Policy Gradient Decomposition via Agnostic Learning of Variable Partitions

Anonymous Author(s)
Affiliation
Address
email

Abstract

Policy gradient (PG) methods on high-dimensional control are widely known to be difficult due to the large variance of the PG estimators. Recent studies on the Rao-Blackwell theorem show that it is efficient to reduce the variance if the PG estimator can be decomposed over the action space. However, previous studies rely on assumptions on the action space and heuristic algorithms without theoretical guarantee. To circumvent these issues, we propose algorithmic approaches that learn the action space partition agnostically, without imposing a stringent assumption. Our algorithms give factor-4 approximate optimality of the PG decomposition via learning the variable partition of the high-dimensional action. Empirical studies demonstrate the performance improvements on synthetic settings and OpenAI Gym's MuJoCo continuous control tasks.

1 Introduction

Policy gradient (PG) methods [Wil92, SMSM00] have been widely applied to various challenging problems including video games [MBM$^+$16], robotics [LFDA16], and continuous control tasks [SWD$^+$17, RLTK17, LHP$^+$15]. It estimates the gradient of the expected cumulative reward directly from the rollouts of the agent trajectories. A major challenge of PG is the high variance of the gradient estimator. Since its inception, the community has been focusing on improving the PG estimator via a variety of variance reduction techniques [Wil92, SMSM00, MBM$^+$16, KWY18, GLG$^+$16, LFM$^+$18, GCW$^+$18, TBG$^+$18]. When dealing with high-dimensional continuous action spaces, methods based on the Rao-Blackwell theorem (RB) [CR96] demonstrate significant efficiency [WRD$^+$18, LW18, RGB14].

The key rationale of RB is to partition the action space into multiple subspaces, and estimate the conditional expectation respectively on each subspace. As shown in [LW18], it is unbiased to decompose the PG estimator if the advantage function $A(s, a)$ can be partitioned into the direct sum $A(s, a_{(1)}) + \cdots + A(s, a_{(k)})$, where $a_{(j)}$ is the action subspace. Previous works [WRD$^+$18, LW18] have assumed the partition structures of the advantage function, which do not hold in general. Our aim is to learn this partition $\{a_{(j)}\}$ agnostically to elicit RB in the PG estimator.

We first rigorously formulate the decomposition of the action space. David et al. [DDG$^+$17] have shown that a boolean function f has a direct sum decomposition $f(X, Y) = g(X) + h(Y)$ if and only if the dependence score $D = f(X, Y) - f(X', Y) - f(X, Y') + f(X', Y')$ is zero for all X, X', Y, Y'. We naturally extend this D to functions of real vectors and extend from testing $f = g + h$ to learning such g and h. Our analysis shows that under the Monte-Carlo sampling of X, X', Y, Y', this estimator D is robust to the error. Namely, even if there exists a decomposition error

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

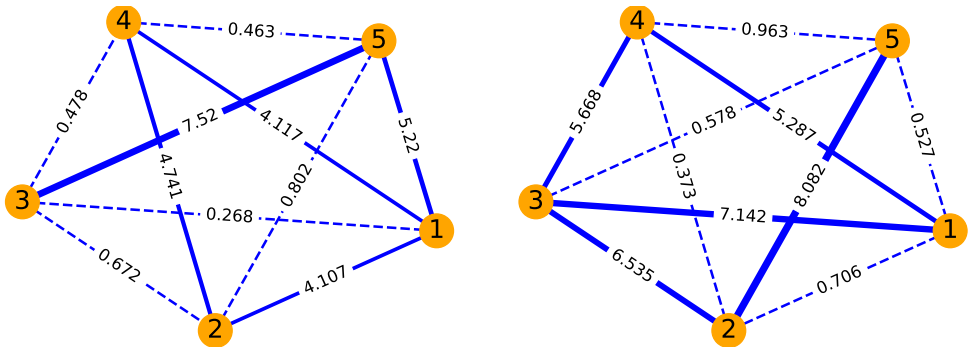

Figure 1: Illustrative examples of $a \in \mathbb{R}^5$. Each node represents one coordinate. Solid edges represent pairs of nodes with dependency where the line width scales with the dependency measure; the numbers on the edges are the exact dependency measure; dashed lines represent nodes without dependency; the measure on the dashed line is nonzero which is analogous to the noise in the measure.

Table 1: Comparisons of our algorithms with previous ones

PG estimator	Variance	Assumptions	Partitioning	Guarantees	Limits
A2C [MBM$^+$16]	CV	-	-	-	-
Wu et al. [WRD$^+$18]	CV & RB	yes	fully	no	$k = m$
Li and Wang [LW18]	CV & RB	yes	greedy	no	**no**
PE (ours)	CV & RB	**no**	greedy	factor-$\mathcal{O}(kn^2)$	**no**
SM (ours)	CV & RB	**no**	**optimal**	**factor-4**	$k = 2$

of f, the characterization still holds approximately. Thus, we can find the direct sum decomposition of $A(s, a)$ and the corresponding action space partition $\{a_{(j)}\}$ by minimizing the expectation of D.

It suffices to find an efficient way to minimize the expectation of D. We propose two algorithms to achieve this. Our first algorithm is a greedy approach that leverages pairwise information. The algorithm builds a weighted complete graph at the beginning, where each node corresponds to one action coordinate and each edge is assigned the weight as the dependence score over the two coordinates. The algorithm removes edges with the smallest weight until there are exactly k connected components. We show that the output partition is a factor-$\mathcal{O}(kn^2)$ approximation of the best possible one, where n and k are the dimension and partition size, respectively. Our second algorithm works only when $k = 2$, but it improves the optimality to almost factor-4. The main insight behind this algorithm is that the dependence score is a submodular function of the bipartition. We can therefore find the bipartition efficiently by existing studies of submodular minimization algorithms [GLS81, Sch00, IFF01]. We then discuss approaches to extend the algorithm to $k > 2$.

We present empirical results to corroborate our theoretical guarantees and to demonstrate the efficiency of the proposed algorithms. We evaluate our algorithms on both synthetic settings and a variety of high-dimensional continuous control tasks from OpenAI Gym. The algorithms consistently and significantly outperform the baselines, and have the best variance reduction among the RB based methods. In practical, both the algorithms are efficient and do not incur a notable computational cost.

Our contributions are summarized as follows:

1. We propose decomposed PG. It elicits RB on the PG estimator and strictly reduces the variance of the estimator. The algorithm learns the PG decomposition agnostically, thus do not require assumptions on the action space structure.
2. We rigorously formulate the problem of partitioning the action space using a novel dependence score D. We show in **Theorem 4** that D is robust to error, namely, $\delta^p(\boldsymbol{X}, \boldsymbol{Y}) \leq D(\boldsymbol{X}, \boldsymbol{Y}) \leq 4\delta^p(\boldsymbol{X}, \boldsymbol{Y})$, where $\delta^p(\boldsymbol{X}, \boldsymbol{Y})$ is the ground truth of the partition error.
3. We propose a greedy-based algorithm such that it outputs a k-partition $\mathcal{P}$ efficiently. The partition error $\delta^p(\mathcal{P})$ is proved in **Theorem 6** to be $\mathcal{O}(kn^2)$.
4. We show that the expected error $\mathbb{E}[D]$ can be minimized efficiently with respect to the bipartition $\boldsymbol{X}, \overline{\boldsymbol{X}}$, as $\mathbb{E}[D(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$ is a submodular function, shown in **Theorem 7**.

Related works. [Kos18] and [WRD+18] decompose the action space completely, where each subspace includes exactly one coordinate. Though the method demonstrates variance reduction on a variety of tasks, it relies on the assumption that all variables are independent with each other among the action space. Under a relatively weaker assumption, [LW18] uses the Hessian value to measure the pairwise dependency between variables and subsequently proposes a greedy-based algorithm. The algorithm depends on the heuristic that two variables are likely to be independent if the corresponding Hessian element is close to zero. While the Hessian element of two independent variables is zero, the converse does not hold. The comparisons between existing methods and our algorithm via pairwise estimation (PE) and submodular minimization (SM) are summarized in Table 1 (CV in the table abbreviates Control Variates).

Furthermore, previous approaches do not find the globally optimal partition, as they consider only local dependencies between pairs of coordinates. We show this local optimality on two examples of 5 coordinates in Figure 1. Both examples demonstrate a one-step MDP on $a \in \mathbb{R}^5$ where the objective is quadratic $a^T H a$. The edge between node i and j is weighted by H_{ij}, and in this case partitioning A is equivalent to finding a minimum cut of the graph. In example 1 (left) the optimal partition is $(1, 2, 4)$, $(3, 5)$ with **7.90** partition error. But the greedy algorithm yields the partition $(1, 3, 5)$, $(2, 4)$ with **11.66** partition error. Similarly, in example 2 (right) the optimal partition is $(1, 3, 4)$, $(2, 5)$ with error **9.68** while the greedy algorithm finds $(1, 2, 3, 5)$, (4) with error **12.29**. These examples are further explained in the experiments and in Appendix E.

2 Variance Reduction of Policy Gradient

2.1 Preliminaries on PG Methods and PG Decomposition

We consider policy optimization in the discrete-time Markov decision process (MDP) setting, denoted as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \rho_0, \beta)$. That includes $\mathcal{S} \in \mathbb{R}^m$ the m dimensional state space, $\mathcal{A} \in \mathbb{R}^n$ the n dimensional action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ the environment transition probability function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, ρ_0 the initial state distribution and $\beta \in [0, 1)$ the unnormalized discount factor. Policy optimization learns a stochastic policy $\pi_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$, which is parameterized by θ, to maximize the expected cumulative reward $J(\theta) = \mathbb{E}_{s\sim\rho_\pi, a\sim\pi}[\sum_{t=0}^{\infty} \beta^t r(s_t, a_t)]$, where $\rho_\pi(s) = \sum_{t=1}^{\infty} \beta^{t-1} \mathbb{P}(s_t = s)$ is the discounted state visitation distribution. Define the action-state value function $Q^\pi(s_t, a_t) = \mathbb{E}_\pi[\sum_{t'\geq t}^{\infty} \beta^{t'-t} r(s_{t'}, a_{t'})|s_t, a_t, \pi]$ to be the expected discounted cumulative reward of policy π at state s_t after taking action a. Also define $V^\pi(s_t) = \mathbb{E}_{a_t\sim\pi(a|s_t)}[Q^\pi(s_t, a_t)]$ as the state-value function and $A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$ the advantage function. When it is clear from the context we omit t and write $J(\theta) = \mathbb{E}_{\pi,\rho_\pi}[r(s, a)]$.

The decomposed policy gradient estimator with action space partition is derived in [WRD+18] and [LW18]. Let the action space be partitioned into $a_{(1)}, \dots a_{(k)}$, where $a_{(j)}$ is the j-th subset of the action coordinates. If the policy function is parameterized by θ, we have

$$\nabla_\theta J(\theta) = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)})], \tag{1}$$

where k is the number of partitions of the action space.

For the rest of the discussion where $a \in \mathbb{R}^n$, we treat the advantage function $A(s, a)$ as a function of n variables. Each variable denotes one coordinate of a. We use bold lowercase letters, e.g. $\boldsymbol{x}$ and $\boldsymbol{y}$, to denote a single variable, and bold uppercase letters, e.g. $\boldsymbol{X}$ and $\boldsymbol{Y}$, to denote sets of variables.

An important observation is that $\nabla_\theta J(\theta)$ is unbiased (we show it later in Proposition 3) if the advantage function can is partitioned into a set of action subspaces. This advantage decomposition property is first introduced in Assumption 1 of [LW18], where it requires $U = 0$ as an assumption. We re-formulate the property as below with a nonzero error term, and thereafter focus on minimizing the error term in our algorithm. In this way, our algorithm learns agnostically the partition and use that partition to find the decomposed policy gradient estimator $\nabla_\theta J(\theta)$.

Definition 1 (Advantage Decomposition)**.** *If the advantage function $A^\pi(s, a)$ can be locally approximated additively by a set of decomposed functions with respect to a, that is $A^\pi(s, a) = A_1^\pi(s, a_{(1)}) + \cdots + A_k^\pi(s, a_{(k)}) + U(s, a)$, we define $U(s, a)$ as the advantage decomposition approximation error.*

2.2 Variance Reduction via Rao-Blackwellization

The following proposition is a standard argument on conditional probabilities, also known as Rao-Blackwellization, which indicates that the decomposed score function estimator has a lower variance than the original one.

Proposition 2 (Conditioning and Rao-Blackwellization)**.** *Let (X, Y) have joint distribution f and let $f(X, Y)$ satisfy $\mathrm{Var}[f(X, Y)] < +\infty$. Define $h(X') = \mathbb{E}[f(X, Y)|X = X']$ for $X, Y \sim \mathbb{P}_f$. Suppose that $X_i, Y_i \sim \mathbb{P}_f$. Then, we have*

$$\mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} h(X_i)] \leq \mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} f(X_i, Y_i)].$$

Combining both the definition of the advantage decomposition and the variance reduction property, we have the following proposition. The key observation to conclude the proposition is the property of the score function $\mathbb{E}_{p(X,Y)}[\nabla \log p(X, Y)] = 0$. We include the full proof in Appendix A.

Proposition 3. *Suppose that the function $f : \mathbb{R}^n \times \to \mathbb{R} \cup \{+\infty\}$ satisfies zero partition error in Definition 1 with two sets of variables $X \in \mathbb{R}^{m_1}$ and $Y \in \mathbb{R}^{m_2}$ (i.e., $f(X, Y) = f_X + f_Y$, $m_1 + m_2 = n$). $P(X, Y)$ is a distribution enjoying the independence structure $P(X, Y) = P(X)P(Y)$ and $\mathbb{E}_{P(X,Y)}[\|f(X, Y)\|^2] < +\infty$. Then, we have*

- *(Unbiased Estimator) In the estimation of $\mathbb{E}_{P(X,Y)}[\nabla \log P(X, Y)f(X, Y)]$,*

$$\mathbb{E}_{P(X,Y)}[\nabla_\theta \log P(X, Y)f(X, Y)] = \mathbb{E}_{P(X)}[\nabla_\theta \log P(X)f_X] + \mathbb{E}_{P(Y)}[\nabla_\theta \log P(Y)f_Y].$$

- *(Variance Reduction) For the two estimators $g(X, Y) = \nabla \log P(X, Y)f(X, Y)$ and $h(X, Y) = \nabla_\theta \log P(X)f_X + \nabla \log P(Y)f_Y$,*

$$\mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} h(X_i, Y_i)] \leq \mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} g(X_i, Y_i)],$$

  where n_s is the number of Monte-Carlo samples.

We substitute $p(X, Y)$ with $\pi(a_{(1)}, a_{(2)}|s)$ and $f(X, Y)$ with $A(a_{(1)}, a_{(2)}, s)$ in Proposition 3 and yield that the variance of $\nabla_\theta J(\theta)$ is strictly less than policy gradient estimator proposed in [MBM$^+$16], [LFM$^+$18], and [GCW$^+$18].

3 Formulation of Direct Sum Decomposition

We rigorously formulate the problem of learning the partition in this section. As pointed out in Definition 1, we consider the general family $F(\boldsymbol{V})$ as a function of a product set. A direct sum decomposition of F is a partition $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ of the set of variables $\boldsymbol{V}$ such that

$$F(\boldsymbol{V}) = F_1(\boldsymbol{X}_1) + \cdots + F_k(\boldsymbol{X}_k) \tag{2}$$

for some functions $F_1, \ldots, F_k$. We are interested in efficient (sublinear-time) approximate algorithms for computing the direct sum decomposition given oracle access to F. In policy gradient, the $F(\cdot)$ function represents the advantage function. But we believe that our problem formulation is sufficiently general to be potentially useful in other contexts.

Given a partition $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ of the variables, the quality of the decomposition with respect to this partition is measured by

$$\delta^p(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) = \min_{F_1,\ldots,F_k} \|F(X_1, \ldots, X_k) - F_1(X_1) - \cdots - F_k(X_k)\|_p,$$

where $\|\cdot\|_p$ is the ℓ^p-norm over the product measure $\|F\| = \mathbb{E}\big[|F(X_1, \ldots, X_k)|^p\big]^{1/p}$. In Sections 4 and 5 the norm index p can be any integer at least 1. In Section 6 $|\cdot|$ is absolute value, and p equals 2. We omit p from the notation when it is clear from context.

The objective is to find an approximation of the best-possible partition, which minimizes the objective

$$\delta_2^p(F) = \min \delta^p(\boldsymbol{X}, \overline{\boldsymbol{X}})$$

where the minimum is taken over all bipartitions $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ of $\boldsymbol{V}$ and $\overline{\boldsymbol{X}}$ denotes the complement of $\boldsymbol{X}$. More generally, we consider partitions into (at least) k nonempty sets, giving rise to the objective value $\delta_k^p(F) = \min \delta^p(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$.

Our algorithms are based on the following dependence estimator inspired by the rank-1 test of [DDG+17]. Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two disjoint sets of variables. The dependence estimator $D_F(\boldsymbol{X}, \boldsymbol{Y})$ is the random variable

$$D_F = F(X, Y, Z) + F(X', Y', Z) - F(X', Y, Z) - F(X, Y', Z)$$

where X, X' are independent samples of the $\boldsymbol{X}$ variable, Y, Y' are independent samples of the $\boldsymbol{Y}$ variable, and Z is a random sample of the remaining variables.

If F decomposes into a direct sum that partitions the $\boldsymbol{X}$ and $\boldsymbol{Y}$ variables then D_F equals zero. Conversely, $\|D_F\|_p$ measures the quality of the approximation in the ℓ^p-norm.

In the analysis it will be convenient to use the notation $F \approx_\delta G$ for $\|F - G\|_p \leq \delta$. The following two facts are immediate:

Fixing: If $F(X, Z) \approx_\delta G(X, Z)$ then there exists a function $\underline{X}(Z)$ such that $F(\underline{X}(Z), Z) \approx_\delta G(\underline{X}(Z), Z)$.

Triangle inequality: If $F \approx_\delta G$ and $G \approx_{\delta'} H$ then $F \approx_{\delta+\delta'} H$.

4 Estimating the Quality of a Bipartition

In this section we show that $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p$ is an approximate estimator for the quality $\delta^p(\boldsymbol{X}, \boldsymbol{Y})$ of a decomposition, namely

$$\delta^p(\boldsymbol{X}, \boldsymbol{Y}) \leq \|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p \leq 4 \cdot \delta^p(\boldsymbol{X}, \boldsymbol{Y}). \tag{3}$$

This corresponds to our second main contribution listed at the end of Section 1. We then state upper and lower bounds on the quality of the sampling approximators for this value. Taken together, we obtain efficient factor $(4 + \varepsilon)$-approximation algorithms (where the error ε is from the estimation of $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p$) for $\delta^p(\boldsymbol{X}, \boldsymbol{Y})$ of a given partition.

Theorem 4. *There is an algorithm that given a bipartition $\boldsymbol{X}, \boldsymbol{Y}$ of the variables and a parameter $\varepsilon > 0$, outputs a value $\hat{D}$ such that*

$$\delta^p(\boldsymbol{X}, \boldsymbol{Y}) \leq \hat{D} \leq 4 \cdot \delta^p(\boldsymbol{X}, \boldsymbol{Y}) + \varepsilon,$$

with probability at least $1 - \gamma$ from $K^p \log(1/\gamma)/\epsilon^{2p}$ queries to F in time linear in the number of queries, where K is an absolute constant.

The proof of inequality (3) is in Claim 9 (completeness) and 10 (soundness). The bound of the estimation error ε is stated in Claim 5 and the proof can be found in Appendix B. It is worth note that the bound has efficient, logarithmic dependence on $1/\gamma$.

The analysis in fact applies to any pair of disjoint subsets $\boldsymbol{X}, \boldsymbol{Y}$ that do not necessarily partition all the variables. In this more general setting (which will be useful in Section 5) distance is measured by the formula

$$\delta^p(\boldsymbol{X}, \boldsymbol{Y}) = \min_{A,B} \|F(X, Y, Z) - A(X, Z) - B(Y, Z)\|_p. \tag{4}$$

To finish the proof of Theorem 4 we use the sampling bounds derived from the Hoeffding Inequality in Lemma 11 and the p-norm scaling inequality in Claim 12. We then conclude the following claim of the estimation quality. See Appendix B for the complete analysis.

Claim 5. *The value $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p$ can be estimated within ε from $K^p \log(1/\gamma)/\epsilon^{2p}$ queries to F in linear time with probability $1 - \gamma$. Note that K is the absolute constant.*

Theorem 4 now follows from Claim 5 and inequality (3).

5 Variable Partitioning via Pairwise Estimates

Inspired by previous approaches [WRD+18, LW18], we design a greedy algorithm of variable partitioning. We analyze the algorithm and give an upper bound of the approximation error.

As we have shown in Equation (4), Section 4, the dependence score $D_F(\boldsymbol{X}, \boldsymbol{Y})$ can be applied to any pair of disjoint sets $\boldsymbol{X}$ and $\boldsymbol{Y}$. It is natural to apply them to the set with size one, namely, single variables. In fact, in [WRD+18] and [LW18] they consider the pairwise dependency between the single variables. The following Algorithm 1 treat the n variables as a complete graph. It greedily finds the pairs with the lowest $\|D_F(\boldsymbol{x}, \boldsymbol{y})\|$ values and removes the corresponding edge in the graph, until the graph is partitioned. We show that the algorithm achieves $\mathcal{O}(kn^2)$-factor approximation in terms of the partition error.

Algorithm 1 Approximate partition via pairwise estimates

1: **Input:** number of partitions k
2: **Output:** partition $\mathcal{P}$
3: For every pair of distinct variables $\boldsymbol{x}$, $\boldsymbol{y}$, calculate $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$ for $e(\boldsymbol{x}, \boldsymbol{y}) = \|D_F(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})\|_p$;
4: Create a weighted graph with vertices $\boldsymbol{V}$ and weights $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$;
5: Order the edges in increasing weight;
6: **repeat**
7: Remove the edge with the smallest weight;
8: **until** The graph has exactly k connected components

Theorem 6. *Assuming $e(\boldsymbol{x}, \boldsymbol{y}) \le \hat{e}(\boldsymbol{x}, \boldsymbol{y}) \le e(\boldsymbol{x}, \boldsymbol{y}) + \varepsilon$ for all $\boldsymbol{x}$ and $\boldsymbol{y}$,*

$$\delta^p(\mathcal{P}) \le (8k - 10)n^2(4\delta_k^p(F) + \varepsilon). \tag{5}$$

The ε is a relatively small error due to the estimation in line 3 of Algorithm 1. Namely, if the estimation is implemented by empirically averaging over $K^p \log(1/\gamma)/\epsilon^{2p}$ random samples then approximation guarantee (5) holds with probability at least $1-\gamma$. Note that K is the absolute constant in Theorem 4.

In the case $k = 2$, the leading constant $8k-10 = 6$ can be improved to 1 by using Claim 16 instead of Claim 13 and the fact that at most $n^2/4$ pairs cross the partition. See Appendix C for the description of both the claims and a rigorous proof of Theorem 6.

6 Variable Bi-partitioning via Submodular Function Minimization

We have analyzed the pairwise-based algorithm in Section 5. Because of the greedy nature of the approach, it leaves a gap between the factor $\mathcal{O}(kn^2)$ and the factor 4 that is possible for $\|D_F\|$. In this section, we show that almost factor-4 approximation is possible for bipartition. We then explain how this factor-4 approximation algorithm can be used agnostically for policy gradient methods. Let $f(\boldsymbol{X}) = \|D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})\|_2^2 = \mathbb{E}[D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$. The main result of this section is the below theorem.

Theorem 7. *$f(\boldsymbol{X})$ is a submodular function.*

The theorem is proved in Appendix D.

By equation (3), $\delta_2^2(F) \le \min_{\boldsymbol{X}} \|D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})\|_2 \le 4\delta_2^2(F)$, where the minimum is taken over all nontrivial bipartitions. Since submodular functions can be minimized efficiently [GLS81, Que98, Sch00, IFF01], by Theorem 7 we efficiently compute a factor-4 approximation to $\delta_2^2(F)$ given the oracle access to f.

To be exact, there exists an algorithm that, given oracle access to F, runs in time polynomial in n, $1/\varepsilon$, and $\log(1/\gamma)$, and outputs a set $\boldsymbol{X}$ such that $\delta_2(F) \le f(\boldsymbol{X}) \le 4\delta_2(F) + \varepsilon$ with probability at least $1 - \gamma$.

As is guaranteed by Theorem 7, we can agnostically learn the partition such that the decomposition error defined in Definition 1 is minimized. Finally, we demonstrate our approximately partitioned policy optimization algorithm with submodular optimization, in Algorithm 2. The algorithm is based on proximal policy optimization [SWD+17] and generalized advantage estimator [SML+15, DWS12].

Algorithm 2 Approximately partitioned policy optimization with submodular minimization

1: **Input:** Total number of samples T, batch size B, partition frequency M_p, number of value iterations M_w, initial policy parameter θ, initial value and advantage parameters w and μ;
2: **Output:** Optimized policy π_θ;
3: **for** each iteration j in $[T/B]$ **do**
4: Collect a batch of trajectory data $\{s_t^{(i)}, a_t^{(i)}, r_t^{(i)}\}_{i=1}^B$;
5: **for** M_θ iterations **do**
6: Update θ by one SGD step using PPO with the gradient estimator Eq. (1);
7: **end for**
8: **for** M_w iterations **do**
9: Update w and μ by minimizing $\|V^w(s_t) - R_t\|_2^2$ and $\|\hat{A} - A^\mu(s_t, a_t)\|_2^2$ in one step;
10: **end for**
11: Estimate $\hat{A}(s_t, a_t)$ using $V^w(s_t)$ by GAE;
12: **if** $j \equiv 0 \pmod{M_p}$ **then**
13: Define estimation $f(\boldsymbol{X}) = \mathbb{E}[D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$;
14: Run submodular minimization on $f(\boldsymbol{X})$;
15: Assign $\boldsymbol{X}$ and $\bar{\boldsymbol{X}}$ to $a_{(1)}$ and $a_{(-1)}$ in (1), respectively;
16: **end if** } Variable Partition
17: **end for**

Remarks on Algorithm 2. The algorithm is based on PPO [SWD+17]. We highlight the difference between our algorithm and PPO. Line 3-11 describe the policy gradient method with PPO and GAE in [SWD+17]. Line 6 achieves variance reduction of PG via RB. Line 12-16 find the near-optimal variable partition by minimizing the dependence score D, by the Theorem 7.

Extending bi-partitioning to k-partitioning. We have shown in Appendix E that a one-step MDP with quadratic reward function is equivalent to a Min-cut problem. It is natural to extend our algorithm as to how Min-cut is extended to Min-k-cut, which is a heavily studied topic [GBH00, Man17, SV95]. An example is to partition the set with the smallest partition error, and repeat until there exist exactly k partitions.

7 Experiments

7.1 Variable Partition on Synthetic Tasks

We study the performance of our variable partition algorithms, namely, the algorithm via pairwise estimates (PE, Algorithm 1) and via submodular minimization (SM, line 13-15 of Algorithm 2). We have shown in Figure 1 the difference between local and global optimality on Min-cut. Now we show this difference quantitatively, in Table 2 and Table 3.

We first study the Min-cut problem, which is equivalent to partitioning without additional error from estimation and optimization. The experiment is conducted on 10000 randomly generated graphs. Submodular minimization always finds the optimal partition. For the greedy algorithm, the *correctness* is defined as the number of times that it finds exactly the optimal partition. The *optimality* is defined as the average of partition error over optimal partition error. The experiments explain the need to design a global optimal algorithm, such as our algorithm via SM.

Table 2: Performance of the greedy algorithm on variable partition

#Nodes n	$n = 5$	$n = 10$	$n = 20$	$n = 40$	$n = 100$
Submodular	-	-	-	-	-
Greedy (correctness)	7753	6271	4226	2380	1101
Greedy (optimality)	1.060	1.203	1.408	1.352	1.250

Table 3 compares the performance of the algorithms, given only the oracle access of the objective $a^T H a$. Both [LW18] and our PE algorithm are greedy-based algorithms. Additionally, [LW18] has

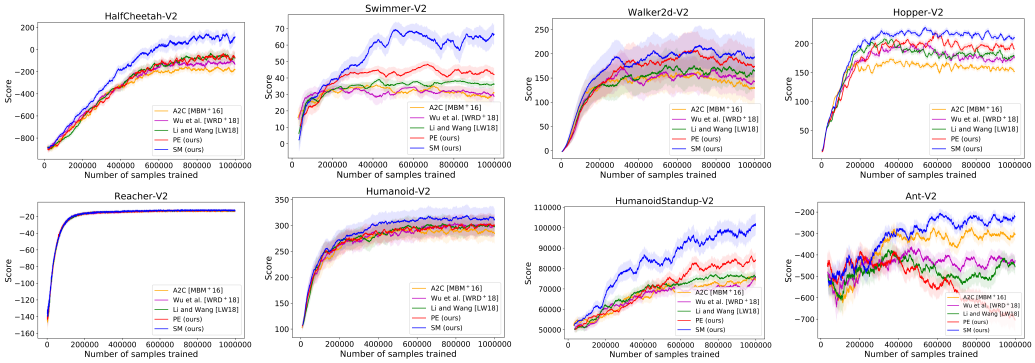

Figure 2: Empirical comparisons of performance on OpenAI Gym's MuJoCo high-dimensional control tasks. Each curve is averaged over 10 random seeds.

error from function approximation and calculating Hessian. Our PE algorithm incurs error from the estimation of D. Our SM algorithm also has error from the estimation of D, while it involves the error from the Queyranne's algorithm [Que98] used to minimize $\mathbb{E}[D^2]$.

Table 3: Comparisons of the algorithms on variable partition

#Nodes n	$n = 5$	$n = 10$	$n = 20$	$n = 40$	$n = 100$
Li and Wang [LW18] (correctness)	7553	5651	2929	1251	400
PE (correctness)	7709	6108	4001	2020	918
SM (correctness)	**9896**	**9630**	**9243**	**8193**	**6802**
Li and Wang [LW18] (optimality)	1.150	1.281	1.508	1.501	1.290
Wu et al. [WRD$^+$18] (optimality)	9.049	13.54	20.96	34.42	72.55
PE (optimality)	1.075	1.277	1.452	1.400	1.281
SM (optimality)	**1.020**	**1.028**	**1.101**	**1.110**	**1.025**

7.2 High-dimensional OpenAI Gym's MuJoCo Tasks

We evaluate on High-dimensional MuJoCo tasks the performance of decomposed PG via PE (Algorithm 2, replace line 13-15 with Algorithm 1) and SM (Algorithm 2). We compare PE and SM with the set of baselines, including A2C [MBM$^+$16], Wu et al. [WRD$^+$18], and Li and Wang [LW18]. All algorithms are based on the same implementation of PPO [SWD$^+$17] together with GAE [SML$^+$15], where only the PG estimator will be different. The results are shown in Figure 2.

We have conducted experiments on all eight environments from MuJoCo that has the action dimensional higher than one. We observe that our SM method outperforms existing methods on most of the tasks, which agrees with our theoretical finding that the algorithm agnostically finds the optimal decomposition of the policy gradient. Our PE algorithm is similar to previous PG decomposition methods empirically. There are environments, e.g. Reacher-v2 and Ant-v2, that we do not observe a significant difference. This is expected because there may not naturally exist an action space structure that we are aiming to learn.

8 Conclusion

In this paper, we focus on eliciting the Rao-Blackwell theorem into the policy gradient estimator in an algorithmic way. We avoid using stringent assumptions. Alternatively, we learn the PG decomposition agnostically from the variable partitioning of the action space. We formulate the problem rigorously, providing an estimator that measures the quality of the partitions. Armed with the estimator, we propose two algorithms for the variable partition problem. Our first algorithm is a rigorous extension of the greedy algorithm, while our second algorithm gives an almost factor-4 approximation in terms of the partitioning error. Our algorithms are evaluated empirically on both synthetic tasks and high-dimensional continuous control environments from OpenAI Gym's Mujoco. Both the problem formulation on the variable partition and the algorithms proposed in this paper are general enough to be fully adopted in other domains.

References

[CR96] George Casella and Christian P Robert. Rao-blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[DDG+17] Roee David, Irit Dinur, Elazar Goldenberg, Guy Kindler, and Igor Shinkar. Direct sum testing. *SIAM Journal on Computing*, 46(4):1336–1369, 2017.

[DWS12] Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

[GBH00] Nili Guttmann-Beck and Refael Hassin. Approximation algorithms for minimum k-cut. *Algorithmica*, 27(2):198–207, 2000.

[GCW+18] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

[GLG+16] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

[GLS81] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[IFF01] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, July 2001.

[Kos18] Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr`, 2018.

[KWY18] Sham Kakade, Mengdi Wang, and Lin F Yang. Variance reduction methods for sublinear reinforcement learning. *arXiv preprint arXiv:1802.09184*, 2018.

[LFDA16] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

[LFM+18] Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent control variates for policy optimization via stein identity. In *International Conference on Learning Representations*, 2018.

[LHP+15] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[LW18] Jiajin Li and Baoxiang Wang. Policy optimization with second-order advantage information. *arXiv preprint arXiv:1805.03586*, 2018.

[Man17] Pasin Manurangsi. Inapproximability of maximum edge biclique, maximum balanced biclique and minimum k-cut from the small set expansion hypothesis. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[MBM+16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016.

[Que98] Maurice Queyranne. Minimizing symmetric submodular functions. *Mathematical Programming*, 82(1-2):3–12, 1998.

[RGB14] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

[RLTK17] Aravind Rajeswaran, Kendall Lowrey, Emanuel V Todorov, and Sham M Kakade. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, pages 6550–6561, 2017.

[Sch00] Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Comb. Theory, Ser. B*, 80(2):346–355, 2000.

[SML+15] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[SMSM00] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[SV95] Huzur Saran and Vijay V Vazirani. Finding k cuts within twice the optimal. *SIAM Journal on Computing*, 24(1):101–108, 1995.

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[TBG+18] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. *arXiv preprint arXiv:1802.10031*, 2018.

[Wil92] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[WRD+18] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M.Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. In *International Conference on Learning Representations*, 2018.

A Variance Reduction by the Rao–Blackwell Theorem

We show the unbiasedness and the variance reduction properties of Eq. (1), namely,

$$\nabla_\theta J(\theta) = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)})].$$

It is worth note that in the previous works, the action-dependent baselines (Control Variates) are used when the decomposition $\{a_{(j)}\}$ is available [WRD$^+$18, LW18, LFM$^+$18, GCW$^+$18, TBG$^+$18]. Our work on RB is compatible with this line of research on CV, with the following gradient estimator. Alternatively,

$$\begin{aligned}\nabla_\theta J(\theta) = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)}) \\ - c(s, (a_{(j)}, \tilde{a}_{(-j)}))) - \nabla_\theta f_j(\theta, s, \xi)\nabla_{a_{(j)}} c_j(s, a_{(j)})],\end{aligned} \tag{6}$$

where the reparametrization term $\nabla_\theta f(\theta, s, \xi) \in \mathbb{R}^{N_\theta \times n}$ is divided into k parts as $\nabla_\theta f = [\nabla_\theta f_1, ..., \nabla_\theta f_k]$ and N_θ is the dimension of θ. $\tilde{a}_{(-j)}$ takes the same value as $a_{(-j)}$, but is treated as a constant when taking derivatives.

Proposition 3. *Suppose that the function $f : \mathbb{R}^n \times \to \mathbb{R} \cup \{+\infty\}$ satisfies zero partition error in Definition 1 with two sets of variables $X \in \mathbb{R}^{m_1}$ and $Y \in \mathbb{R}^{m_2}$ (i.e., $f(X, Y) = f_X + f_Y$, $m_1 + m_2 = n$). $P(X, Y)$ is a distribution enjoying the independence structure $P(X, Y) = P(X)P(Y)$ and $\mathbb{E}_{P(X,Y)}[\|f(X, Y)\|^2] < +\infty$. Then, we have*

- *(Unbiased Estimator) In the estimation of $\mathbb{E}_{P(X,Y)}[\nabla \log P(X, Y)f(X, Y)]$,*

$$\mathbb{E}_{P(X,Y)}[\nabla_\theta \log P(X, Y)f(X, Y)] = \mathbb{E}_{P(X)}[\nabla_\theta \log P(X)f_X] + \mathbb{E}_{P(Y)}[\nabla_\theta \log P(Y)f_Y].$$

- *(Variance Reduction) For the two estimators $g(X, Y) = \nabla \log P(X, Y)f(X, Y)$ and $h(X, Y) = \nabla_\theta \log P(X)f_X + \nabla \log P(Y)f_Y$,*

$$\mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} h(X_i, Y_i)] \le \mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} g(X_i, Y_i)],$$

  where n_s is the number of Monte-Carlo samples.

Proof. To verify the unbiasedness, observe that

$$\begin{aligned}\mathbb{E}_{P(X,Y)}[\nabla \log P(X, Y)f(X, Y)] &= \mathbb{E}_{P(X)P(Y)}[(\nabla \log P(X) + \nabla \log P(Y))(f_X + f_Y)] \\ &= \mathbb{E}_{P(X)}[\nabla \log P(X)f_X] + \mathbb{E}_{P(Y)}[\nabla \log P(Y)f_Y],\end{aligned}$$

which holds by the property of score function that $\mathbb{E}[\nabla \log P(X)] = 0$. Define the conditional estimator $\tau_X(X')$, we have

$$\begin{aligned}\tau_X(X') &= \mathbb{E}_{P(Y)}[\nabla_\theta \log P_\theta(X, Y)f(X, Y)|X = X'] \\ &= \mathbb{E}_{P(Y)}[(\nabla \log P(X) + \nabla \log P((Y)))(f_X + f_Y)|X = X'] \\ &= \nabla \log P(X')f_{X'} + \mathbb{E}_{P(Y)}[\nabla \log P(Y)f_Y] + \nabla_\theta \log P(X')\mathbb{E}_{P(Y)}[f_Y].\end{aligned} \tag{7}$$

Similarly, also define $\tau_Y(Y') = \mathbb{E}_{P(X)}[\nabla \log P(X, Y)f(X, Y)|Y = Y']$. Without loss of generality, we assume that $\mathrm{Var}[\tau_X(X')] < \mathrm{Var}[\tau_Y(Y')]$. Since the original estimator $g(X, Y)$ is sampled from the joint distribution $P(X, Y)$, the decomposed estimator $h(X, Y)$ does not introduce additional variance on top of $\mathrm{Var}[\tau_Y(Y')]$. Thus,

$$\mathrm{Var}[h(X, Y)] = \max(\mathrm{Var}[\tau_X(X')], \mathrm{Var}[\tau_Y(Y')]).$$

Combining with the Rao-Blackwell theorem, we have $\max(\mathrm{Var}[\tau_X(X')], \mathrm{Var}[\tau_Y(Y')]) \le \mathrm{Var}[g(X, Y)]$. The lemma follows. □

We provide a formal restatement of the last sentence at the end of Section 2.2.

Proposition 8. *When the advantage function can be decomposed into multiple components, the variance of the estimator* (1) *(or* (6)*) is strictly less than the Monte-Carlo policy gradient estimator proposed in [MBM+16] (or [LFM+18] and [GCW+18]).*

Proof. Recall that

$$\nabla_\theta J(\theta) = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)}) \\ - c(s, (a_{(j)}, \tilde{a}_{(-j)}))) - \nabla_\theta f_j(\theta, s, \xi)\nabla_{a_{(j)}} c_j(s, a_{(j)})].$$

We ignore the reparameterization term $\nabla_\theta f_j(\theta, s, \xi)\nabla_{a_{(j)}} c_j(s, a_{(j)})$ term as the variance of a reparameterization term is closed to zero. Thus we focus on the following estimator

$$\sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)}) - c(s, (a_{(j)}, \tilde{a}_{(-j)})))].$$

By Proposition 3 (Variance Reduction), we plug in the $f(a)$ as $A^\pi(s, a) - c(s, a)$. It follows immediately the desired result. □

B Lemmas and Claims for Theorem 4

Claim 9 (Completeness of D_F)**.** *For all disjoint $\boldsymbol{X}$, $\boldsymbol{Y}$, $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p \leq 4 \cdot \delta^p(\boldsymbol{X}, \boldsymbol{Y})$.*

Proof. By definition of $\delta(\boldsymbol{X}, \boldsymbol{Y})$ there exists a decomposition of the form

$$F(V) = A(X, Z) + B(Y, Z) + D(X, Y, Z),$$

where $\|D(X, Y, Z)\| = \delta(\boldsymbol{X}, \boldsymbol{Y})$. In the expansion of D_F all the A and B terms cancel out, leaving

$$\begin{aligned}\|D_F(\boldsymbol{X}, \boldsymbol{Y})\| &= \|D(X, Y, Z) + D(X', Y', Z) - D(X, Y', Z) - D(X', Y, Z)\| \\ &\leq \|D(X, Y, Z)\| + \|D(X', Y', Z)\| + \|D(X, Y', Z)\| + \|D(X', Y, Z)\| \\ &= 4\delta(\boldsymbol{X}, \boldsymbol{Y}).\end{aligned}$$

□

Claim 10 (Soundness of D_F)**.** *For all disjoint $\boldsymbol{X}$, $\boldsymbol{Y}$, $\delta^p(\boldsymbol{X}, \boldsymbol{Y}) \leq \|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_p$.*[1]

Proof. Let $\varepsilon = \|D_F(\boldsymbol{X}, \boldsymbol{Y})\|$. Then

$$F(X, Y, Z) \approx_\varepsilon F(X, Y', Z) - F(X', Y, Z) - F(X', Y', Z).$$

By fixing, there exist functions $x' = \underline{X}'(Z)$ and $y' = \underline{Y}'(Z)$ for which

$$F(X, Y, Z) \approx_\varepsilon F(\underline{X}'(Z), \underline{Y}'(Z), Z) - F(X, \underline{Y}'(Z), Z) - F(\underline{X}'(Z), Y, Z).$$

Therefore $F(X, Y, Z) \approx_\varepsilon A(X, Z) + B(Y, Z)$, where $A(X, Z) = F(X, \underline{Y}'(Z), Z)$ and $B(Y, Z) = F(\underline{X}'(Z), y, Z) - F(\underline{X}'(Z), \underline{Y}'(Z), Z)$ as desired. □

Lemma 11 (Hoeffding Inequality)**.** *Consider the bounded function $f : \mathbb{R}^n \to \mathbb{R}$ satisfying $f(X) \in [a, b]$ with $c = b - a$, the empirical average of $|f(X)|^p$ over N random samples X is within ϵ^p of $\|f\|_p^p$ except with probability $2\exp(-2N\epsilon^{2p}/c^2)$.*

Claim 12. *Assume $t, \hat{t} \geq 0$. If $t^p \leq \hat{t}^p \leq t^p + (\varepsilon/2)^p$ then $t \leq \hat{t} \leq t + \varepsilon$.*

Proof. The left-hand inequalities are immediate. For the right-hand ones we start we consider two cases. If $t \leq \varepsilon/2$, then $\hat{t}^p \leq 2(\varepsilon/2)^p \leq \varepsilon \leq t + \varepsilon$. If $t > \varepsilon/2$ then

$$\hat{t} - t \leq \frac{\hat{t}^p - t^p}{t^{p-1}} \leq \frac{(\varepsilon/2)^p}{(\varepsilon/2)^{p-1}} \leq \varepsilon.$$

□

[1] For Boolean functions under uniform measure this was proved by David et al. [DDG+17].

Claim 5. *The value $\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_p$ can be estimated within ε from $K^p\log(1/\gamma)/\epsilon^{2p}$ queries to F in linear time with probability $1-\gamma$. Note that K is the absolute constant.*

Proof. By the triangle inequality, $\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_p \le 4\|F\|_p$. By convexity $\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_p^p \le 4^{p-1}\|F\|_p^p$. By Lemma 11, an $(1-\gamma)$-probability estimate that lies between $\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_p^p$ and $\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_p^p + (\varepsilon/2)^p$ can be computed by taking the empirical average of $K^p\log(1/\gamma)/\epsilon^{2p}$ samples of D_F. Each sample of D_F can be obtained using four queries to F. The quality of the estimate then follows from Claim 12. □

C Proof of Theorem 6

Theorem 6. *Assuming $e(\boldsymbol{x},\boldsymbol{y}) \le \hat{e}(\boldsymbol{x},\boldsymbol{y}) \le e(\boldsymbol{x},\boldsymbol{y}) + \varepsilon$ for all $\boldsymbol{x}$ and $\boldsymbol{y}$,*

$$\delta^p(\mathcal{P}) \le (8k-10)n^2(4\delta_k^p(F)+\varepsilon).$$

For a partition $\mathcal{P}$ of the variables, let $\Delta(\mathcal{P}) = \sum \delta(\{\boldsymbol{x}\},\{\boldsymbol{y}\})$, where the sum is taken over all pairs that cross the partition. We will deduce Theorem 6 from the following bound on $\delta(\mathcal{P})$

Claim 13. *For every k-partition $\mathcal{P}$, $\delta(\mathcal{P}) \le (16k-20)\Delta(\mathcal{P})$.*

The following fact is immediate from the definitions of δ.

Fact 14. *For any partition $(\boldsymbol{U},\overline{\boldsymbol{U}})$ such that $\boldsymbol{X}\subseteq\boldsymbol{U}$ and $\boldsymbol{Y}\subseteq\overline{\boldsymbol{U}}$, $\delta(\boldsymbol{X},\boldsymbol{Y}) \le \delta(\boldsymbol{U},\overline{\boldsymbol{U}})$.*

Now we prove the theorem, assuming the correctness of Claim 13.

Proof of Theorem 6. By Claim 9 and Fact 14, all edges $(\boldsymbol{x},\boldsymbol{y})$ in the optimal partition must satisfy $e(\boldsymbol{x},\boldsymbol{y}) \le 4\delta_2(F)$. By our assumption on the quality of the approximations,

$$\hat{e}(\boldsymbol{x},\boldsymbol{y}) \le 4\delta_2(F)+\varepsilon. \tag{8}$$

Since the algorithm removes edges in increasing order of weight, all the edges that cross the output partition $\mathcal{P}$ must also satisfy this inequality. Then

$$\begin{aligned}
\delta(\mathcal{P}) &\le (16k-20)\Delta(\mathcal{P}) && \text{by Claim 13,}\\
&\le (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y}\text{ cross }\mathcal{P}} e(\boldsymbol{x},\boldsymbol{y}) && \text{by Claim 10,}\\
&\le (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y}\text{ cross }\mathcal{P}} \hat{e}(\boldsymbol{x},\boldsymbol{y})\\
&\le (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y}\text{ cross }\mathcal{P}} 4\delta_2(F)+\varepsilon && \text{by (8),}\\
&\le (8k-10)n^2\cdot(4\delta_2(F)+\varepsilon).
\end{aligned}$$

The last inequality holds because there are at most $\binom{n}{2} \le n^2/2$ pairs of variables crossing the partition. □

It remains to prove Claim 13. The proof is quite complex so we break it into a few more claims. We use $\boldsymbol{X}\boldsymbol{X}'$ to denote the union of the variable sets $\boldsymbol{X}$ and $\boldsymbol{X}'$.

Claim 15. *For disjoint sets of variables $\boldsymbol{X},\boldsymbol{X}',\boldsymbol{Y}$, $\delta(\boldsymbol{X}\boldsymbol{X}',\boldsymbol{Y}) \le \delta(\boldsymbol{X},\boldsymbol{Y}) + 2\delta(\boldsymbol{X}',\boldsymbol{Y})$.*

Proof. For simplicity of notation we omit the dependence on Z. Assume that

$$F(X,X',Y) \approx_\delta A(X,X') + B(X',Y) \qquad \text{and}$$
$$F(X,X',Y) \approx_{\delta'} A'(X,X') + B'(X,Y).$$

By the triangle inequality,

$$A(X,X') + B(X',Y) \approx_{\delta+\delta'} A'(X,X') + B'(X,Y).$$

Fix $X'(Z) = \underline{X}'(Z)$. Writing $C(X) = A(X,\underline{X}') - A'(X,\underline{X}')$ and $D(Y') = B(\underline{X}',Y')$ we get that

$$B'(X,Y) \approx_{\delta+\delta'} C(X) + D(Y).$$

By the triangle inequality (with the second equation), we get that

$$F(X,X',Y) \approx_{\delta+2\delta'} A'(X,X') + C(X) + D(Y).$$ □

Claim 16. *For every bipartition $\boldsymbol{X}, \overline{\boldsymbol{X}}$ of the variables, $\delta(\boldsymbol{X}, \overline{\boldsymbol{X}}) \leq 4 \cdot \Delta(\boldsymbol{X}, \overline{\boldsymbol{X}})$.*

Proof. By Claim 15,

$$\delta(\boldsymbol{X}'\{\boldsymbol{x}\}, \{\boldsymbol{y}\}) \leq \delta(\boldsymbol{X}', \{\boldsymbol{y}\}) + 2\delta(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})$$

for all $\boldsymbol{X}' \subseteq \boldsymbol{X} \setminus \{x\}$ and $\boldsymbol{y}$. Applying this inequality iteratively we conclude that $\delta(\boldsymbol{X}, \{\boldsymbol{y}\}) \leq 2\sum_{\boldsymbol{x} \in \boldsymbol{X}} \delta(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})$. Also by Claim 15

$$\delta(\boldsymbol{X}, \boldsymbol{Y}'\{\boldsymbol{y}\}) \leq \delta(\boldsymbol{X}, \boldsymbol{Y}') + 2\delta(\boldsymbol{X}, \boldsymbol{Y}'\{\boldsymbol{y}\}),$$

so $\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq 2\sum_{\boldsymbol{y} \in \boldsymbol{Y}} \delta(\boldsymbol{X}, \{\boldsymbol{y}\})$. Combining the two inequalities we obtain the desired conclusion. □

To extend the proof to larger k, we generalize the first inequality in this sequence to k-partitions.

The same sequence of inequalities then yields the conclusion of Theorem 6. It remains to prove Claim 13.

Claim 17. *For every $2k$-partition $(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k, \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k)$,*

$$\delta(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k, \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k) \leq 2\delta(\boldsymbol{Y}_1\boldsymbol{Z}_1, \ldots, \boldsymbol{Y}_k\boldsymbol{Z}_k) + 3\delta(\boldsymbol{Y}_1 \ldots \boldsymbol{Y}_k, \boldsymbol{Z}_1 \ldots \boldsymbol{Z}_k).$$

Proof. Assume that

$$F(\mathcal{V}) \approx_{\delta} F_1(Y_1, Z_1) + \cdots + F_t(Y_t, Z_t)$$
$$F(\mathcal{V}) \approx_{\delta'} A(Y_1, \ldots, Y_t) + B(Z_1, \ldots, Z_t).$$

By the triangle inequality

$$A(Y_1, \ldots, Y_t) + B(Z_1, \ldots, Z_t) \approx_{\delta+\delta'} F_1(Y_1, Z_1) + \cdots + F_t(Y_t, Z_t).$$

Fixing $Z_1, \ldots, Z_t$ to values $\underline{Z}_1, \ldots, \underline{Z}_t$ we get the decomposition

$$A(Y_1, \ldots, Y_t) \approx_{\delta+\delta'} F_1(Y_1, \underline{Z}_1) + \cdots + F_t(Y_t, \underline{Z}_t) - B(\underline{Z}_1, \ldots, \underline{Z}_t).$$

and similarly

$$B(Z_1, \ldots, Z_t) \approx_{\delta+\delta'} F_1(\underline{Y}_1, Z_1) + \cdots + F_t(\underline{Y}_t, Z_t) - A(\underline{Y}_1, \ldots, \underline{Y}_t).$$

Plugging these into the second equation gives the desired decomposition. □

Proof of Claim 13. We assume that k is a power of two and prove by induction that $\delta(\mathcal{P}) \leq c_k\Delta(\mathcal{P})$, where c_k is the sequence $c_{2k} = 2c_k + 12$, $c_2 = 4$. The base case $k = 2$ follows from Claim 16. Assume the claim holds for k and apply Claim 17 to $\mathcal{P}$. By inductive assumption and Claim 16,

$$\delta(\mathcal{P}) \leq 2 \cdot c_k\Delta(\boldsymbol{Y}_1\boldsymbol{Z}_1, \ldots, \boldsymbol{Y}_k\boldsymbol{Z}_k) + 3 \cdot 4\Delta(\boldsymbol{Y}_1 \ldots \boldsymbol{Y}_k, \boldsymbol{Z}_1 \ldots \boldsymbol{Z}_k).$$

Since $\mathcal{P}$ is a refinement of both these partitions, it follows that $\delta(\mathcal{P}) \leq (2c_k + 12)\Delta(\mathcal{P}) = c_{2k}\Delta(\mathcal{P})$, concluding the induction.

The recurrence solves to $c_k = 8k - 12$, proving the claim when k is a power of two. When it is not, the same reasoning applies to the closest power of two exceeding k (by taking some of the sets in the partition to be empty), which is at most $2k - 1$, proving the desired bound. □

D Proof of Theorem 7

Theorem 7. *$f(\boldsymbol{X})$ is a symmetric submodular function.*

We arm ourselves with a series of claims to prove Theorem 7.

Claim 18. *For every partition $(\boldsymbol{X}, \boldsymbol{Y})$, $\mathbb{E}[D_F(\boldsymbol{X}, \boldsymbol{Y})^2] = 4 \cdot \mathbb{E}[F(X, Y) \cdot D_F(X, Y, X', Y')]$.*

Proof. Let X_0, X_1 be independent copies of $\boldsymbol{X}$ and Y_0, Y_1 be independent copies of $\boldsymbol{Y}$. We can express D_F in the form

$$D_F = \sum_{i,j\in\{0,1\}} (-1)^{i+j} F(X_i, Y_j).$$

Then

$$\begin{aligned}
\mathbb{E}[D_F^2] &= \mathbb{E}\big[\sum_{i,j\in\{0,1\}} (-1)^{i+j} F(X_i, Y_j) \cdot \sum_{i',j'\in\{0,1\}} (-1)^{i'+j'} F(X_{i'}, Y_{j'})\big] \\
&= \mathbb{E}\big[\sum_{i,j,i',j'\in\{0,1\}} (-1)^{i+j+i'+j'} F(X_i, Y_j) \cdot F(X_{i'}, Y_{j'})\big] \\
&= \sum_{i,i',j,j'\in\{0,1\}} (-1)^{(i\oplus i')+(j\oplus j')} \cdot \mathbb{E}\big[F(X_i, Y_j) \cdot F(X_{i'}, Y_{j'})\big],
\end{aligned}$$

where $\oplus$ is addition modulo 2. By symmetry of the variables

$$\mathbb{E}\big[F(X_i, Y_j) \cdot F(X_{i'}, Y_{j'})\big] = \mathbb{E}\big[F(X_0, Y_0) \cdot F(X_{i\oplus i'}, Y_{j\oplus j'})\big]$$

substituting in the previous expression and changing the order of the expectation and the summation, we obtain

$$\mathbb{E}[D_F^2] = \mathbb{E}\big[F(X_0, Y_0) \cdot \sum_{i,i',j,j'\in\{0,1\}} (-1)^{(i\oplus i')+(j\oplus j')} F(X_{i\oplus i'}, Y_{j\oplus j'})\big].$$

The summation is exactly $4D_F(X_0, Y_0, X_1, Y_1)$ as desired. □

Let $D_F(\boldsymbol{X}, \boldsymbol{Y} \mid Z)$ denote $D_{F_Z}(\boldsymbol{X}, \boldsymbol{Y})$, where F_Z is obtained from F by fixing $\boldsymbol{Z}$ to Z.

Claim 19. *$D_F(\boldsymbol{XY}, \boldsymbol{ZW}) + D_F(\boldsymbol{XZ}, \boldsymbol{YW}) - D_F(\boldsymbol{XYZ}, \boldsymbol{W}) - D_F(\boldsymbol{X}, \boldsymbol{YZW})$ is equal to $D_F(\boldsymbol{Y}, \boldsymbol{Z} \mid XW') + D_F(\boldsymbol{Y}, \boldsymbol{Z} \mid X'W)$.*

Proof. Expanding the left-hand side yields sixteen terms of the form $F(X^*Y^*Z^*W^*)$ with a leading plus or minus sign, where the asterisk mark "$*$" indicates a primed or unprimed superscript. Each of the terms $F(XYZW)$ and $F(X'Y'Z'W')$ occurs exactly twice with a plus sign and twice with a minus sign, so those eight terms cancel out. There remain four terms of the type $F(XY^*Z^*W')$ and four of the type $F(X'Y^*Z^*W)$. Among those, the terms where exactly one of Y, Z is primed are negative and the others are positive. Therefore the type $F(XY^*Z^*W')$ terms add up to $D_F(\boldsymbol{Y}, \boldsymbol{Z} \mid XW')$ and the type $F(X'Y^*Z^*W)$ terms add up to $D_F(\boldsymbol{Y}, \boldsymbol{Z} \mid X'W)$. □

Claim 20. *$\mathbb{E}[F(XYZW)D_F(YZY'Z' \mid X'W)]$ is non-negative.*

Proof. We have by definition

$$\begin{aligned}
\mathbb{E}[F(XYZW)D_F(YZY'Z' \mid X'W)] = \mathbb{E}[F(XYZW)((F(X'YZW) - F(X'YZ'W)) \\
- (F(X'Y'Z'W) - F(X'Y'ZW)))].
\end{aligned}$$

For simplicity of the notation we omit the $\mathbb{E}_W$ at the beginning of each term and the trailing W in $F(\cdot)$, as W is not involved in the computation. Consider that

$$\begin{aligned}
\mathbb{E}_Y \operatorname{Var}_Z E_X[F(XYZ)] &= \mathbb{E}_{YZ}[(\mathbb{E}_X[F(XYZ)] - \mathbb{E}_{XZ}[F(XYZ)])^2] \\
&= \mathbb{E}_{YZ}[\mathbb{E}_X[F(XYZ)]^2] + \mathbb{E}_{YZ}[\mathbb{E}_{XZ}[F(XYZ)]^2] \\
&\quad - 2\mathbb{E}_{YZ}[\mathbb{E}_{XZ}[F(XYZ)]\mathbb{E}_X[F(XYZ)]] \\
&= \mathbb{E}_{YZ}[\mathbb{E}_X[F(XYZ)]^2] - \mathbb{E}_Y[\mathbb{E}_{XZ}[F(XYZ)]^2] \\
&= \mathbb{E}_{YZ}[\mathbb{E}_X[F(XYZ)]\mathbb{E}_{X'}[F(X'YZ)]] \\
&\quad - \mathbb{E}_Y[\mathbb{E}_{XZ}[F(XYZ)]\mathbb{E}_{X'Z'}[F(X'YZ')]],
\end{aligned}$$

where the last equation follows the independency of X and X' and the independency of Z and Z'. Similarly, we have

$$\begin{aligned}\operatorname{Var}_Z \mathbb{E}_Y E_X[F(XYZ)] &= \mathbb{E}_Z[(\mathbb{E}_{XY}[F(XYZ)] - \mathbb{E}_{XYZ}[F(XYZ)])^2] \\ &= \mathbb{E}_Z[\mathbb{E}_{XY}[F(XYZ)]^2] + \mathbb{E}_Z[\mathbb{E}_{XYZ}[F(XYZ)]^2] \\ &\quad - 2\mathbb{E}_Z[\mathbb{E}_{XY}[F(XYZ)]\mathbb{E}_{XYZ}[F(XYZ)]] \\ &= \mathbb{E}_Z[\mathbb{E}_{XY}[F(XYZ)]^2] - \mathbb{E}_{XYZ}[F(XYZ)]^2 \\ &= \mathbb{E}_Z[\mathbb{E}_{XY}[F(XYZ)]\mathbb{E}_{X'Y'}[F(X'Y'Z)]] \\ &\quad - \mathbb{E}_{XYZ}[F(XYZ)]\mathbb{E}_{X'Y'Z'}[F(X'Y'Z')].\end{aligned}$$

We plug both equations in and get

$$\mathbb{E}[F(XYZW)D_F(YZY'Z' \mid X'W)] = \mathbb{E}_Y \operatorname{Var}_Z \mathbb{E}_X[F(XYZ)] - \operatorname{Var}_Z \mathbb{E}_Y \mathbb{E}_X[F(XYZ)].$$

We show that the right-hand side of the equation is non-negative. In fact, let $F' = \mathbb{E}_X[F(XYZ)] - \mathbb{E}_{XZ}[F(XYZ)]$, then

$$\mathbb{E}_Y \operatorname{Var}_Z \mathbb{E}_X[F(XYZ)] = \mathbb{E}_Y \mathbb{E}_Z[F'^2] = \mathbb{E}_Z[\mathbb{E}_Y[F'^2]].$$

By the Cauchy-Schwarz inequality, we conclude

$$\begin{aligned}\operatorname{Var}_Z \mathbb{E}_Y \mathbb{E}_X[F(XYZ)] &= \mathbb{E}_Z[(\mathbb{E}_Y \mathbb{E}_X[F(XYZ)] - \mathbb{E}_Z \mathbb{E}_Y \mathbb{E}_X[F(XYZ)])^2] \\ &= \mathbb{E}_Z[(\mathbb{E}_Y[\mathbb{E}_X[F(XYZ)] - \mathbb{E}_{XZ}[F(XYZ)]])^2] \\ &\le \mathbb{E}_Z[\mathbb{E}_Y[(\mathbb{E}_X[F(XYZ)] - \mathbb{E}_{XZ}[F(XYZ)])^2]] \\ &= \mathbb{E}_Z[\mathbb{E}_Y[F'^2]]\end{aligned}$$

as desired. □

Proof of Theorem 7. Proving submodularity of f amounts to verifying the inequality

$$\mathbb{E}[D_F(\boldsymbol{XY}, \boldsymbol{ZW})^2] + \mathbb{E}[D_F(\boldsymbol{XZ}, \boldsymbol{YW})^2] - \mathbb{E}[D_F(\boldsymbol{XYZ}, \boldsymbol{W})^2] - \mathbb{E}[D_F(\boldsymbol{X}, \boldsymbol{YZW})^2] \ge 0$$

for every partition $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W})$ of the variables. This follows by applying Claims 18, 19, and 20 to each of the two expressions in order. □

E Experiment Details

E.1 Details of the Two Examples and the Synthetic Task

The two examples and the 10000 random graphs in the experiments are generated as an analogous to practical RL environments, where the dependency between variables are sparse. Each edge has a 0.3 probability to be a dependency where the value is uniformly sampled from $[2, 10]$. Otherwise the edge is deemed noise and is uniformly sampled from $[0, 1]$. We make both the matrix tractable in the optimization by substituting an identity matrix. The matrix used in the experiments are $H^1 - 11.65\mathbb{I}$ and $H^2 - 14.98\mathbb{I}$.

$$H^1 = \begin{bmatrix} 0 & 7.14 & 6.53 & 5.67 & 0.58 \\ 7.14 & 0 & 0.71 & 5.29 & 0.52 \\ 6.53 & 0.71 & 0 & 0.37 & 8.08 \\ 5.67 & 5.29 & 0.37 & 0 & 0.96 \\ 0.58 & 0.53 & 8.08 & 0.96 & 0 \end{bmatrix}, H^2 = \begin{bmatrix} 0 & 0.27 & 0.67 & 0.48 & 7.52 \\ 0.27 & 0 & 4.11 & 4.12 & 5.22 \\ 0.67 & 4.11 & 0 & 4.74 & 0.80 \\ 0.48 & 4.12 & 4.74 & 0 & 0.46 \\ 7.52 & 5.22 & 0.80 & 0.46 & 0 \end{bmatrix}.$$

E.2 Understanding the Synthetic Environment

The Minimum Cut Problem (Min-cut) on a weighted undirected graph is a well-known problem whose objective is a submodular function. We show that the variable partition on our synthetic environment is equivalent to Min-cut. This also provides an intuitive understanding of our Theorem 7 and Algorithm 2.

Let $a_{\boldsymbol{X}}$ denotes elements of the vector a whose index is in $\boldsymbol{X}$ and $H_{\boldsymbol{X},\boldsymbol{X}}$ denotes elements of the matrix H whose row and column indexes are in $\boldsymbol{X}$. Then the variable partition problem of the synthetic environment is

$$\underset{\boldsymbol{X}}{\text{minimize}} \quad F(\boldsymbol{X}) = \mathbb{E}_a\left[a_{\boldsymbol{X}}^T H_{\boldsymbol{X},\boldsymbol{X}} a_{\boldsymbol{X}} + a_{\overline{\boldsymbol{X}}}^T H_{\overline{\boldsymbol{X}},\overline{\boldsymbol{X}}} a_{\overline{\boldsymbol{X}}}\right]. \tag{9}$$

Proposition 21. *Under Gaussian policies, minimizing the dependence score D over the bipartition $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ on the synthetic environment is equivalent to a Min-cut problem.*

Proof. Without loss of generality let $H = \mathbf{1}$ be a all-one matrix. This will correspond to an unweighted graph. For a general H we just have to change the weight of the edge (i, j) to H_{ij}. Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_j)$, $\boldsymbol{Y} = \overline{\boldsymbol{X}} = (\boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_n, \boldsymbol{y}_{j+1}, .., \boldsymbol{y}_n)$. We have

$$\begin{aligned}
D_F(\boldsymbol{X}, \boldsymbol{Y}) &= F(X, Y) + F(X', Y') - F(X', Y) - F(X, Y') \\
&= (x_1 + \cdots + x_n)^2 + (y_1 + \cdots + y_n)^2 + (x'_1 + \cdots + x'_n)^2 + (y'_1 + \cdots + y'_n)^2 \\
&\quad - (x_1 + \cdots + x_i + x'_{i+1} + \cdots + x'_n)^2 - (y_1 + \cdots + y_j + y'_{j+1} + \cdots + y'_n)^2 \\
&\quad - (x'_1 + \cdots + x'_i + x_{i+1} + \cdots + x_n)^2 - (y'_1 + \cdots + y'_j + y_{j+1} + \cdots + y_n)^2.
\end{aligned}$$

Let $x_1 \ldots x_i \circ x_{i+1} \ldots x_n$ denote $\sum_{1 \le k \le i, i+1 \le l \le n} x_k x_l$, the above equals to

$$\begin{aligned}
D_F(\boldsymbol{X}, \boldsymbol{Y}) = x_1 \ldots x_i \circ x_{i+1} \ldots x_n &+ y_1 \ldots y_j \circ y_{j+1} ... y_n \\
&+ x'_1 \ldots x'_i \circ x'_{i+1} \ldots x'_n + y'_1 \ldots y'_j \circ y'_{j+1} \ldots y'_n \\
&- x_1 \ldots x_i \circ x'_{i+1} \ldots x'_n - y_1 \ldots y_j \circ y'_{j+1} \ldots y'_n \\
&- x'_1 \ldots x'_i \circ x_{i+1} \ldots x_n - y'_1 \ldots y'_j \circ y_{j+1} \ldots y_n.
\end{aligned}$$

Denoting $dx_k = x_k - x'_k$ and $dy_k = y_k - y'_k$, respectively, it is then equal to

$$D_F(\boldsymbol{X}, \boldsymbol{Y}) = dx_1 \ldots dx_i \circ dx_{i+1} \ldots dx_n + dy_1 \ldots dy_j \circ dy_{j+1} \ldots dy_n.$$

Since each x_k is a Gaussian random variable, we have $dx_k \sim \mathcal{N}(0, \sqrt{2}\sigma)$. Hence,

$$\begin{aligned}
\mathbb{E}[D_F^2] &= \mathbb{E}[dx_1^2 \ldots dx_i^2 \circ dx_{i+1}^2 \ldots dx_n^2 + dy_1^2 \ldots dy_j^2 \circ dy_{j+1}^2 \ldots dy_n^2] \\
&= \mathbb{E}[dx_1^2]\mathbb{E}[dx_2^2](i(n-i) + j(n-j)) \\
&= 4\sigma^4(i(n-i) + j(n-j)),
\end{aligned}$$

where $i(n-i) + j(n-j)$ is the number of edge cut by the partition $(\boldsymbol{X}, \boldsymbol{Y})$. Minimizing $D_F(\boldsymbol{X}, \boldsymbol{Y})$ is exactly equivalent to finding the solution of the Min-cut problem. □

E.3 Details on The MuJoCo Tasks

We use three neural networks: a policy network for PPO, a value network for GAE, and an advantage network solely used in the partition (line 13-15 of the algorithm). The networks have the same architecture as is in the original A2C and PPO work [MBM$^+$16, SWD$^+$17]. As the number of dimensions of the original environments are relatively low, we augment the environments' dimension such that we can observe the comparison under high-dimensional settings. In the augmentation, the agent controls two independent instances of the sub-agent of the same game. The state and action are the concatenation of the states and actions of the independent agents. The reward the agent receives is the averaged reward of its sub-agents. Correspondingly, we use $k = 2$ in [LW18] and our PE algorithm.