pp. 154–172. ISSN: 0364765X, 15265471. URL: http://www.jstor.org/stable/25151713.

[27] Shakir Mohamed et al. “Monte Carlo Gradient Estimation in Machine Learning”. In: *J. Mach. Learn. Res.* 21 (2020), 132:1–132:62.

[28] Yurii Nesterov and Vladimir G. Spokoiny. “Random Gradient-Free Minimization of Convex Functions”. In: *Foundations of Computational Mathematics* 17 (2017), pp. 527–566.

[29] Paavo Parmas. “Total stochastic gradient algorithms and applications in reinforcement learning”. In: *NeurIPS*. 2018.

[30] G. Ch. Pflug. “Sampling Derivatives of Probabilities”. In: *Computing* 42.4 (Oct. 1989), pp. 315–328. ISSN: 0010-485X. DOI: 10.1007/BF02243227. URL: https://doi.org/10.1007/BF02243227.

[31] Hongyu Ren, Shengjia Zhao, and Stefano Ermon. “Adaptive Antithetic Sampling for Variance Reduction”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 5420–5428. URL: https://proceedings.mlr.press/v97/ren19b.html.

[32] Herbert Robbins and Sutton Monro. “A Stochastic Approximation Method”. In: *The Annals of Mathematical Statistics* 22.3 (1951), pp. 400–407. DOI: 10.1214/aoms/1177729586. URL: https://doi.org/10.1214/aoms/1177729586.

[33] John W. Roberts. “Motor Learning on a Heaving Plate via Improved-SNR Algorithms”. In: 2009.

[34] Mihaela Rosca and Michael Figurnov. *Measure-Valued Derivatives for Approximate Bayesian Inference*. 2019.

[35] Tim Salimans et al. “Evolution Strategies as a Scalable Alternative to Reinforcement Learning”. In: *ArXiv* abs/1703.03864 (2017).

[36] Frank Sehnke et al. “Parameter-exploring policy gradients”. In: *Neural networks : the official journal of the International Neural Network Society* 23 4 (2010), pp. 551–9.

[37] Ozan Sener and Vladlen Koltun. “Learning to Guide Random Search”. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=B1gHokBKwS.

[38] Yunhao Tang, Krzysztof Choromanski, and Alp Kucukelbir. “Variance Reduction for Evolution Strategies via Structured Control Variates”. In: *ArXiv* abs/1906.08868 (2020).

[39] Emanuel Todorov, Tom Erez, and Yuval Tassa. “MuJoCo: A physics engine for model-based control”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

[40] F. Vázquez-Abad. “A Course on Sensitivity Analysis for Gradient Estimation of Des Performance Measures”. In: 2000.

[41] Anirudh Vemula, Wen Sun, and J. Andrew Bagnell. “Contrasting Exploration in Parameter and Action Space: A Zeroth-Order Optimization Perspective”. In: *ArXiv* abs/1901.11503 (2019).

[42] Marc Aurel Vischer, Rupert Lange, and Henning Sprekeler. “On Lottery Tickets and Minimal Task Representations in Deep Reinforcement Learning”. In: *ArXiv* abs/2105.01648 (2021).

[43] Daan Wierstra et al. “Natural Evolution Strategies”. In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)* (2008), pp. 3381–3387.

[44] Ronald J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Machine Learning* 8 (2004), pp. 229–256.

[45] Mike Wu, Noah D. Goodman, and Stefano Ermon. “Differentiable Antithetic Sampling for Variance Reduction in Stochastic Variational Inference”. In: *AISTATS*. 2019.