IEEE TRANSACTIONS ON
SIGNAL PROCESSING
A PUBLICATION OF THE IEEE SIGNAL PROCESSING SOCIETY
www.signalprocessingsociety.org
Indexed in PubMed and MEDLINE, products of the United States National Library of Medicine

Policy Gradient Importance Sampling for Bayesian Inference

Yousef El-Laham, Mónica F. Bugallo
Department of Electrical & Computer Engineering, Stony Brook University, Stony Brook, NY, USA
{yousef.ellaham, monica.bugallo}@stonybrook.edu

***Abstract*—In this paper, we propose a novel adaptive importance sampling (AIS) algorithm for probabilistic inference. The sampler learns a proposal distribution adaptation strategy by framing AIS as a reinforcement learning problem. Under this structure, the proposal distribution of the sampler is treated as an agent whose state is controlled using a parameterized policy. At each iteration of the algorithm, the agent earns a reward that is related to its contribution to the variance of the AIS estimator of the normalization constant of the target distribution. Policy gradient methods are employed to learn a locally optimal policy that maximizes the expected value of the sum of all rewards. Numerical simulations on two different examples demonstrate promising results for the future application of the proposed method to complex Bayesian models.**

I. Introduction

Importance sampling (IS) is a Monte Carlo technique that can be used to approximate intractable integrals [1], [2]. The methodology has a special application in Bayesian signal processing, where it can be used to compute expectations taken w.r.t. a posterior distribution of a set of unknown parameters in a probabilistic model [3]. In IS, samples are drawn from a *proposal* distribution and are weighted w.r.t. the *target* distribution of interest (e.g., a posterior distribution). The collection of samples and weights can be used to approximate integrals w.r.t. the target distribution via a weighted sum. While IS has nice theoretical guarantees, it suffers from the *curse-of-dimensionality* [4], and so its performance deteriorates for targets with a large number of unknowns (high-dimensional). This is analogous to the problem faced by most sampling methods, such as rejection sampling [5], Markov chain Monte Carlo (MCMC) [6], and sequential Monte Carlo (SMC) [7].

One of the key ingredients for scalable and efficient IS methods in high-dimensional scenarios is the choice of the proposal distribution. It is well known that the variance of an IS estimator scales with the mismatch between the target distribution and the proposal distribution. An optimal proposal distribution is one that minimizes the variance of the estimator of the integral of interest [8], which for almost all cases, is intractable and cannot be sampled from directly. To improve the efficiency of IS methods for high-dimensional targets, adaptive IS (AIS) methods have been proposed [9], [10]. A generic AIS algorithm adapts the proposal distribution over iterations to improve the performance of the sampler. Typically, the proposal is adapted in a manner that solves some optimality criterion. For example, in the mixture population Monte Carlo algorithm, a mixture proposal distribution is adapted to minimize the Kullback-Leibler divergence (KLD) between the target and the proposal [11]. In several other works, the proposal distribution is adapted to minimize the variance of the importance weights [12], [13].

Although AIS algorithms obtain better proposals with increasing iterations, the performance of the samplers can be burdened by samples drawn from poorly fit proposals used in the early iterations of the algorithm. Several solutions have been proposed to tackle this problem based on re-weighting previously drawn samples. In [14], samples are re-weighed at each iteration of the algorithm using a temporal deterministic mixture, which reduces the variance of the importance weights. In [15], a weighted AIS method is proposed which reweighs the samples drawn at each iteration according to the variance of the corresponding IS estimator. More recently, there have been several pushes to improve the performance of adaptive Monte Carlo methods using ideas from optimal control. In [16], techniques in reinforcement learning (RL) are used to learn the sampling policy (over discrete states) of an MCMC sampling scheme in a way maximizes the performance of the algorithm. In [17], a controlled SMC scheme is proposed which optimizes the sequence of proposals iteratively using an approximate dynamic programming procedure. There are also approximate inference methods that use dynamic programming to optimize the transition kernel of a Markov process, so that the terminal distribution of the process is the posterior [18]. These methods can also be used to improve SMC methods. However, unlike SMC samplers, which apply IS sequentially to approximate intermediate target distributions whose product is the posterior [19], AIS methods build an approximation by weighting directly according to the posterior. Importantly, in AIS, the samples and weights obtained in all iterations of the algorithm contribute to the approximation to the target posterior distribution. Therefore, to the best of our knowledge, the use of RL methods to optimize the performance of AIS remains relatively unexplored.

In this article, we propose a novel formulation for optimizing the performance of an AIS sampler that is based on stochastic optimal control, where the adaptation of proposals is controlled using a parameterized policy. This leads to an AIS scheme, where policy gradient (PG) methods from the RL literature are employed to find the optimal policy parameters. Unlike standard AIS algorithms that use a fixed

The authors thank the support of the National Science Foundation (NSF) under Award CCF-1617986. The Institute for Advanced Computational Science at Stony Brook University allowed for access to the high-performance SeaWulf computing system sponsored by NSF (OAC-1531492).

Digital Object Identifier 10.1109/TSP.2021.3093792

adaptation strategy to optimize the proposal distribution, the proposed method instead learns an adaptation strategy that optimizes the entire iterative procedure of the AIS algorithm and is robust to the initialization of the proposal parameters. Our main contributions are as follows: 1) We show that for the considered AIS sampler, finding the optimal policy is akin to solving a RL problem, where the rewards are related to the variance of the importance weights; 2) We show that the considered reward function is related to a statistical distance between the target and the proposal distributions; 3) We derive a surrogate reward function, such that when its return is maximized, the proposed algorithm minimizes an upper bound on the cumulative sum of a family of divergences between the target and the proposal distributions; and 4) We describe in detail the implementation of the method. Simulation results indicate that the novel sampler outperforms other AIS samplers for high-dimensional targets given sufficient training iterations.

Summary of Notation: Vectors are denoted by boldface lower-case letters (e.g., $\mathbf{x}$ and $\boldsymbol{\theta}$). Matrices are denoted by boldface upper-case letters (e.g., $\mathbf{X}$ and $\boldsymbol{\Sigma}$). The superscript $(\cdot)^\top$ is used to denote the transpose operation. Sets are denoted by calligraphic letters (e.g., $\mathcal{D}$ and Θ). The identity matrix of dimension d is denoted by $\mathbf{I}_d$, while a d-dimensional column vector of zeros and a d-dimensional column vector of ones are denoted by $\mathbf{0}_d$ and $\mathbf{1}_d$, respectively. The expected value and the variance of a function $f(\mathbf{x})$ taken with respect to a probability distribution $g(\mathbf{x})$ are denoted by $\mathbb{E}_g[f(\mathbf{x})]$ and $\mathbb{V}_g[f(\mathbf{x})]$, respectively. Finally, subscripts are used to denote data and iteration index (e.g., $\mathbf{y}_n$ and $\boldsymbol{\theta}_i$), while parenthesized superscripts denote sample index (e.g., $\mathbf{x}^{(m)}$).

II. Problem Formulation

We formulate the addressed problem under the Bayesian framework, and the objective is to infer the posterior distribution of an unknown parameter $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^{d_x}$ given an observed dataset $\mathbf{y}_{1:N} = [\mathbf{y}_1, \ldots, \mathbf{y}_N] \in \mathbb{R}^{d_y \times N}$, where we have $\mathbf{y}_n \in \mathbb{R}^{d_y \times 1}$ for $n = 1, \ldots, N$. By Bayes' theorem, the posterior of $\mathbf{x}$ given $\mathbf{y}_{1:N}$ can be expressed as

$$\varphi(\mathbf{x}) \triangleq p(\mathbf{x}|\mathbf{y}_{1:N}) = \frac{p(\mathbf{y}_{1:N}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y}_{1:N})}, \tag{1}$$

where $\varphi(\mathbf{x})$ is referred to as the *target distribution* (i.e., posterior distribution), $p(\mathbf{y}_{1:N}|\mathbf{x})$ is the *likelihood* of $\mathbf{x}$, $p(\mathbf{x})$ is the *prior distribution* of $\mathbf{x}$, and $Z \triangleq p(\mathbf{y}_{1:N})$ is called the *normalization constant* or *evidence*. In general, the evidence,

$$Z = \int_{\mathcal{X}} p(\mathbf{y}_{1:N}|\mathbf{x})p(\mathbf{x})d\mathbf{x}, \tag{2}$$

is an intractable integral and one can only evaluate the numerator of (1), which we denote by $\tilde{\varphi}(\mathbf{x})$ and refer to as the *unnormalized target distribution*, i.e.,

$$\tilde{\varphi}(\mathbf{x}) \triangleq p(\mathbf{y}_{1:N}|\mathbf{x})p(\mathbf{x}) = Z\varphi(\mathbf{x}). \tag{3}$$

Our goal is to compute expectations w.r.t. the target posterior $\varphi(\mathbf{x})$ under the restriction that we can only evaluate the unnormalized target $\tilde{\varphi}(\mathbf{x})$. In particular, we would like evaluate integrals of the form

$$\mathcal{I}(f) \triangleq \int_{\mathcal{X}} f(\mathbf{x})\varphi(\mathbf{x})d\mathbf{x} = \mathbb{E}_{\varphi}[f(\mathbf{x})], \tag{4}$$

where $f(\mathbf{x})$ is a function assumed to be integrable w.r.t. $\varphi(\mathbf{x})$. For general functions $f(\mathbf{x})$, the integral in (4) is intractable, and one must employ numerical integration methods that only require the evaluation of $\tilde{\varphi}(\mathbf{x})$ to estimate $\mathcal{I}(f)$.

III. Importance Sampling

We consider IS techniques to estimate expectations w.r.t. to the target $\varphi(\mathbf{x})$. In standard IS, we can approximate posterior expectations using samples drawn from an alternative distribution $q(\mathbf{x};\boldsymbol{\theta})$, called a *proposal distribution*, where $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{d_\theta}$ are the parameters of the proposal distribution. This is accomplished by rewriting the expectation in (4) in terms of the unnormalized target $\tilde{\varphi}(\mathbf{x})$ and the proposal $q(\mathbf{x};\boldsymbol{\theta})$, i.e.,

$$\mathcal{I}(f) = \frac{\int_{\mathcal{X}} f(\mathbf{x})\tilde{\varphi}(\mathbf{x})d\mathbf{x}}{\int_{\mathcal{X}} \tilde{\varphi}(\mathbf{x})d\mathbf{x}} = \frac{\mathbb{E}_{q_{\boldsymbol{\theta}}}\left[\frac{f(\mathbf{x})\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right]}{\mathbb{E}_{q_{\boldsymbol{\theta}}}\left[\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right]}, \tag{5}$$

where $\mathbb{E}_{q_{\boldsymbol{\theta}}}[\cdot]$ denotes an expectation taken w.r.t. $q(\mathbf{x};\boldsymbol{\theta})$. We define the unnormalized standard importance weight as $\tilde{w}(\mathbf{x},\boldsymbol{\theta}) \triangleq \tilde{\varphi}(\mathbf{x})/q(\mathbf{x};\boldsymbol{\theta})$. Then, the *self-normalized* IS estimator of $\mathcal{I}(f)$ based on a proposal $q(\mathbf{x};\boldsymbol{\theta})$ is given by

$$\hat{\mathcal{I}}_{\text{IS}}^M = \frac{\frac{1}{M}\sum_{m=1}^M \tilde{w}(\mathbf{x}^{(m)},\boldsymbol{\theta})f(\mathbf{x}^{(m)})}{\frac{1}{M}\sum_{m=1}^M \tilde{w}(\mathbf{x}^{(m)},\boldsymbol{\theta})}, \tag{6}$$

where $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(M)} \overset{i.i.d.}{\sim} q(\mathbf{x};\boldsymbol{\theta})$ are the M samples drawn from the proposal distribution. We remark that the denominator of (6) is an unbiased estimator of Z [20], i.e., if

$$\hat{Z}_{\text{IS}}^M = \frac{1}{M}\sum_{m=1}^M \tilde{w}(\mathbf{x}^{(m)},\boldsymbol{\theta}), \tag{7}$$

then $\mathbb{E}[\hat{Z}_{\text{IS}}^M] = Z$. Furthermore, it is well known that the estimator in (6) is consistent, i.e., for any $\epsilon > 0$ we have

$$\lim_{M\to\infty} P(\|\mathcal{I}(f) - \hat{\mathcal{I}}_{\text{IS}}^M\| > \epsilon) = 0, \tag{8}$$

where $P(A)$ denotes the probability of an event A and $\|\cdot\|$ denotes a norm (e.g., a vector norm or a matrix norm). Despite this theoretical guarantee, only a finite number of samples contribute to estimates in practice, and if $\boldsymbol{\theta}$ is poorly chosen, then IS estimators can suffer from high variance.

Let $\mathbb{V}_{q_{\boldsymbol{\theta}}}[\cdot]$ denote the variance taken w.r.t. $q(\mathbf{x};\boldsymbol{\theta})$. A common optimality criterion utilized to choose proposal distributions for IS methods is the *minimum-variance importance weights* criterion [21], [22]. The goal is to find the $\boldsymbol{\theta}$ that solves the following optimization problem:

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}\in\Theta} C(\boldsymbol{\theta}), \tag{9}$$

where the cost function $C(\boldsymbol{\theta}) \triangleq \mathbb{V}_{q_{\boldsymbol{\theta}}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta})]$ corresponds to the variance of the standard importance weights under proposal $q(\mathbf{x};\boldsymbol{\theta})$. It can be shown that this is equivalent to finding the parameters which minimize the variance of the estimator of the evidence, $\hat{Z}_{\text{IS}}^{M}$. Moreover, the optimal proposal parameters $\boldsymbol{\theta}^{\star}$ are those for which $q(\mathbf{x};\boldsymbol{\theta}^{\star})$ best approximates $\varphi(\mathbf{x})$.

IV. An Optimal Adaptive Importance Sampler

Over the past years, large efforts have been devoted to the development of adaptive MC methods which learn a suitable proposal distribution through an iterative sampling process [23]. In the case of IS, these efforts have led to the development of AIS methods. At each iteration of an AIS algorithm, M samples are drawn from a proposal $q(\mathbf{x};\boldsymbol{\theta}_i)$, where i denotes iteration index. The generated samples are weighted and then, the proposal parameters $\boldsymbol{\theta}_i$ are adapted. The adaptation of $\boldsymbol{\theta}_i$ depends on an auxiliary variable $\mathbf{u}_i \in \Upsilon$ that determines how one obtains $\boldsymbol{\theta}_{i+1}$ from $\boldsymbol{\theta}_i$. In most implementations, the adaptation rule at the ith iteration can be described using a deterministic function of the proposal parameters $\boldsymbol{\theta}_i$ and the auxiliary variables $\mathbf{u}_i$. More generally, however, we can represent it by using a conditional probability density function $h : \Theta \times \Theta \times \Upsilon \to [0,\infty)$, i.e.,

$$\boldsymbol{\theta}_{i+1} \sim h(\boldsymbol{\theta}_{i+1}|\boldsymbol{\theta}_i, \mathbf{u}_i). \tag{10}$$

The auxiliary variable $\mathbf{u}_i$ is usually a function of the history of generated samples. For example, if $\boldsymbol{\theta}_i$ corresponds to a location parameter, one possible adaptation rule is

$$\boldsymbol{\theta}_{i+1} = \left(\frac{1}{\sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i)} \sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i)\mathbf{x}_i^{(m)} \right) + \boldsymbol{\epsilon}_i,$$

which is a weighted sample mean perturbed with additive Gaussian noise $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}_{d_\theta}, \sigma_p^2 \mathbf{I}_{d_\theta})$. Here, the auxiliary variable $\mathbf{u}_i$ corresponds to the samples generated at the ith iteration of the algorithm, their corresponding weights, and the variance of the perturbation.

Using the collection of samples and importance weights, we can obtain the self-normalized AIS estimator of $\mathcal{I}(f)$ as

$$\hat{\mathcal{I}}_{\text{AIS}}^{M\times I} = \frac{\frac{1}{MI}\sum_{i=1}^{I}\sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i) f(\mathbf{x}_i^{(m)})}{\frac{1}{MI}\sum_{i=1}^{I}\sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i)}. \tag{11}$$

Similar to the estimator in (6), the denominator of (11) corresponds to an estimator of the evidence, i.e.,

$$\hat{Z}_{\text{AIS}}^{M\times I} = \frac{1}{MI}\sum_{i=1}^{I}\sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i). \tag{12}$$

One can show that $\hat{Z}_{\text{AIS}}^{M\times I}$ is an unbiased estimator of Z, i.e., $\mathbb{E}\big[\hat{Z}_{\text{AIS}}^{M\times I}\big] = Z$ via a straightforward application of the law of total expectation.

One strategy for determining the sequence of proposal parameters to be used in AIS is to employ a stochastic gradient descent (SGD) method to explicitly solve (9) [12], [13]. Essentially, one begins with an initial parameter $\boldsymbol{\theta}_0$ and applies an SGD update at the end of each AIS iteration:

$$\boldsymbol{\theta}_{i+1} = \Pi_\Theta \left(\boldsymbol{\theta}_i - \eta_i g(\boldsymbol{\theta}_i) \right), \tag{13}$$

where $\eta_i > 0$ is a positive learning rate, $g(\boldsymbol{\theta}_i)$ is a stochastic gradient that satisfies $\mathbb{E}\left[g(\boldsymbol{\theta}_i)|\boldsymbol{\theta}_i\right] = \nabla_{\boldsymbol{\theta}_i} C(\boldsymbol{\theta}_i)$, and $\Pi_\Theta(\cdot)$ denotes a projection onto the feasibility set Θ. The update in (13) can be embedded as the adaptation step in the AIS sampler, where stochastic gradients are obtained using the set of samples drawn from the most recent proposal. In this context, the auxiliary variables $\mathbf{u}_i$ are reflected in the learning rate η_i and the stochastic gradient $g(\boldsymbol{\theta}_i)$. Provided that the sequence of learning rates are chosen according to the Robbins-Monro stochastic approximation conditions, the proposal parameters converge in expectation to a local minimum of $C(\boldsymbol{\theta})$ [24], [25], [26].

Using SGD to solve (9) leads to a locally optimal IS proposal in the limit as the number of iterations tends to infinity; however, the proposal parameter values obtained in the early iterations of an SGD algorithm may contribute to a high variance of the AIS estimator of the normalization constant. For this reason, it is important to optimize the *sequence of proposal parameters* in an AIS algorithm since they all play a role in the performance of the estimators in (11) and (12). Therefore, a more suitable goal would be to optimize the auxiliary variables $\mathbf{u}_{1:I} \triangleq \{\mathbf{u}_1, \dots, \mathbf{u}_I\}$ responsible for generating the sequence of proposal parameters in an AIS method, so that the variance of $\hat{Z}_{\text{AIS}}^{M\times I}$ is minimized. In the case that the auxiliary variables are deterministic, this instantiates the following optimization problem over $\mathbf{u}_{1:I}$:

$$\mathbf{u}_{1:I}^{\star} = \underset{\{\mathbf{u}_i \in \Upsilon\}_{i=1}^{I}}{\arg\min}\ \mathbb{V}\left[\sum_{i=1}^{I}\sum_{m=1}^{M} \tilde{w}(\mathbf{x}_i^{(m)}, \boldsymbol{\theta}_i) \middle| \mathbf{u}_{1:I} \right], \tag{14}$$

where Υ is the feasibility set of $\mathbf{u}_i$. Unfortunately, such an objective function is generally difficult to optimize due to the difficulty in dealing with the set of deterministic auxiliary variables. Therefore, an alternative formulation should be considered. Our proposed alternative assumes the auxiliary variables as stochastic and distributed according to some parameterized probability distribution. The parameters of this distribution are optimized according to some optimality criterion related to the performance of the AIS sampler.

V. Policy Gradient Importance Sampling

We propose a novel class of samplers that frames AIS as a stochastic control problem. To this end, the proposal distribution in the AIS algorithm is treated as an agent, whose state corresponds to the value of its parameters. The agent takes actions using a parameterized policy that governs the adaptation of the proposal parameters at each iteration of the algorithm. Using this framework, we reformulate the objective in (14) to an optimization problem over the policy parameters.

Consider an AIS sampler which samples from a proposal distribution at the ith iteration using the following steps:

$$\mathbf{u}_i \sim \pi(\mathbf{u}_i|\boldsymbol{\theta}_{i-1};\boldsymbol{\psi}), \tag{15}$$

$$\boldsymbol{\theta}_i \sim h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i), \tag{16}$$

$$\mathbf{x}_i^{(m)} \sim q(\mathbf{x};\boldsymbol{\theta}_i), \quad m = 1,\dots,M, \tag{17}$$

for $i = 1,\dots,I$. The distribution $\pi(\mathbf{u}_i|\boldsymbol{\theta}_i;\boldsymbol{\psi})$ is called the *policy* and is parameterized by a set of parameters $\boldsymbol{\psi} \in \boldsymbol{\Psi}$. Define $\mathbf{u}_{1:I} \triangleq \{\mathbf{u}_1,\dots,\mathbf{u}_I\}$ and $\boldsymbol{\theta}_{0:I} \triangleq \{\boldsymbol{\theta}_0,\dots,\boldsymbol{\theta}_I\}$. The joint distribution $\beta_{\boldsymbol{\psi}}$ of $\mathbf{u}_{1:I}$ and $\boldsymbol{\theta}_{0:I}$ is then

$$\beta(\mathbf{u}_{1:I}, \boldsymbol{\theta}_{0:I};\boldsymbol{\psi}) = p(\boldsymbol{\theta}_0)\prod_{i=1}^{I} h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1},\mathbf{u}_i)\pi(\mathbf{u}_i|\boldsymbol{\theta}_{i-1};\boldsymbol{\psi}). \tag{18}$$

Algorithm 1 Policy Controlled Importance Sampling (PCIS)

```
**Input:** A parameterized policy $\pi(\mathbf{a}_i|\mathbf{s}_i;\boldsymbol{\psi})$ and a parameterized proposal distribution $q(\mathbf{x};\boldsymbol{\theta})$.
**Initialization:** Sample the initial proposal parameters according to $\boldsymbol{\theta}_0 \sim p(\boldsymbol{\theta}_0)$ and set $\mathbf{s}_1 = \boldsymbol{\theta}_0$.
**for** $i = 1$ **to** $I$ **do**
    1. Sample the auxiliary variable $\mathbf{u}_i \sim \pi(\mathbf{u}_i|\boldsymbol{\theta}_{i-1};\boldsymbol{\psi})$.
    2. Sample the proposal parameters $\boldsymbol{\theta}_i \sim h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1},\mathbf{u}_i)$.
    3. Set $\mathbf{a}_i = \mathbf{u}_i$ and $\mathbf{s}_{i+1} = \boldsymbol{\theta}_i$.
    4. Draw $M$ samples from the proposal distribution:
           $\mathbf{x}_i^{(m)} \sim q(\mathbf{x};\boldsymbol{\theta}_i), \quad m = 1,\dots,M.$
    5. Compute the unnormalized importance weights:
           $\tilde{w}_i^{(m)} = \frac{\tilde{\varphi}(\mathbf{x}_i^{(m)})}{q(\mathbf{x}_i^{(m)};\boldsymbol{\theta}_i)}, \quad m = 1,\dots,M.$
    6. Use samples and weights to estimate the reward $r_i$.
**end for**
**Return:** The random measures $\mathcal{X}_i = \{\mathbf{x}_i^{(m)}, \tilde{w}_i^{(m)}\}_{m=1}^M$ for all $i$ and the experience $\mathcal{E} = \{(\mathbf{s}_1,\mathbf{a}_1,r_1),\dots,(\mathbf{s}_I,\mathbf{a}_I,r_I)\}$.
```

In this context, the AIS agent's state corresponds to the proposal parameters at the previous iteration $\boldsymbol{\theta}_{i-1}$, while its action corresponds to the auxiliary variable $\mathbf{u}_i$ that determines the next proposal parameters $\boldsymbol{\theta}_i$. The time horizon over which which the AIS agent interacts with the environment corresponds number of iterations I of the AIS simulation. The initial proposal parameters $\boldsymbol{\theta}_0$ and the policy parameters $\boldsymbol{\psi}$ are what characterize the behavior of the sampler. Since we assume that $\boldsymbol{\theta}_0 \sim p(\boldsymbol{\theta}_0)$, then we can optimize this sampler by appropriate choice of the policy parameters $\boldsymbol{\psi}$. The optimal $\boldsymbol{\psi}$ maximize the so-called *expected return*, i.e.,

$$\mathcal{J}(\boldsymbol{\psi}) = \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\gamma^{i-1}R(\boldsymbol{\theta}_i)\right], \tag{19}$$

where $\gamma \in [0,1]$ is a discounting factor and $R : \Theta \to \mathbb{R}$ is a reward function used to award the agent for taking "favorable" actions. After choosing a discounting factor and a reward function, the policy parameters are obtained as

$$\boldsymbol{\psi}^\star = \underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\max}\, \mathcal{J}(\boldsymbol{\psi}). \tag{20}$$

The policy parameters are optimized using the episodic REINFORCE algorithm [27] in Algorithm 3, where experiences are collected using the *policy controlled importance sampling* (PCIS) algorithm shown in Algorithm 1. We provide a brief review of policy gradient methods in Appendix A.

A. Rewards based on Minimum Variance Optimality

In order to learn the optimal policy parameters, we require a definition for how the agent is awarded for the actions it takes. In the following proposition, we state an optimality criterion for choosing the policy parameters to minimize the variance of the AIS estimator of the evidence, i.e., $\hat{Z}_{\text{AIS}}^{M\times I}$.

PROPOSITION 1 [Minimum Variance for Policy Parameters in Adaptive Importance Sampling] *Consider the AIS method formulated in* (15)-(17). *The optimal policy parameters $\boldsymbol{\psi}^\star$ that minimize the variance of $Z_{\text{AIS}}^{M\times I}$ are those that solve the following optimization problem:*

$$\boldsymbol{\psi}^\star = \underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\max}\, \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\tilde{R}_V(\boldsymbol{\theta}_i)\right], \tag{21}$$

where $\tilde{R}_V(\boldsymbol{\theta}_i) = -\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]$.

The proof of Proposition 1 can be found in Appendix B. The proposition implies that minimizing the variance of $\hat{Z}_{\text{AIS}}^{M\times I}$ can be formulated as a RL problem, where the expected return $\mathcal{J}(\boldsymbol{\psi})$ is determined by a reward function $\tilde{R}_V(\boldsymbol{\theta}_i) = -\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]$ and a discounting factor that is $\gamma = 1$. We can expand the reward function $\tilde{R}_V(\boldsymbol{\theta}_i)$ as

$$\begin{aligned}\tilde{R}_V(\boldsymbol{\theta}_i) &= -\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]\\ &= -\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[(\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i))^2\right] + \Big(\mathbb{E}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]]\Big)^2\\ &= R_V(\boldsymbol{\theta}_i) + Z^2,\end{aligned}$$

where $R_V(\boldsymbol{\theta}_i) = -\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[(\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i))^2\right]$. Since $R_V(\boldsymbol{\theta}_i)$ is just a shifted version of $\tilde{R}_V(\boldsymbol{\theta}_i)$, one can also use $R_V(\boldsymbol{\theta}_i)$ as the reward function to find the minimum variance policy parameters. This choice is recommended in practice, since Z is an unknown quantity that will need to be estimated if one wishes to use $\tilde{R}_V(\boldsymbol{\theta}_i)$.

B. Rewards based on Divergence Measures

The maximization problem in (21) can be shown to be equivalent to minimizing a sum of (Pearson) χ^2 divergences. The χ^2 divergence from $\varphi(\mathbf{x})$ to $q(\mathbf{x};\boldsymbol{\theta}_i)$ is defined as [28]:

$$\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i}) = \int_{\mathcal{X}}\frac{\varphi(\mathbf{x})^2}{q(\mathbf{x};\boldsymbol{\theta}_i)}d\mathbf{x} - 1. \tag{22}$$

We formally state the result in the following proposition, which we prove in Appendix C of this text.

PROPOSITION 2 [Minimal Sum of Pearson χ^2 Divergences] *The maximization problem in* (21) *is equivalent to the following minimization problem:*

$$\boldsymbol{\psi}^\star = \underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\min}\, \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i})\right]. \tag{23}$$

The equivalent formulation in Proposition 2 gives the idea that reward functions should be related to a measure of discrepancy between the target and the proposal. Therefore, we can consider to solve the more general problem

$$\boldsymbol{\psi}^\star = \underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\min}\, \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\mathcal{D}_{\alpha}(\varphi\|q_{\boldsymbol{\theta}_i})\right], \tag{24}$$

where $\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i})$ denotes a discrepancy measure between the $\varphi(\mathbf{x})$ and $q(\mathbf{x};\boldsymbol{\theta}_i)$ parameterized by a real-valued parameter α. For example, we can consider a family of divergences, such as the Rényi divergence (of order α) between $\varphi(\mathbf{x})$ and $q(\mathbf{x};\boldsymbol{\theta}_i)$ [29] as the chosen discrepancy measure, which is given by

$$\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i}) = \tfrac{1}{\alpha-1}\log\left(\int_{\mathcal{X}}\left(\tfrac{\varphi(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^\alpha q(\mathbf{x};\boldsymbol{\theta}_i)d\mathbf{x}\right), \tag{25}$$

where $\alpha \in \mathbb{R}^+ \setminus \{1\}$. The case $\alpha = 2$ is a monotonic transformation of the χ^2 divergence defined in (22) [30], while in the limit we also have that $\lim_{\alpha\to 1} \mathcal{D}_\alpha(\varphi \| q_{\boldsymbol{\theta}})$ as the KLD between $\varphi(\mathbf{x})$ and $q(\mathbf{x}; \boldsymbol{\theta}_i)$.

Although one can straightforwardly define a reward function to convert the minimization problem in (24) into a maximization over expected return, i.e.,

$$R_\alpha(\boldsymbol{\theta}_i) = -\mathcal{D}_\alpha(\varphi \| q_{\boldsymbol{\theta}_i}), \tag{26}$$

it is desirable to choose a reward function that does not require to select a value of α. One possibility is to maximize a lower bound to (24). It turns out that there exists a reward function, whose corresponding expected return is a lower bound to the objective in (24). We state this result in the following proposition and provide a proof in Appendix D.

PROPOSITION 3 [Cumulative Evidence Lower Bound Maximization] *Consider the optimization problem in* (24) *for any* $\alpha > 1$. *Then, the maximization problem*

$$\boldsymbol{\psi}^\star = \underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\max}\, \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I} R_E(\boldsymbol{\theta}_i)\right], \tag{27}$$

where $R_E(\boldsymbol{\theta}_i) = \mathbb{E}_{q_{\boldsymbol{\theta}_i}}[\log \tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i)]$, *is equivalent to maximizing a lower bound to the objective in* (24).

The reward function $R_E(\boldsymbol{\theta}_i) = \mathbb{E}_{q_{\boldsymbol{\theta}_i}}[\log \tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i)]$ corresponds to the evidence lower bound (ELBO) as defined in [31] and is the function of interest in varitational inference methods. It is well known that maximizing $R_E(\boldsymbol{\theta}_i)$ is equivalent to minimizing the KLD from $q(\mathbf{x}; \boldsymbol{\theta}_i)$ to $\varphi(\mathbf{x})$:

$$\mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\theta}_i} \| \varphi) = \int_{\mathcal{X}} q(\mathbf{x}; \boldsymbol{\theta}_i) \log\left(\frac{q(\mathbf{x}; \boldsymbol{\theta}_i)}{\varphi(\mathbf{x})}\right) d\mathbf{x}. \tag{28}$$

Thus, maximizing the expected return obtained by the reward function $R_E(\boldsymbol{\theta}_i)$ is equivalent to minimizing the sum of KLDs. Importantly, this means that $R_E(\boldsymbol{\theta}_i)$ is a reward function that, like $R_V(\boldsymbol{\theta}_i)$, is related to a measure of discrepancy between the target and the proposal.

VI. IMPLEMENTATION

Here, we discuss the implementation details of the proposed algorithm. In particular, we discuss the policy choice considered in this work, how one can design the dynamics of the adaptation step in the algorithm, and the details on the estimation of the derived reward functions. We also discuss how the trained policy can be used for AIS methods which utilize more than one proposal. The proposed algorithm which uses multiple proposals is called *multiple policy controlled IS* (MPCIS) and is summarized in Algorithm 2

A. Choice of Parameterized Policy

For simplicity, we consider a linear and Gaussian policy

$$\pi(\mathbf{u}_i|\boldsymbol{\theta}_i; \boldsymbol{\psi}) = \mathcal{N}(\mathbf{u}_i; \mathbf{A}\phi_1(\boldsymbol{\theta}_i), \exp(\mathbf{b}^\top \phi_2(\boldsymbol{\theta}_i))\mathbf{I}_{d_u}) \tag{29}$$

where $\phi_1 : \mathbb{R}^{d_u} \to \mathbb{R}^{d_{\phi_1}}$ and $\phi_2 : \mathbb{R}^{d_u} \to \mathbb{R}^{d_{\phi_2}}$ are basis functions used to transform the parameter $\boldsymbol{\theta}_i$, and the policy parameters are $\boldsymbol{\psi} = \{\mathbf{A}, \mathbf{b}\}$ with $\mathbf{A} \in \mathbb{R}^{d_\theta \times d_{\phi_1}}$ and $\mathbf{b} \in \mathbb{R}^{d_{\phi_2}}$. To optimize this policy using a PG algorithm, we need access to the gradients $\nabla_{\mathbf{A}} \log \pi(\mathbf{u}_i|\boldsymbol{\theta}_i; \boldsymbol{\psi})$ and $\nabla_{\mathbf{b}} \log \pi(\mathbf{u}_i|\boldsymbol{\theta}_i; \boldsymbol{\psi})$. These gradients can be computed using matrix calculus and are given by the following expressions:

$$\nabla_{\mathbf{A}} \log \pi(\mathbf{u}_i|\boldsymbol{\theta}_i; \boldsymbol{\psi}) = \frac{\mathbf{z}_i \phi_1(\boldsymbol{\theta}_i)^\top}{\exp(\mathbf{b}^\top \phi_2(\boldsymbol{\theta}_i))}, \tag{30}$$

$$\nabla_{\mathbf{b}} \log \pi(\mathbf{u}_i|\boldsymbol{\theta}_i; \boldsymbol{\psi}) = \frac{\phi_2(\boldsymbol{\theta}_i)}{2}\left(\frac{\mathbf{z}_i^\top \mathbf{z}_i}{\exp(\mathbf{b}^\top \phi_2(\boldsymbol{\theta}_i))} - d_u\right), \tag{31}$$

where $\mathbf{z}_i = \mathbf{u}_i - \mathbf{A}\phi_1(\boldsymbol{\theta}_i)$. While the choice of $\phi_1(\boldsymbol{\theta}_i)$ and $\phi_2(\boldsymbol{\theta}_i)$ impacts the performance of the algorithm, for simplicity we consider the linear basis $\phi_1(\boldsymbol{\theta}_i) = \phi_2(\boldsymbol{\theta}_i) = [1, \boldsymbol{\theta}_i^\top]^\top$. Remark that the mean and covariance in (29) are not required to be linear and one could, for example, use a more complex function like a deep neural network to model them [32].

B. Choice of State Dynamics

In most applications where RL methods are employed, the dynamics that define the system of interest are fixed and cannot be controlled. In framing AIS as an RL problem, we can actually decide what type of dynamics we would like our proposal parameters to follow. In other words, the practitioner can choose the distribution $h(\boldsymbol{\theta}_{i+1}|\boldsymbol{\theta}_i, \mathbf{u}_i)$ as they see fit. We note that the choice of $h(\boldsymbol{\theta}_{i+1}|\boldsymbol{\theta}_i, \mathbf{u}_i)$ has heavy influence on the capabilities of the proposed scheme. One possible choice is to consider linear Gaussian dynamics, i.e.,

$$\boldsymbol{\theta}_{i+1} = \mathbf{C}\boldsymbol{\theta}_i + \mathbf{D}\mathbf{u}_i + \boldsymbol{\nu}_i, \tag{32}$$

with $\mathbf{C} \in \mathbb{R}^{d_\theta \times d_\theta}$, $\mathbf{D} \in \mathbb{R}^{d_\theta \times d_u}$, and $\boldsymbol{\nu}_i \sim \mathcal{N}(\mathbf{0}_{d_\theta}, \nu^2 \mathbf{I}_{d_\theta})$. We remark that linear and Gaussian dynamics do not use any information about the geometry of the target distribution $\varphi(\mathbf{x})$. Alternatively, one could consider alternative dynamics inspired by Langevin diffusion which take into account the gradient of $\log \varphi(\mathbf{x})$ [33], [34]; however, for simplicity in presentation, we will investigate such dynamics in an extension of this work. In the following, we discuss a joint learning strategy of the parameters of the dynamics and the policy using a simple modification of Algorithm 3.

C. Learning the Dynamics

Under the assumption that the dynamics are parameterized by $\boldsymbol{\xi} = \{\mathbf{C}, \mathbf{D}\}$ as they are in (32), then the joint distribution of $\mathbf{u}_{1:I}$ and $\boldsymbol{\theta}_{0:I}$ now depends on $\boldsymbol{\xi}$, i.e.,

$$\beta(\mathbf{u}_{1:I}, \boldsymbol{\theta}_{0:I}; \boldsymbol{\psi}, \boldsymbol{\xi}) = p(\boldsymbol{\theta}_0) \textstyle\prod_{i=1}^{I} h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi})\pi(\mathbf{u}_i|\boldsymbol{\theta}_{i-1}; \boldsymbol{\psi}). \tag{33}$$

Applying the log derivative trick in the same way as it is applied for $\boldsymbol{\psi}$, we can also employ the REINFORCE algorithm to optimize the parameters $\boldsymbol{\xi}$:

$$\nabla_{\boldsymbol{\xi}} \log h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi}) = \frac{\nabla_{\boldsymbol{\xi}} h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi})}{h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi})}. \tag{34}$$

When the dynamics are linear and Gaussian as they are in (32), then the log gradients can be computed as

$$\nabla_{\mathbf{C}} \log h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi}) = \tfrac{1}{\nu^2}(\boldsymbol{\theta}_i - \mathbf{C}\boldsymbol{\theta}_{i-1} - \mathbf{D}\mathbf{u}_i)\boldsymbol{\theta}_{i-1}^\top, \tag{35}$$

$$\nabla_{\mathbf{D}} \log h(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{u}_i; \boldsymbol{\xi}) = \tfrac{1}{\nu^2}(\boldsymbol{\theta}_i - \mathbf{C}\boldsymbol{\theta}_{i-1} - \mathbf{D}\mathbf{u}_i)\mathbf{u}_i^\top. \tag{36}$$

Given the gradients in (35) and (36), one can use the collected AIS experiences to optimize the parameters of the dynamics along with the parameters of the policy are optimized.

D. Estimation of Rewards

Unfortunately, we cannot evaluate the considered reward functions analytically since they involve integrals that are generally intractable. One can instead compute unbiased or consistent estimates in place of the true reward function using the samples and weights obtained at each iteration of the sampler in Algorithm 1. For instance, an unbiased estimator of $R_E(\boldsymbol{\theta}_i)$ can be determined using just the samples drawn in the ith iteration of the algorithm,

$$R_E(\boldsymbol{\theta}_i) \approx \frac{1}{M}\sum_{m=1}^{M} \log \tilde{w}_i^{(m)}, \tag{37}$$

since $R_E(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}_i}}[\log \tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i)]$ can be written as an expectation w.r.t. $q(\mathbf{x}; \boldsymbol{\theta}_i)$. One can obtain a consistent estimate of $R_V(\boldsymbol{\theta}_i)$ (up to a scaling factor) by expressing $R_V(\boldsymbol{\theta}_i)$ as follows:

$$\begin{aligned}
R_V(\boldsymbol{\theta}_i) &= -\mathbb{E}_{q_{\boldsymbol{\theta}_i}}[(\tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i))^2] \\
&= -\int_{\mathcal{X}} \left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x}; \boldsymbol{\theta}_i)}\right)^2 q(\mathbf{x}; \boldsymbol{\theta}_i) d\mathbf{x} \\
&= -\int_{\mathcal{X}} \left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x}; \boldsymbol{\theta}_i)}\right) \tilde{\varphi}(\mathbf{x}) d\mathbf{x} \\
&= -Z\int_{\mathcal{X}} \left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x}; \boldsymbol{\theta}_i)}\right) \varphi(\mathbf{x}) d\mathbf{x} \\
&\propto \mathbb{E}_{\varphi}[\tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i)].
\end{aligned}$$

Since $R_V(\boldsymbol{\theta}_i)$ can also be expressed in terms of an expectation taken w.r.t. $\varphi(\mathbf{x})$, we can use the self-normalized IS estimator in (6) to obtain the following estimate:

$$R_V(\boldsymbol{\theta}_i) \propto \mathbb{E}_{\varphi}[\tilde{w}(\mathbf{x}, \boldsymbol{\theta}_i)] \approx -\frac{1}{\sum_{j=1}^{M} \tilde{w}_i^{(j)}} \sum_{m=1}^{M} (\tilde{w}_i^{(m)})^2. \tag{38}$$

As a remark, we point out that using the estimator in (38), which only approximates the reward $R_V(\boldsymbol{\theta}_i)$ up to a scaling factor, is acceptable, since multiplication by a scaling factor does not change the solution to optimization problem in (21).

E. Penalized Reward Functions

It is possible to modify the reward functions to include a penalty term to improve the performance of the optimized policy/dynamics. Recall that reward function $R_E(\boldsymbol{\theta}_i)$ is the ELBO, whose optimization in variational inference has shown to lead to posterior approximations with underestimated variance. Therefore, the use of $R_E(\boldsymbol{\theta}_i)$ in the proposed method could lead detrimental performance, as it can lead to a proposal distribution with thinner tails than the target distribution. One way to mitigate this effect is to consider a penalized version of this reward function

$$R_{E,S}^{\lambda}(\boldsymbol{\theta}_i) = R_E(\boldsymbol{\theta}_i) - \lambda S(\boldsymbol{\theta}_i), \tag{39}$$

where $S(\boldsymbol{\theta}_i)$ is a penalty function and $\lambda > 0$ is the strength of the penalty. The penalty should be to chosen to encourage policies/dynamics which yield to suitable proposals that will not jeopardize the performance of the AIS method. One possibility is to introduce a penalty based on the norm of the precision matrix of the proposal distribution. Penalizing large precision matrices ensures that the majority of the probability mass of the proposal covers a wider support, hence leading to proposal with fatter tails than the target distribution. Another possibility is to use the Pareto-smoothed IS (PSIS) diagnostic presented in [35], which corresponds to the shape parameter of a generalized Pareto distribution fit to the importance weights. Indeed, this diagnostic has been used as a means to determine the reliability of variational approximations [36], and so it is suitable for our purposes.

Algorithm 2 Multiple PCIS (MPCIS)

```
1: Initialization: Set the initial proposal parameters by drawing from the prior $\boldsymbol{\theta}_{1,0}, \dots, \boldsymbol{\theta}_{D,0} \sim p(\boldsymbol{\theta}_0)$.
2: for $i = 1, \dots, I$ do
3:     for $d = 1, \dots, D$ do
4:         Sample the auxiliary variable of the $d$th proposal:
               $\mathbf{u}_{d,i} \sim \pi(\mathbf{u}_{d,i} | \boldsymbol{\theta}_{d,i-1}; \boldsymbol{\psi})$.
5:         Sample the parameters of the $d$th proposal:
               $\boldsymbol{\theta}_{d,i} \sim h(\boldsymbol{\theta}_{d,i} | \boldsymbol{\theta}_{d,i-1}, \mathbf{u}_{d,i}; \boldsymbol{\xi})$
6:     end for
7:     Draw $K$ samples from each proposal:
           $\mathbf{x}_i^{(d,k)} \sim q(\mathbf{x}; \boldsymbol{\theta}_{d,i}), \quad k = 1, \dots, K, \quad d = 1, \dots, D.$
8:     Compute the deterministic mixture weights:
           $\tilde{w}_i^{(d,k)} = \frac{\tilde{\varphi}(\mathbf{x}_i^{(d,k)})}{\frac{1}{D}\sum_{d=1}^{D} q(\mathbf{x}_i^{(d,k)}; \boldsymbol{\theta}_{d,i})}, \quad k = 1, \dots, K, \quad d = 1, \dots, D.$
9: end for
10: Output: Return $\mathcal{X}_{d,i} = \{\mathbf{x}_i^{(d,k)}, \tilde{w}_i^{(d,k)}\}_{k=1}^{K}$ for all $i$ and for all $d$.
```

F. Multiple Proposal Implementation

After training, the learned policy and dynamics can be used to adapt a population of proposal distributions. We call this multiple proposal implementation summarized in Algorithm 2 the *multiple policy controlled importance sampling* (MPCIS). At the ith iteration of the algorithm, the parameters of the dth proposal, denoted $\boldsymbol{\theta}_{d,i}$, are determined using the trained policy and dynamics. After determining the parameters for each proposal, MPCIS draws K samples from the D proposal distributions for a total of $M = DK$ samples. The importance weights are computed according to the *deterministic mixture* (DM) approach [37], which weighs the influence of all proposal distributions. The use of the DM weighting scheme is common in most robust AIS implementations and typically leads to lower variance IS estimators in practice (see [14], [38], [39] for other examples). Finally, the collection of all samples and weights can be used to compute IS estimators.

G. Distributed Computing

A legitimate concern regarding the implementation of the proposed methodology centers around its computation time.

Unlike standard AIS methods, where the learning phase takes place in a single simulation of the algorithm, episodic REINFORCE requires running a multitude of simulations to learn a policy. In particular, at each iteration of episodic REINFORCE, N simulations of PCIS are run to estimate the gradient of the policy parameters $\boldsymbol{\psi}$. Luckily, since these simulations do not interact, we can leverage the power of distributed computing to obtain them. Therefore, a strong advantage of using RL to train an AIS algorithm is that a computational cluster can be utilized to speed up the training process. The learning phase of standard AIS approaches are typically sequential, and therefore, cannot take advantage of distributed computing.

VII. Examples

We demonstrated the performance of the proposed algorithm through two numerical experiments. The first was a multi-dimensional twisted Gaussian target distribution and the second was on a Bayesian vector autoregressive (VAR) model. In both examples, we compared the proposed method to the following AIS algorithms: population MC (PMC) [40], stochastic gradient PMC (SG-PMC) [41], convex adaptive MC (CAMC) [12], and variational adaptive population IS (VAPIS) [13]. PMC is an AIS algorithm that uses multinomial resampling to resolve the adaptation of location parameters for a population of proposal distributions equal to the number of samples being drawn. SG-PMC is a variant of PMC which uses resampling to explicitly solve an optimization problem related to a moment-matching criterion. Both methods employ the use of DM weights [37], [42], which weighs samples according to the population of proposals to obtain lower variance estimators. CAMC and VAPIS methods use stochastic optimization to explicitly solve (9), where the SGD updates are used as the proposal parameters. VAPIS specifically optimizes the parameters of an equally weighted mixture distribution.

For training, we used episodic REINFORCE to optimize the parameters of the dynamics and policy using collected experiences from running PCIS with $M = 250$ samples and $I = 40$ iterations. We chose a multivariate Gaussian distribution as the proposal distribution in PCIS, i.e., $q(\mathbf{x};\boldsymbol{\theta}_i) = \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$, where $\boldsymbol{\mu}_i$ denotes the mean and $\boldsymbol{\Sigma}_i$ denotes the covariance matrix. We considered two options for adapting the proposal parameters:

1) Mean and isotropic covariance matrix, i.e., $\boldsymbol{\theta}_i = [\boldsymbol{\mu}_i, \sigma_i^2]^\top$ with $\boldsymbol{\Sigma}_i = \sigma_i^2\mathbf{I}_{d_x}$. Here, $d_\theta = d_x + 1$.
2) Mean with diagonal covariance matrix, i.e., $\boldsymbol{\theta}_i = [\boldsymbol{\mu}_i, \sigma_{1,i}^2, \dots, \sigma_{d_x,i}^2]^\top$ with $\boldsymbol{\Sigma}_i = \operatorname{diag}(\sigma_{1,i}^2, \dots, \sigma_{d_x,i}^2)$. Here, $d_\theta = 2d_x$.

The REINFORCE algorithm was run to train the linear Gaussian policy and linear dynamics of the sampler until the criterion $|\mathcal{J}(\boldsymbol{\psi}_j) - \mathcal{J}(\boldsymbol{\psi}_{j-1})| < 0.005$ was satisfied or until had trained for $J = 5000$ gradient steps. The high-performance SeaWulf computing system was used to simultaneously collect $N = 240$ simulations at each REINFORCE iteration to estimate the gradient of the policy parameters. In each of these simulations, the variance of the dynamics was $\nu^2 = 0.05$. After training the policy, the MPCIS algorithm was run with the trained policy/dynamics to compare to the other methods, where the variance of the dynamics was $\nu^2 = 10^{-6}$. A larger variance was used for training so that the agent could efficiently explore the state-space (exploration), while a smaller variance was used after the policy is trained to exploit the best possible sequence of proposal parameters (exploitation).

A. Multi-Dimensional Twisted Gaussian Target

We considered that $\mathbf{x} = [x_1, \dots, x_{d_x}] \in \mathbb{R}^{d_x}$, where $d_x \geq 2$ and the unnormalized target distribution was defined as

$$\tilde{\varphi}(\mathbf{x}) = \mathcal{N}(x_1 + a_1(x_2^2 + a_2^2); 0, a_2^2) \prod_{d=2}^{d_x} \mathcal{N}(x_d; 0, 1), \quad (40)$$

where $\mathcal{N}(x;\mu,\sigma^2)$ denotes a univariate Gaussian probability density function with mean μ and variance σ^2 evaluated at x and $a_1, a_2 \in \mathbb{R}$. Such a density is obtained from applying a transformation of a Gaussian random vector [43] and results in a banana-shaped distribution with normalization constant $Z = 1$. For a given choice of $a_1, a_2 \in \mathbb{R}$, the mean is

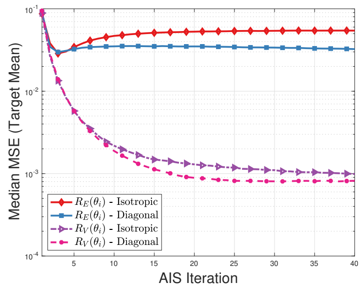
$$\mathbb{E}_\varphi[\mathbf{x}] = [-a_1(1 + a_2^2), 0, \dots, 0]^\top.$$

In this example, we set $a_1 = a_2 = 1$. The considered performance metrics in this example were the median mean-square error (MSE) of the target mean and median absolute error (MAE) of the normalization constant, which are obtained by running 1000 independent Monte Carlo simulations.

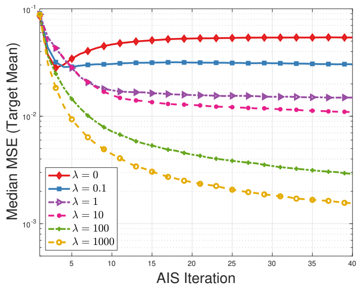
Figure 1 summarizes the performance of PCIS for the twisted Gaussian target when $d_x = 2$ after the policy and dynamics were trained. Figures. 1a and 1d show a comparison of trained policies for the different reward functions and adaptation schemes, where it is clear that the best trained policy corresponds to the one where the reward function used was $R_V(\boldsymbol{\theta}_i)$ and a diagonal covariance matrix was considered. We note that the policy trained with the reward function $R_E(\boldsymbol{\theta}_i)$ performed poorly since it led to a proposal that underestimated the variance of the target, as postulated in Section VI-E. To resolve this issue, we considered a penalized reward in (39), where the penalty $S(\boldsymbol{\theta}_i)$ corresponded to the squared Frobenius norm of the precision matrix of the isotropic proposal, i.e.,

$$S(\boldsymbol{\theta}_i) = \|\boldsymbol{\Sigma}_i^{-1}\|_F^2 = \left(\frac{1}{\sigma_i^2}\right)^2. \quad (41)$$

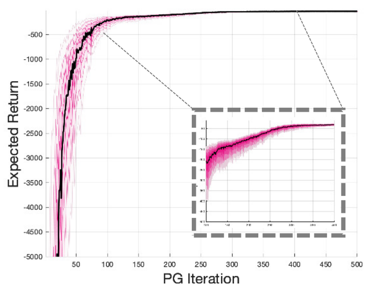
Figures 1b and 1e show the performance of policies trained using the penalized reward function $R_{E,S}^\lambda(\boldsymbol{\theta}_i)$ for different values of λ. The use of the penalty improved the performance of the sampler, where for $\lambda = 1000$, the performance of a policy trained with $R_{E,\lambda}(\boldsymbol{\theta}_i)$ had similar performance to a policy trained with the reward $R_V(\boldsymbol{\theta}_i)$. Furthermore, it was evident that for this particular example, larger values of λ were necessary to obtain improved performance. In general, it is difficult to choose the value of λ, as it will depend on the properties of the target distribution (e.g., tail behavior, skewness, and shape) and the scaling of the reward function. Therefore, we recommend that for general problems, a cross-validation scheme should be used to choose λ. Finally, Figs.



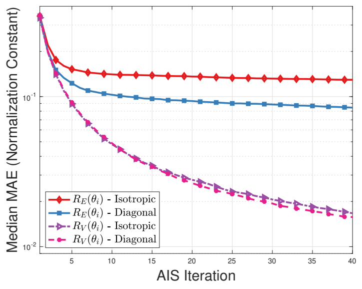
(a)



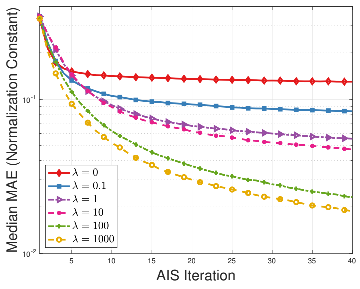
(b)



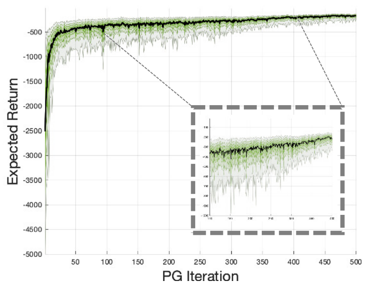
(c)



(d)

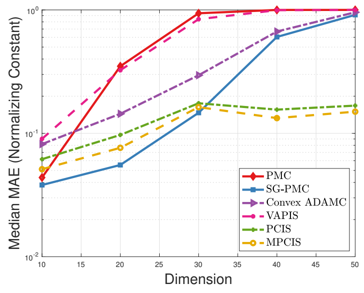


(e)

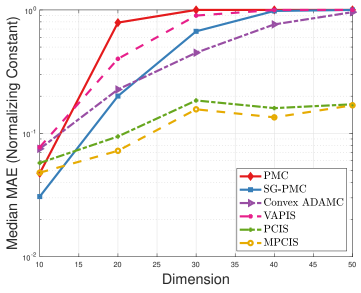


(f)

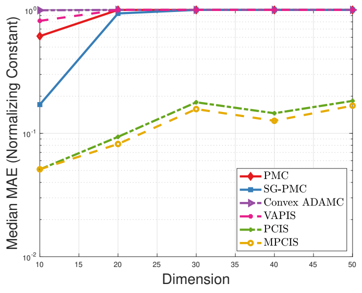
Fig. 1: Performance of the proposed method for the two-dimensional twisted Gaussian target.



(a) $\sigma_0^2 = 1$



(b) $\sigma_0^2 \sim \mathcal{U}(1, 10)$



(c) $\sigma_0^2 \sim \mathcal{U}(1, 100)$

Fig. 2: Performance of the proposed method for the multi-dimensional twisted Gaussian target (of varying dimension).

1c and 1f depict the evolution of the expected return during training of the policies for the $R_E(\boldsymbol{\theta}_i)$ and $R_V(\boldsymbol{\theta}_i)$ reward functions, respectively. The expected return converged faster when the considered reward function was $R_E(\boldsymbol{\theta}_i)$. This was because the variance of the estimated reward for $R_E(\boldsymbol{\theta}_i)$ was significantly less than that of the estimated reward for $R_V(\boldsymbol{\theta}_i)$. This is because, unlike $R_V(\boldsymbol{\theta}_i)$, the reward function $R_E(\boldsymbol{\theta}_i)$ is only a function of the logarithm of the importance weights. This makes it more suitable for high-dimensional target distributions, where numerical underflow can be an issue for the importance weights. Therefore, it is advantageous to train based on the penalized reward function $R^{\lambda}_{E,S}(\boldsymbol{\theta}_i)$ when considering high-dimensional target distributions, since it is numerically stable and one can obtain similar performance to the reward function $R_V(\boldsymbol{\theta}_i)$.

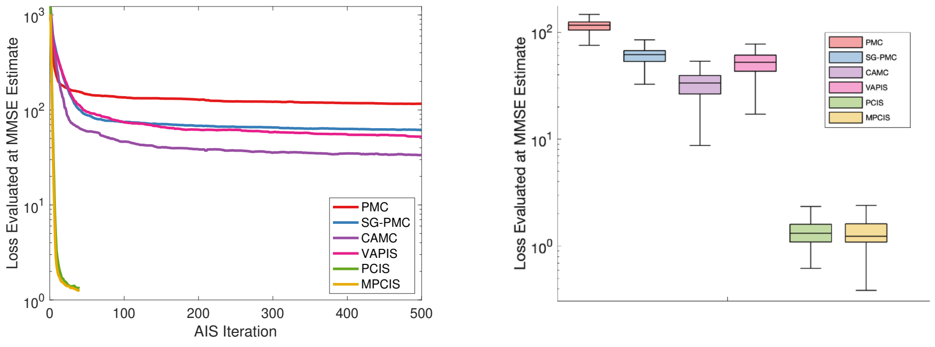
Figure 2 shows the performance of the proposed algorithm for a trained policy using the reward function $R^{100}_{E,S}(\boldsymbol{\theta}_i)$ as compared to other AIS methods under different initialization of an isotropic covariance matrix $\boldsymbol{\Sigma}_0 = \sigma_0^2 \mathbf{I}_{d_x}$. For each method, we draw $M = 250$ samples over $I = 400$ iterations, where for the VAPIS and MPCIS, we use a population of $D = 10$ proposals each drawing $K = 25$ samples. We can see that PCIS and MPCIS outperform the competing algorithms as the dimension of the target increases. Importantly, the performance of both PCIS and MPCIS is guaranteed regardless of the initialization scheme for the isotropic covariance matrix, while the other methods perform worse for broader initialization and thus are less robust for arbitrary targets. Finally, we remark that because MPCIS uses multiple proposals and a DM weighting scheme, it slightly outperforms the PCIS algorithm.

B. Vector Autoregression with Sparse Interactions

We considered a time series model, which described the time evolution of observations $\mathbf{y}_n \in \mathbb{R}^{d_y}$ according to:

$$\mathbf{y}_n = \boldsymbol{\Omega}\mathbf{y}_{n-1} + \mathbf{v}_n, \quad n = 2, \ldots, N, \tag{42}$$

where $\boldsymbol{\Omega} \in \mathbb{R}^{d_y \times d_y}$, $\mathbf{y}_1 \sim p(\mathbf{y}_1)$ and $\mathbf{v}_n \sim \mathcal{N}(\mathbf{0}_{d_y}, \sigma_v^2 \mathbf{I}_{d_y})$. Regardless of the distribution of the initial observation $p(\mathbf{y}_1)$,



(a) (b)

Fig. 3: Performance of the proposed method for the VAR model.

one can straightforwardly show that the likelihood function $p(\mathbf{y}_{1:N}|\mathbf{\Omega})$ is proportional to a matrix normal distribution

$$p(\mathbf{y}_{1:N}|\mathbf{\Omega}) \propto \mathcal{MN}(\mathbf{\Omega}; \mathbf{M}_{\mathbf{\Omega}}, \mathbf{U}_{\mathbf{\Omega}}, \mathbf{V}_{\mathbf{\Omega}}) \tag{43}$$

$$= \frac{\exp\left(-\frac{1}{2}\operatorname{tr}\left[\mathbf{V}_{\mathbf{\Omega}}^{-1}(\mathbf{\Omega}-\mathbf{M}_{\mathbf{\Omega}})^\top \mathbf{U}_{\mathbf{\Omega}}^{-1}(\mathbf{\Omega}-\mathbf{M}_{\mathbf{\Omega}})\right]\right)}{\sqrt{(2\pi)^{d_y^2}|\mathbf{V}_{\mathbf{\Omega}}|^{d_y}|\mathbf{U}_{\mathbf{\Omega}}|^{d_y}}} \tag{44}$$

where the location matrix $\mathbf{M}_{\mathbf{\Omega}}$, and the scale matrices $\mathbf{U}_{\mathbf{\Omega}}$ and $\mathbf{V}_{\mathbf{\Omega}}$ are given by

$$\mathbf{M}_{\mathbf{\Omega}} = \left(\sum_{t=2}^{T} \mathbf{y}_{t-1}\mathbf{y}_t^\top\right)\left(\sum_{t=2}^{T} \mathbf{y}_{t-1}\mathbf{y}_{t-1}^\top\right)^{-1}, \tag{45}$$

$$\mathbf{U}_{\mathbf{\Omega}} = \sigma_v^2\left(\sum_{t=2}^{T} \mathbf{y}_{t-1}\mathbf{y}_{t-1}^\top\right), \quad \mathbf{V}_{\mathbf{\Omega}} = \mathbf{I}_{d_y}. \tag{46}$$

Defining $\mathbf{x} = \operatorname{vec}(\mathbf{\Omega}) \in \mathbb{R}^{d_x}$, where $\operatorname{vec}(\cdot)$ denotes vectorization and $d_x = d_y^2$, the likelihood of $\mathbf{x}$ is proportional to a multivariate Gaussian distribution

$$p(\mathbf{y}_{1:N}|\mathbf{x}) \propto \mathcal{N}(\mathbf{x}; \tilde{\boldsymbol{\mu}}, \tilde{\mathbf{\Sigma}}), \tag{47}$$

where $\tilde{\boldsymbol{\mu}} = \operatorname{vec}(\mathbf{M}_{\mathbf{\Omega}})$ and $\tilde{\mathbf{\Sigma}} = \mathbf{U}_{\mathbf{\Omega}} \otimes \mathbf{V}_{\mathbf{\Omega}}$ with $\otimes$ denoting the Kronecker product. Our interest was in obtaining a sparse estimation of $\mathbf{x}$ under the Bayesian paradigm. This amounted to obtaining the posterior distribution of $\mathbf{x}$ given $\mathbf{y}_{1:N}$ under the choice of a sparse inducing prior $p(\mathbf{x}) \propto \prod_{k=1}^{d_x} \exp(-\frac{|x_k|}{2b})$, where b was considered to be a fixed hyperparameter. Thus, for this example, the unnormalized posterior $\tilde{\varphi}(\mathbf{x})$ was

$$\tilde{\varphi}(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \tilde{\boldsymbol{\mu}}, \tilde{\mathbf{\Sigma}}) \prod_{k=1}^{d_x} \exp\left(-\frac{|x_k|}{2b}\right). \tag{48}$$

Our goal was to compute the minimum mean-square error (MMSE) estimator of $\mathbf{x}$. We compared the performance of the MPCIS with the other AIS methods through evaluation of a loss function $L(\mathbf{x}) = -\log \tilde{\varphi}(\mathbf{x})$ at the MMSE estimate obtained from each algorithm. The policy/dynamics used in the MPCIS scheme were obtained from training with a penalized reward function, where the penalty $S(\boldsymbol{\theta}_i)$ was the PSIS diagnostic and $\lambda = 10$. We drew $M = 500$ samples per iteration, where the VAPIS and MPCIS methods used a population of $D = 10$ proposals each drawing $K = 50$ samples.

Figure 3a shows the evolution of the loss function, where the competing algorithms were run for ten times more iterations than PCIS and MPCIS. The results demonstrate that they both outperformed all competing algorithms by nearly two orders of magnitude. Furthermore, it is evident that even if the competing algorithms were run for many more iterations, the MPCIS algorithm would still outperform them. This is also evident from the box plot in Fig. 3b, which shows the loss for each method at the final iteration.

VIII. Conclusions and Future Directions

In this work, we proposed a novel adaptive importance sampling (AIS) algorithm that uses reinforcement learning (RL) to obtain a minimum variance estimator of the evidence in a given Bayesian model. In this framework, the proposal distribution in the AIS sampler is treated as an RL agent, where the state of the agent are the proposal parameters and the actions are the auxiliary variables used for adaptation in the sampler. We showed that suitable reward functions are related to divergence measures between the target distribution and the proposal distribution. Moreover, we showed the advantages of considered penalized rewards, where the imposed penalty can be related to importance sampling diagnostic measures. An intriguing advantage of the proposed methodology is that it is able to exploit distributed computing in its learning phase, something that cannot easily be done with standard AIS methods. Simulation results on two examples demonstrated the superior performance of the proposed method as compared to other AIS algorithms, and offers the prospect of its application to complex Bayesian models.

There are a number of possible future directions for this line of work. For instance, an interesting prospect is the use of more complex policies and dynamics, such as a ones defined by a neural network. This could give the proposed approach more flexibility and allow it to deal with target distributions that are nonlinear and/or non-Gaussian (e.g., multimodal distributions). Another opportunity is to include meta-learning in this framework [44]. By adding a meta-learning procedure to the framework, one can earn a policy for one target distribution (initial task) and then reuse that same policy on a different, but similar target distribution (new task). This would be incredibly useful in the Bayesian setting, where one may wish to obtain the posterior distribution for the parameters of one type of probabilistic model for a variety of different datasets. Finally, the extension of the proposed work to AIS methods that learn multiple proposals and use unique weighting schemes (e.g., temporal deterministic mixture weights [14]) is something that ought to be explored in the future.

Appendix A: Review of Policy Gradient Methods

In this section, we review the basic RL problem and describe a class of methods called PG methods [45], which are able to directly learn a locally optimal policy of an RL agent.

In RL, an agent interacts with an environment over a time horizon I. At each instant i, the agent at state $\mathbf{s}_i \in \mathcal{S}$ takes an action $\mathbf{a}_i \in \mathcal{A}$ and receives a reward $r_i \triangleq R(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}_{i+1}) \in \mathbb{R}$ according to a reward function $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$. Given a state $\mathbf{s}_i$ and an action $\mathbf{a}_i$, the next state $\mathbf{s}_{i+1}$ is determined by a state-transition distribution $\mathbf{s}_{i+1} \sim h(\mathbf{s}_{i+1}|\mathbf{s}_i, \mathbf{a}_i)$ that depends on the environment the agent interacts with. An agent takes actions according to a policy $\pi_{\boldsymbol{\psi}}$, which we assume is parameterized by some $\boldsymbol{\psi} \in \boldsymbol{\Psi}$. The policy can be deterministic, implying that $\pi_{\boldsymbol{\psi}} : \mathcal{S} \to \mathcal{A}$ is a deterministic function mapping from states to actions. The policy can also be stochastic. If the action space is continuous and the policy is stochastic, then the policy is represented by a conditional probability density function $\pi_{\boldsymbol{\psi}} : \mathcal{A} \times \mathcal{S} \to [0, \infty)$ mapping from the action space to the positive reals. The joint distribution of states and actions over I steps is

$$\beta(\mathbf{s}_{1:I+1}, \mathbf{a}_{1:I}; \boldsymbol{\psi}) = p(\mathbf{s}_1) \textstyle\prod_{i=1}^{I} h(\mathbf{s}_{i+1}|\mathbf{s}_i, \mathbf{a}_i)\pi(\mathbf{a}_i|\mathbf{s}_i; \boldsymbol{\psi}), \quad (49)$$

where $p(\mathbf{s}_1)$ is the distribution of the initial state. The objective in RL is to find a policy that maximizes the *long-term reward* or *return*, which is defined as follows:

$$G_{1:I} \triangleq \sum_{i=1}^{I} \gamma^{i-1} r_i \quad (50)$$

where $\gamma \in [0, 1]$ is a discounting factor. Since the policy, environment and reward function can be stochastic, the goal amounts to maximizing the agent's expected return, which is equivalent to determining the policy parameters $\boldsymbol{\psi}^\star$ that solves the following optimization problem:

$$\boldsymbol{\psi}^\star = \arg\max_{\boldsymbol{\psi} \in \boldsymbol{\Psi}} \mathcal{J}(\boldsymbol{\psi}), \quad (51)$$

where $\mathcal{J}(\boldsymbol{\psi}) \triangleq \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I} \gamma^{i-1} r_i\right]$ is defined to be the *expected return* and is a function of $\boldsymbol{\psi}$ since the policy itself depends on $\boldsymbol{\psi}$. The optimization problem in (51) is solved using a set of collected experiences, where each experience $\mathcal{E}_j$ is a sequence of observed states, actions, and rewards.

PG methods aim to learn a locally optimal policy directly by using a set of collected experiences. Here, the policy is assumed to be stochastic and have parametric form $\pi(\mathbf{a}_i|\mathbf{s}_i; \boldsymbol{\psi})$, where SGD is used to update the policy parameters. Gradient estimators with reduced variance can be obtained by application of the *policy gradient theorem* (PGT) [46], which exploits the fact that future actions do not depend on rewards incurred in the past. Let $\{\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(N)}\}$ be a batch of RL experiences collected using a policy $\pi(\mathbf{a}_i|\mathbf{s}_i; \boldsymbol{\psi})$, where each $\mathcal{E}^{(n)}$ is a set of tuples comprised of states, actions, and rewards, i.e., $\mathcal{E}^{(n)} = \{(\mathbf{s}_1^{(n)}, \mathbf{a}_1^{(n)}, r_1^{(n)}), \ldots, (\mathbf{s}_I^{(n)}, \mathbf{a}_I^{(n)}, r_I^{(n)})\}$. Given these experiences, the PGT gradient estimator is given by

$$g_{\text{PGT}}(\boldsymbol{\psi}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{I} (G_{i:I}^{(n)} - b) \nabla_{\boldsymbol{\psi}} \log \pi(\mathbf{a}_i^{(n)}|\mathbf{s}_i^{(n)}; \boldsymbol{\psi}), \quad (52)$$

where $G_{i:I}^{(n)} = \sum_{k=i}^{I} \gamma^{i-k} r_k^{(n)}$ is the return from step i, $b \in \mathbb{R}$ is a baseline, and $g_{\text{PGT}}(\boldsymbol{\psi})$ satisfies $\mathbb{E}\left[g_{\text{PGT}}(\boldsymbol{\psi})|\boldsymbol{\psi}\right] = \nabla_{\boldsymbol{\psi}} \mathcal{J}(\boldsymbol{\psi})$. The baseline is used as means to further reduce the variance of the gradient estimator. A simple heuristic for choosing the baseline is the *average reward* [47], which for the collected experiences $\{\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(N)}\}$ is given by:

$$b = \frac{1}{N} \sum_{n=1}^{N} \left(\frac{1}{I} \sum_{i=1}^{I} \gamma^{i-1} r_i^{(n)} \right). \quad (53)$$

Episodic REINFORCE [27] is an example of a PG method that can use the gradient estimator in (52) to learn the policy parameters $\boldsymbol{\psi}^\star$. The implementation of the algorithm considered in this work is shown in Algorithm 3. The algorithm presented in this work employs an adaptive learning rate η_j at each step of episodic REINFORCE. The learning rates can be adapted in many ways, however, in this work we utilize the RMSprop optimizer [48].

Algorithm 3 Batch Episodic REINFORCE

```
Input: A parameterized policy $\pi(\mathbf{a}_i|\mathbf{s}_i; \boldsymbol{\psi})$.
Initialization: Initial learning rate $\eta_1 > 0$ and initial policy parameters $\boldsymbol{\psi}_0$.
for $j = 1$ to $J$ do
    1. Collect $N$ experiences $\{\mathcal{E}_j^{(n)}\}_{n=1}^N$ using the policy $\pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\psi}_{j-1})$.
    2. Compute the average baseline $b$ according to (53).
    3. Compute $g_{\text{PGT}}(\boldsymbol{\psi}_{j-1})$ according to (52).
    4. Obtain $\boldsymbol{\psi}_j$ using a stochastic gradient step
           $\boldsymbol{\psi}_j = \boldsymbol{\psi}_{j-1} + \eta_j g_{\text{PGT}}(\boldsymbol{\psi}_{j-1})$.
    5. Determine the next learning rate $\eta_{j+1}$.
end for
Return: Policy parameters $\boldsymbol{\psi}_J \approx \boldsymbol{\psi}^\star$.
```

Appendix B: Proof of Proposition 1

Proof. By applying the law of total variance [49], where we condition on $\mathbf{u}_{1:I}$ and $\boldsymbol{\theta}_{0:I}$, we can rewrite the variance of $\hat{Z}_{\text{AIS}}^{M\times I}$ as follows

$$\begin{aligned}\mathbb{V}\left[\hat{Z}_{\text{AIS}}^{M\times I}\right] =& \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\mathbb{V}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\mathbf{u}_{1:I}, \boldsymbol{\theta}_{0:I}\right]\right] \\ &+ \mathbb{V}_{\beta_{\boldsymbol{\psi}}}\left[\mathbb{E}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\mathbf{u}_{1:I}, \boldsymbol{\theta}_{0:I}\right]\right].\end{aligned} \quad (54)$$

Recall that that if the proposal parameters are given, then the estimator $\hat{Z}_{\text{AIS}}^{M\times I}$ is unbiased. Therefore,

$$\begin{aligned}\mathbb{E}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\mathbf{u}_{1:I}, \boldsymbol{\theta}_{0:I}\right] &= \mathbb{E}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\boldsymbol{\theta}_{1:I}\right] \\ &= Z.\end{aligned} \quad (55)$$

This implies that the second term in (54) is

$$\mathbb{V}_{\beta_{\boldsymbol{\psi}}}\left[\mathbb{E}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\mathbf{u}_{1:I},\boldsymbol{\theta}_{0:I}\right]\right]=0, \tag{56}$$

since the variance of a constant is 0. Furthermore, since the importance weights are conditionally independent of one another when the proposal parameters are given, we can rewrite the first term in (54) as

$$\begin{aligned}\mathbb{E}_{\beta_{\boldsymbol{\psi}}}&\left[\mathbb{V}\left[\hat{Z}_{\text{AIS}}^{M\times I}\middle|\mathbf{u}_{1:I},\boldsymbol{\theta}_{0:I}\right]\right]\\ &\propto \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\mathbb{V}\left[\sum_{i=1}^{I}\sum_{m=1}^{M}\tilde{w}(\mathbf{x}_i^{(m)},\boldsymbol{\theta}_i)\middle|\boldsymbol{\theta}_{1:I}\right]\right]\\ &=\mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\sum_{m=1}^{M}\mathbb{V}\left[\tilde{w}(\mathbf{x}_i^{(m)},\boldsymbol{\theta}_i)|\boldsymbol{\theta}_i\right]\right]\\ &\propto \mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[-\sum_{i=1}^{I}\tilde{R}_V(\boldsymbol{\theta}_i)\right],\end{aligned}$$

where $\tilde{R}_V(\boldsymbol{\theta}_i)=-\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]$. Then, one can find the $\boldsymbol{\psi}^\star$ that minimizes $\mathbb{V}\left[\hat{Z}_{\text{AIS}}^{M\times I}\right]$ by solving the following equivalent optimization problem:

$$\boldsymbol{\psi}^\star=\underset{\boldsymbol{\psi}\in\boldsymbol{\Psi}}{\arg\min}\,\mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[-\sum_{i=1}^{I}\tilde{R}_V(\boldsymbol{\theta}_i)\right]. \tag{57}$$

By multiplying the objective function in (57) by -1, we arrive at the result given by (21). $\square$

Appendix C: Proof of Proposition 2

Proof. We rewrite (22) as

$$\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i})=\frac{1}{Z^2}\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^2\right]-1.$$

Multiplying $\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i})$ by Z^2 yields

$$\begin{aligned}L_{\chi^2}(\boldsymbol{\theta}_i)&\triangleq Z^2\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i})\\ &=Z^2\left(\frac{1}{Z^2}\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^2\right]-1\right)\\ &=\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^2\right]-Z^2\\ &=\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^2\right]-\left(\int_{\mathcal{X}}\tilde{\varphi}(\mathbf{x})d\mathbf{x}\right)^2\\ &=\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^2\right]-\left(\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\tilde{\varphi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)\right]\right)^2,\end{aligned}$$

which is exactly $\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]$. Since minimizing $L_{\chi^2}(\boldsymbol{\theta}_i)=\mathbb{V}_{q_{\boldsymbol{\theta}_i}}[\tilde{w}(\mathbf{x},\boldsymbol{\theta}_i)]$ is equivalent to minimizing $\mathcal{D}_{\chi^2}(\varphi\|q_{\boldsymbol{\theta}_i})$, the result claimed in (23) follows. $\square$

Appendix D: Proof of Proposition 3

Proof. We begin by rewriting $\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i})$ as

$$\mathcal{D}_\alpha\left(\varphi\|q_{\boldsymbol{\theta}_i}\right)=\frac{1}{\alpha-1}\log\left(\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\left(\frac{\varphi(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^\alpha\right]\right). \tag{58}$$

Since $\log(z)$ is a concave function in $z\in\mathbb{R}^+$, we can apply Jensen's inequality [50] to obtain the following:

$$\begin{aligned}\mathcal{D}_\alpha\left(\varphi\|q_{\boldsymbol{\theta}_i}\right)&\geq\frac{1}{\alpha-1}\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\log\left(\left(\frac{\varphi(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)^\alpha\right)\right]\\ &=\frac{\alpha}{\alpha-1}\mathbb{E}_{q_{\boldsymbol{\theta}_i}}\left[\log\left(\frac{\varphi(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta}_i)}\right)\right]\triangleq\mathcal{L}_\alpha(\boldsymbol{\theta}_i),\end{aligned}$$

with equality if an only if $q(\mathbf{x};\boldsymbol{\theta}_i)=\varphi(\mathbf{x})$. Since $\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i})\geq\mathcal{L}_\alpha(\boldsymbol{\theta}_i)$ for all i, this implies that

$$\sum_{i=1}^{I}\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i})\geq\sum_{i=1}^{I}\mathcal{L}_\alpha(\boldsymbol{\theta}_i)$$

with equality if and only if $q(\mathbf{x};\boldsymbol{\theta}_1)=\cdots=q(\mathbf{x};\boldsymbol{\theta}_i)=\varphi(\mathbf{x})$. Taking expectations on both sides of the inequality w.r.t. $\beta_{\boldsymbol{\psi}}$, we have

$$\mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\mathcal{D}_\alpha(\varphi\|q_{\boldsymbol{\theta}_i})\right]\geq\mathbb{E}_{\beta_{\boldsymbol{\psi}}}\left[\sum_{i=1}^{I}\mathcal{L}_\alpha(\boldsymbol{\theta}_i)\right]. \tag{59}$$

Noticing that $R_E(\boldsymbol{\theta}_i)=\frac{\alpha-1}{\alpha}\left(\mathcal{L}_\alpha(\boldsymbol{\theta}_i)+\log(Z)\right)$, we conclude that maximizing the right-hand side of the inequality in (59) is equivalent to maximizing (27), which is a lower bound for the expected sum of Rényi divergences of any order $\alpha>1$. $\square$

References

[1] T. Kloek and H. K. Van Dijk, "Bayesian estimates of equation system parameters: an application of integration by Monte Carlo," *Econometrica: Journal of the Econometric Society*, pp. 1–19, 1978.

[2] C. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer Science & Business Media, 2013.

[3] J. V. Candy, *Bayesian Signal Processing: Classical, Modern, and Particle Filtering Methods*, vol. 54, John Wiley & Sons, 2016.

[4] B. Li, T. Bengtsson, and P. Bickel, "Curse-of-dimensionality revisited: Collapse of importance sampling in very large scale systems," .

[5] Y. Chen, "Another look at rejection sampling through importance sampling," *Statistics & probability letters*, vol. 72, no. 4, pp. 277–283, 2005.

[6] S. Brooks, A. Gelman, G. Jones, and X. Meng, *Handbook of Markov chain Monte Carlo*, CRC press, 2011.

[7] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," *Handbook of Nonlinear Filtering*, vol. 12, no. 656-704, pp. 3, 2009.

[8] N. Kurtz and J. Song, "Cross-entropy-based adaptive importance sampling using Gaussian mixture," *Structural Safety*, vol. 42, pp. 35–44, 2013.

[9] M. Oh and J. O. Berger, "Adaptive importance sampling in Monte Carlo integration," *Journal of Statistical Computation and Simulation*, vol. 41, no. 3-4, pp. 143–168, 1992.

[10] M. F. Bugallo, L. Martino, and J. Corander, "Adaptive importance sampling in signal processing," *Digital Signal Processing*, vol. 47, pp. 36–49, 2015.

[11] O. Cappé, R. Douc, A. Guillin, J. Marin, and C. P. Robert, "Adaptive importance sampling in general mixture classes," *Statistics and Computing*, vol. 18, no. 4, pp. 447–459, 2008.

[12] E. K. Ryu and S. P. Boyd, "Adaptive importance sampling via stochastic convex programming," *arXiv preprint arXiv:1412.4845*, 2014.

[13] Y. El-Laham, P. M. Djurić, and M. F. Bugallo, "A variational adaptive population importance sampler," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5052–5056.

[14] J. Cornuet, J. Marin, A. Mira, C. P. Robert, et al., "Adaptive multiple importance sampling," *Scandinavian Journal of Statistics*, vol. 39, no. 4, pp. 798–812, 2012.

[15] F. Portier and B. Delyon, "Asymptotic optimality of adaptive importance sampling," in *Advances in Neural Information Processing Systems*, 2018, pp. 3134–3144.

[16] T. A. Bojesen, "Policy-guided Monte Carlo: Reinforcement-learning Markov chain dynamics," *Physical Review E*, vol. 98, no. 6, pp. 063303, 2018.

[17] J. Heng, A. N. Bishop, G. Deligiannidis, and A. Doucet, "Controlled sequential Monte Carlo," *arXiv preprint arXiv:1708.08396*, 2017.

[18] E. Bernton, J. Heng, A. Doucet, and P. E. Jacob, "Schrödinger bridge samplers," *arXiv preprint arXiv:1912.13170*, 2019.

[19] C. A. Naesseth, F. Lindsten, and T. B. Schön, "Elements of sequential monte carlo," *arXiv preprint arXiv:1903.04797*, 2019.

[20] A. Gelman and X. Meng, "Simulating normalizing constants: From importance sampling to bridge sampling to path sampling," *Statistical Science*, pp. 163–185, 1998.

[21] C. Snyder, "Particle filters, the "optimal" proposal and high-dimensional systems," in *Proceedings of the ECMWF Seminar on Data Assimilation for Atmosphere and Ocean*, 2011, pp. 1–10.

[22] B. Delyon and F. Portier, "Safe and adaptive importance sampling: a mixture approach," *arXiv preprint arXiv:1903.08507*, 2019.

[23] F. Liang, C. Liu, and R. Carroll, *Advanced Markov chain Monte Carlo Methods: Learning from Past Samples*, vol. 714, John Wiley & Sons, 2011.

[24] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[25] J. Kiefer, J. Wolfowitz, et al., "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

[26] S. Andradóttir, "A projected stochastic approximation algorithm," Tech. Rep., Institute of Electrical and Electronics Engineers (IEEE), 1991.

[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, 2018.

[28] F. Nielsen and R. Nock, "On the Chi square and higher-order Chi distances for approximating f-divergences," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 10–13, 2013.

[29] T. Van Erven and P. Harremos, "Rényi divergence and Kullback-Leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[30] E. K. Ryu, *Convex optimization for Monte Carlo: Stochastic optimization for importance sampling*, Ph.D. thesis, Stanford University, 2016.

[31] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[32] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intelligent Transport Systems*, vol. 11, no. 7, pp. 417–423, 2017.

[33] T. Xifara, C. Sherlock, S. Livingstone, S. Byrne, and M. Girolami, "Langevin diffusions and the Metropolis-adjusted Langevin algorithm," *Statistics & Probability Letters*, vol. 91, pp. 14–19, 2014.

[34] V. Elvira and E. Chouzenoux, "Langevin-based strategy for efficient proposal adaptation in population Monte Carlo," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5077–5081.

[35] A. Vehtari, D. Simpson, A. Gelman, Y. Yao, and J. Gabry, "Pareto smoothed importance sampling," *arXiv preprint arXiv:1507.02646*, 2015.

[36] Y. Yao, A. Vehtari, D. Simpson, and A. Gelman, "Yes, but did it work?: Evaluating variational inference," *arXiv preprint arXiv:1802.02538*, 2018.

[37] A. Owen and Y. Zhou, "Safe and effective importance sampling," *Journal of the American Statistical Association*, vol. 95, no. 449, pp. 135–143, 2000.

[38] L. Martino, V. Elvira, D. Luengo, and J. Corander, "An adaptive population importance sampler: Learning from uncertainty," *IEEE Transactions on Signal Processing*, vol. 63, no. 16, pp. 4422–4437, 2015.

[39] L. Martino, V. Elvira, D. Luengo, and J. Corander, "Layered adaptive importance sampling," *Statistics and Computing*, vol. 27, no. 3, pp. 599–623, 2017.

[40] O. Cappé, A. Guillin, J. Marin, and C. P. Robert, "Population Monte Carlo," *Journal of Computational and Graphical Statistics*, vol. 13, no. 4, pp. 907–929, 2004.

[41] Y. El-Laham and M. F. Bugallo, "Stochastic gradient population Monte Carlo," *IEEE Signal Processing Letters*, 2019.

[42] V. Elvira, L. Martino, D. Luengo, and M. F. Bugallo, "Improving population Monte Carlo: Alternative weighting and resampling schemes," *Signal Processing*, vol. 131, pp. 77–91, 2017.

[43] H. Haario, E. Saksman, and J. Tamminen, "Adaptive proposal distribution for random walk Metropolis algorithm," *Computational Statistics*, vol. 3, no. 14, pp. 375–395, 1999.

[44] A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine, "Meta-reinforcement learning of structured exploration strategies," *arXiv preprint arXiv:1802.07245*, 2018.

[45] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 2219–2225.

[46] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, 2000, pp. 1057–1063.

[47] L. Weaver and N. Tao, "The optimal reward baseline for gradient-based reinforcement learning," *arXiv preprint arXiv:1301.2315*, 2013.

[48] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[49] N. A. Weiss, *A Course in Probability*, Addison-Wesley, 2006.

[50] W. Rudin, *Real and Complex Analysis*, McGraw-Hill, New York, NY, USA, 3 edition, 1986.