IS RANDOM EXPERT DISTILLATION TRUSTWORTHY? A MORE EFFICIENT EXPERT POLICY SUPPORT ESTIMATION METHOD VIA EXPLICIT VARIANCE

Lei Shi
Department of Biostatistics and Epidemiology
University of California, Berkeley
leishi@berkeley.edu

Yunzhe Zhou
Department of Biostatistics and Epidemiology
University of California, Berkeley
ztzyz615@berkeley.edu

ABSTRACT

Imitation Learning is concerned with the problem when the dataset consists of a finite number of expert trajectories and no reward function is accessible. Many classical approaches including inverse reinforcement learning and generative adversarial methods are targeted at constructing suitable reward functions, but they suffer from the instability and high computational cost. Recently, a new method called Random Expert Distillation (RED) is proposed to compute a reward function by expert policy support estimation. In this article, we present that the potential problems of RED lie in the dominant support estimation bias and extra sources of variability. To this end, we propose a new explicit variance based approach. The efficacy of our proposed method is demonstrated via extensive simulation. As a byproduct, we provide theoretical insights for its connection with support density estimation. The code is available at `https://github.com/LeiShi-rocks/CS285-FinalProject`.

Keywords Imitation Learning · Variance Estimation · Kernel Density · Expert Policy · Support Estimation

1 Introduction

Imitation learning is the task of learning optimal policy from the expert trajectories [Hussein et al., 2017]. It aims to mimic human behavior in a given task. An agent is normally trained to perform a task from demonstrations by learning a mapping between observations and actions. Conventionally, the imitator has access to both state and action information generated by an expert performing the task and the export demonstrations is generally a offline dataset, that means only finite number of trajectories is available without any further access to any other online data resourses. There are many classic approaches to solve imiation learning problem.

Behavior cloninig (BC) is a supervised learning method to learn a policy mapping from the state space to the action space with the offline expert datasets [Pomerleau, 1991]. However, BC typically requires an extremely large sample size to guarantee the shifting error of the learned policy caused by the complex data structure. For instance, in autonomous driving, behavior cloning has the well known limitations (e.g., dataset bias and overfitting), new generalization issues (e.g., dynamic objects and the lack of a causal modeling), and training instabilities [Codevilla et al., 2019]. Inverse reinforcement learning (IRL) focuses on the problem of extracting a reward function given observed, optimal policy [Ng et al., 2000]. It has the potential to use data recorded in performing a task to build autonomous agents capable of modeling others without intervening in the performance of the task [Arora and Doshi, 2021] and avoid the problem of compounding errors. However, the computation cost might become potential problems by IRL to extract reward functions and then send it to reinforcement learning algorithm, which requires reinforcement learning in an inner loop. The main reason for such expensive computation is that IRL learns a cost function, which explains expert behavior but does not directly tell the learner how to act. Generative Adversarial Imitation Learning (GAIL) is an approach to connect IRL to the Generative Adversarial Networks [Ho and Ermon, 2016]. However, it suffers from the problem of instability caused by the use of GAN.

Recently, a new framework for imitation learning called Random Expert Distillation (RED) has been proposed by estimating the support of the expert policy to compute a fixed reward function [Wang et al., 2019]. The effacy of the proposed method was demonstrated on both discrete and continous domians. Although with very extensive theoretical motivations of support estimation, implementation of the algorithm simply uses a random network distillation technique to estimate the support of domain. In this article, we will demonstrate the potential bias of such method, which could lead to the instability of the performance for the learned reward function, as shown by the wide standard deviation region of the simulation results in original paper.

Rethinking about the philosophy of random network distillation for RED algorithm, it basically tries to extract the density information of the state and action space of the expert trajectories by calculating the empirical MSE of the neural network. Intuitively, if the state and action tuple appears sparsely in some area, the empirical MSE will be relatively large. On the other hand, if the state and action tuple falls into a dense region, this can lead to a relatively small empirical MSE. As we all know the famous tradeoff when we firstly got into the field of machine learning, the MSE can be factorized into bias and variance parts [Neal et al., 2018]. Actually, the variance term is more concerned with the distribution of the state and action space. This inspires us to estimate the variance or the density function directly.

Random network distillation is actually closely related to using Monte Carlo method for estimation of empirical MSE. However, the explicit variance estimation formula can be constructed for many traditional models. As our target is to infer the density support of expert data, it shouldn't be only limited to using neural network as the surrogate function. Variance estimation is an important topic in traditional statistics, which is an inevitable step in the statistical inference. It targets to quantify the uncertainly of the estimated models and provide insights for the model interpretations. As a mysterious black box, there is lack of work to quantify the uncertainty of the neural network. The only method might be through bootstrap or Monte Carlo simulations. But this could lead to potential bias. In contrast, as for many traditional statistical models, complete theories have been established to guarantee the correctness of the estimated variance. For example, the most simple case is the linear regression. The confidence interval of the expectation mean can be easily constructed [Shrestha and Solomatine, 2006]. Random forest has been known to be a black model with lack of understanding. However, in recent years, many methods have been successfully applied to estimate the variance of the predictions from random forest. The infinitesimal jackknife is one of the most efficient method for variance estimation in random forest [Wager et al., 2014]. It can achieve reasonable coverage rate for the confidence interval of the predictor.

Another direction one might consider is to extract support information from an explicit density estimation. Kernel density estimation serves as a class of well-cultivated schemes, which possesses statistical accuracy as well as simplicity in terms of computation. As a generalization of the traditional histogram estimator, kernel estimation is able to utilize information from the whole dataset and introduce smoothness into the results[Friedman et al., 2001]. Kernel estimation provides large flexibility in density exploration due to a rich collection of choices in terms of kernel functions(e.g. gaussian kernel, cosine kernel, Epanechnikov kernel and so on, see Epanechnikov [1969]) and the freedom of tuning the window size. Theoretically, many kernels lead to a minimax optimality for density estimation as long as certain smoothness condition is satisfied and the window is specified at a proper rate[Wahba, 1975]. In practice, many appealing procedures have been proposed to provide guidelines for parameter tuning, such as cross validation[Hall et al., 1992] and plug-in selection[Sheather and Jones, 1991]. Overall, it's of great interest to investigate how much we can gain by combining this class of nonparametric estimation schemes with reward learning in various tasks.

In this article, we propose a new framework for expert policy support estimation with explicit variance calculation. Empirically, we will illustrate its advantage by comparing it with RED through extensive simulations. Theoretically, we provide insights for its connection with support density estimation. In addition, we also explore the performance of the kernel density estimation based methods for support estimation and evaluate it in several experiment settings.

The rest of the article is organized as follows. We introduce the background and potential limitations of RED in Section 2. This becomes the motivation of our proposed method. In Section 3, we present our proposed explicit variance based support estimation framework. Kernel density techniques are also discussed as a byproduct. We provide some insights for establishing the connection between our proposed method with support density estimation in Section 4. Extensive experiments studies are conducted in Section 5. Conclusions and future work are discussed in Section 6.

2 Motivation

In this section, we firstly have a brief review of the RED method and illustrates how it works. Next, we point out its potential limitations caused by the dominant bias error and the other sources of model variability. This finally leads to the motivation of our proposed methodology.

2.1 Random Expert Distillation

RED is a framework proposed to learn reward function via expert policy support estimation [Wang et al., 2019] . The main idea is that the expert trajectories encode the distribution information of the state and action space. The successful mimic of the expert distribution space can potentially lead to promising performance of reinforcement learning algorithm. So RED is inspired to extract reward function through support density estimation. Suppose that we have n expert trajectories $\{a_1^{(i)}, s_1^{(i)}, a_2^{(i)}, s_2^{(i)}, \cdots, a_T^{(i)}, s_T^{(i)}\}_1^n$ with same length T. If the state and action space is discrete, the reward function can be constructed as a indicator function for the support of expert policy π_E. That is

$$r_E(s,a) = \begin{cases} 1 & \text{if}(s,a) \in \text{supp}(\pi_E) \\ 0 & \text{if}(s,a) \notin \text{supp}(\pi_E) \end{cases} \tag{1}$$

For the continuous state and action spaces, the score of the support estimation can be directly used for constructing the reward function, with the rationale that if an action is similar to that of the expert in any given state, it would receive high scores from the support estimation. So if we assume that support estimation of expert policy is strongly related to the reward function of imitation learning, the reward function can push reinforcement learning algorithm to mimic the behavior of the expert at the state and action spaces. However, explicitly estimating the support can be tricky and if the support of expert policy coincide with the whole space under the continuous case, the support estimation will be possibly uninformative. Thus, RED used the Random Network Distillation (RND) technique to extract the reward function.

Consider a neural network f_θ parametrized by $\theta \in \Theta$, where Θ is the parameter space of the neural network. We can sample $\theta_1, \theta_2, \cdots, \theta_K$ accroding to a specific distribution over Θ. Then given training dataset $\{x_i\}_{i=1}^N$ sampled from the expert policy π_E (we concatenate the original n trajectories with length T into $N = n \times T$ samples with $x = (s,a)$). Then we need to solve K regression problems

$$\hat{\theta}_k = \min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} (f_\theta(x_i) - f_{\theta_k}(x_i))^2 \tag{2}$$

for every $k = 1, 2, \cdots, K$. Then for any new point $x = (s,a)$, the prediction error can be extracted by using the following formula

$$L(x) = \frac{1}{K} \sum_{k=1}^{K} (f_{\hat{\theta}_k}(x) - f_{\theta_k}(x))^2 \tag{3}$$

Then the reward function can be constructed in the below which has very good empirical performance

$$r(x) = \exp(-\sigma_1 L(x)) \tag{4}$$

where the hyperparameter σ_1 is tuned such that $r(s)$ form the expert demonstrations are mostly close to 1.

2.2 Limitations of RED

If we rethink about the RED method carefully, we can find that it basically extracts the information of the state and action space by calculating the prediction error and then transform it into a more flat form by using the exponential function. However, the question could be raised that whether using such prediction error of neural network is actually reasonable for expert support estimation? Actually, the prediction error in this case is equivalent to the empirical MSE, which can be factorized into bias and variance term:

$$L(x) = L_{\text{bias}}(x) + L_{\text{var}}(x) \tag{5}$$

Actually, only the variance loss captures the distribution of the expert's state and action space while bias loss is more concerned with the properties of the constructed estimator. In random neural network distillation, both the bias and variance terms are considered, which gives birth to a potential problem: if the bias term is dominant, the prediction error cannot reflect the distribution of expert trajectories anymore. We demonstrate this phenomenon by the following theorem:

Theorem 2.1. *Suppose that the state and action tuples of expert policy has the density $q(s, a)$ and assume that the statistical model guarantees that $[q(s_1, a_1) - q(s_2, a_2)][L_{var}(s_1, a_1) - L_{var}(s_2, a_2)] \geq 0$ for any two tuples (s_1, a_1) and (s_2, a_2). If $L_{var}/L_{bias} = o(1/N^{\alpha})$ for any $\alpha > 0$, as N is sufficiently large and there is no linear trend relationship between $q(s, a)$ and L_{bias}, we can find two tuples (s'_1, a'_1) and (s'_2, a'_2) such that $q(s'_1, a'_1) < q(s'_2, a'_2)$ but $L(s'_1, a'_1) > L(s'_2, a'_2)$.*

Theorem 2.1 illustrates that when the bias term is dominant as the sample size N goes towards infinity, the prediction error based estimation can be misleading since a higher error value doesn't mean the state and action tuple happens rarely in the expert domain. This bias term is normally referred to as the approximation error [Mohri et al., 2018].

There is also another potential issue of RED: the variability of neural network doesn't only come from the data variability but also come from procedural variability [Huang et al., 2021]. In contrast to many traditional statistical models, the neural network has a more complex structure and training process, including early stopping, stochastic gradient descent, dropout, and random initialization. As a result, they could lead to the procedural variability of neural network. The procedural variability comes from the two main resources. The first one is caused by the sub-optimal solution of the empirical risk function for neural network. The loss function of neural network is highly non-convex and also the stochastic gradient descent is used for solving the optimization problem with early stopping. So the global optimal solution cannot be attained. This leads to the uncertainty of the optimization. Another source of uncertainty comes from the random initialization. Since we randomly initialize the network parameters, we introduce the additional epistemic uncertainty. We write Γ as the random variable corresponding to procedural variability and use $\hat{\pi}_{D_{\text{tr}}}$ to denote the data distribution, the trained parameter $\hat{\theta}$ is an instantiation of an outcome form one training run based on Γ and $\hat{\pi}_{D_{\text{tr}}}$. The total epistemic uncertainty of neural network can be written as

$$h_{\hat{\theta}} - f^* =: \text{UQ}_{\text{EU}} = \text{UQ}_{\text{MU}} + \text{UQ}_{\text{PV}} + \text{UQ}_{\text{DV}} \tag{6}$$

where $\text{UQ}_{\text{MU}} = 0$ in this case since the neural network is used as the true model. UQ_{PV} is the procedural variability, which consists of random initialization and sub-optimal empirical risk minimization. UQ_{DV} is the data variability, which is strongly correlated to the data distribution and can reflect the support density of the expert policy. However, the main problem of the random network distillation is that it calculate all the three terms in the equation 6, but only the third term is the one that is concerned with the support density. So if the second term is dominant, this could potentially mislead the random network distillation.

In order to overcome this limitations, we are inspired to come up a framework to merely estimate the term UQ_{DV} without introducing any potential estimation bias or procedure variability. The specific algorithm will be discussed in the next section, which can be directly targeted at the variance term that is strongly correlated with the density distribution of the expert policy.

3 Methodology

In this section, we demonstrate our explicit variance based estimation method via several specific examples. As a byproduct, we also explore other potential methods, such as the kernel density based approach for extracting reward function. The detailed algorithm is summarized.

3.1 Explicit Variance Evaluation

Let's consider the previous case. Suppose that we have n export trajectories $\{a_1^{(i)}, s_1^{(i)}, a_2^{(i)}, s_2^{(i)}, \cdots, a_T^{(i)}, s_T^{(i)}\}_1^n$ with the same length T. We can concatenate it into the dataset with $N = T \times n$ elements, which is $\{x_1, x_2, \cdots, x_N\}$, with x representing the tuple (s, a). Then consider a statistical model f_θ parametrized by $\theta \in \Theta$, where Θ is the parameter space of the statistical model. Then we consider the following data generating model,

$$r = f_\theta(a, s) + \epsilon \tag{7}$$

where $\epsilon \sim N(0, \sigma)$ and σ is a fixed value for the standard deviation. By following this generating process, we can generate the response reward values $\{r_1, r_2, \cdots, r_N\}$ based on the original state and action trajectories. We can estimate the parameter θ by optimizing the loss function, such as least square or log-likelihood. This yields out the estimator $\hat{\theta}$.

To quantify the support density of the expert trajectories, we are more interested in the prediction variability for a given test point of expert state and action tuple. For traditional statistical models, this variance can be explicitly estimated based on the empirical sample, which is denoted as $\hat{\Omega}(x)$ for any testing point x. We demonstrate more details for the

Algorithm 3.1 EXPLICIT VARIANCE BASED SUPPORT ESTIMATION

Input: data $D = \{s_i, a_i\}_{i=1}^N$, a randomly specified statistical model f_θ, initial policy π_0.
Estimate the parameter via minimizing the loss function $\hat{\theta} = \arg\min_\theta L(\theta, \hat{\theta}, D)$
Use explicit variance estimation technique in section 3.2 for the support estimation. This yields out $\hat{\Omega}$.
Extract the reward function by calculating $r(\cdot) = \exp(-\sigma_1 \hat{\Omega})$.
$\pi = \text{REINFORCEMENTLEARNING}(r, r_{\text{term}}, \pi_0)$
Return: π

variance estimation in the next subsection 3.2. Then similarly to RED, we construct the reward function by using an exponential function transformation, that is

$$r(s, a) = \exp(-\tau \hat{\Omega}(s, a)) \tag{8}$$

where τ is a hyperparameter for tuning. A higher value of τ might lead to a sharper boundary of the extracted reward function curve while a small value will lead to a more flat curve. The whole procedure is summarized in Algorithm 3.1.

3.2 Implementation Examples

In section 3.1, we came up a explicit variance based support estimation algorithm, with more simple procedure and more efficient computation efficacy. In this subsection, we will demonstrate several specific examples for the explicit variance evaluation based on different traditional statistical models. The efficiency of such method will be further illustrated with extensive experiments at section 5.

3.2.1 Gaussian Additive Model with Basis Functions

Basis function is a popular tool in statistical modeling. It has very good approximation power with relatively complex structure. On the other hand, it still maintain a good property for model interpretation. There are many commonly used basis function including Fourier basis, Radial basis, Neural Network basis and etc [Huang et al., 2021, Paciorek, 2007]. With rigorous theoretical guarantee, the explicit variance evaluation can be mathematically constructed and has promising simulation performance.

Suppose that we have sieve base functions $\Psi(s) = \{\Psi_i(s)\}_{1 \le i \le L}$, initial parameter $\theta_{(0)} = \{\theta_1^{(0)}, ..., \theta_m^{(0)}\}$,where $\theta_i^{(0)} = \{\theta_{i,1}^{(0)}, ..., \theta_{i,L}^{(0)}\}$ and the number of iterations K. Let $R_{\theta^{(0)}}(a, s) = \xi^T(a, s)\theta^{(0)}$ be the initial linear sieve estimate with parameter $\theta^{(0)}$. Then we can update the parameter by solving $\theta^{(1)} \in \text{argmin}_\theta \sum_{i=1}^N \Big(R_i - \xi^T(a_i, s_i)\theta\Big)^2$. Next, the explicit variance can be constructed by using the following formula

$$\hat{\Omega}(a, s) = t_{\alpha/2, N-(L+1)} \cdot \hat{\sigma}_1 \sqrt{\xi^T(a, s)\Big(\sum_{i=1}^N \xi(A_i, S_i)\xi^T(A_i, S_i)\Big)^{-1}\xi(a, s)} \tag{9}$$

This constructed variance has the theoretical guarantee for the correct coverage rate. This estimator can be prove to be unbiased and it has natural property of robustness which can lead to more stable performance of reinforcement learning algorithm.

3.2.2 Infinitesimal Jackknife

The Infinitesimal Jackknife (IJ) [Wager et al., 2014] is a framework for estimating the variance of any statistic. If we use P^0 to represent the uniform probability distribution over the empirical dataset and re-write the estimate more generally as $\hat{\theta}(P^*)$ where P^* is any re-weighting of the empirical distribution, then we can give the directional derivatives as

$$U_i = \lim_{\epsilon \to 0} \frac{\hat{\theta}(P^0 + \epsilon(\delta_i - P^0)) - \hat{\theta}(P^0)}{\epsilon}, \quad i = 1, \cdots, n \tag{10}$$

Under suitable conditions, the variance of our targeted estimator can be calculated by $\frac{1}{n^2}\sum_{i=1}^n U_i^2$. This is defined as the Infinitesimal Jackknife variance estimate for the estimator $\hat{\theta}(P^0)$. Infinitesimal Jackknife can be theoretically used for the variance estimation of any M-estimator. This good property can be extended for the explicit variance evaluation of complex statistical model, such as the random forest. The directional derivatives of random forest is constructed by

$$U_i(x) = n \cdot \text{cov}_b(N_{i,b}, T_b(x)) \tag{11}$$

where the covariance is calculated over $b = 1, 2, \cdots, B$ trees and $N_{i,b}$ is the number of times the i^{th} training data point is included in the smaple used to train the b^{th} tree and T_b is the b^{th} tree kernel. It is well known that such estimator will introduce an upward bias and the reason of such bias generally comes from the incomplete U-statistics because the number of trees used are very small compared to the total number of possible trees that could be used.

3.2.3 Neural Tangent Kernel

The estimation of the variance for neural network has been a tractable problem. The most naive and straightforward way is through the bootstrap. However, the estimator can suffer from strong bias. As the aritificial neural networks are equivalent to Gaussian processes in the infinite-width limit, this inspires researchers to connect them to kernel methods. Neural Tangent Kernel (NTK) [Jacot et al., 2018] has been brought up to approximate the behaviour of artificial neural network with strict theoretical guarantee. By considering the NTK, [Huang et al., 2021] successfully construct the explicit variance evaluation based on the kernel theories. The result is quite fascinating, which inspires us to use NTK as the baseline model to generate reward function and use related variance for extracting the reward function. The directional derivatives can be constructed based on the kernel theories, which can be used for explaining the data variability. This can be strongly correlated to the expert data distribution.

3.3 Kernel Density Estimation

For tasks that are located in moderate dimensions, we consider another strategy: estimating the density underlying the data points using nonparametric methods. In other words, we don't assume any parametric probabilistic form but try to extract distributional information directly from the data points. While neural networks typically serve as a powerful tool for reconstructing or mimicking a highly non-linear function, we hope to avoid fitting deep networks for ease and stability of numerical computation. The raw idea is simply counting the frequency of appearance for data points around a certain location:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{\|x_i - x\| \leq h\}.$$

This corresponds to a traditional statistical tool for data analysis: histograms. However, directly applying frequency estimation fails to generate a smooth density estimator. Besides, it cannot borrow information from points that are far from the currently studied location. A more natural choice is to apply a kernel trick, by which we introducing kernels to give weights for the data points. Closer points get higher scores and vice versa. Mathematically we have

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} K_H(x_i - x).$$

$K(\cdot)$ is typically chosen to be a density function, and K_H is given by

$$K_H = |H|^{-1/2} K(H^{-1/2}x).$$

Theoretically the choice of K does not matter much as along as it satisfies several smoothness conditions. One popular choice is the multivariate normal kernel:

$$K_H = (2\pi)^{-d/2}|H|^{-1/2}\exp\left(-\frac{1}{2}x^\top H^{-1}x\right).$$

H is typically called a window width and serves as a tuning parameter here. Starting from a simple pattern, H can be taken as hI_p for some $h > 0$. This structure specifies an identical window for all components and assumes spatial homogeneity across dimensions. More generally, one can choose H as a diagonal matrix or a positive definite matrix to facilitate heterogeneity.

To measure the accuracy of nonparametric kernel density estimation, one can consider the mean integrated squared error(MISE):

$$\text{MISE} = E\int \left(\hat{f}_H(x) - f(x)\right)^2 dx.$$

It can be shown that the optimal MISE(in a minimax sense) is achieved by any reasonable bandwidth selector H that has $H = O(n^{-2/(d+4)})$, where the big O notation is applied elementwise. Substituting this into the MISE formula yields that the optimal MISE is of the order $O(n^{-4/(d+4)})$. In practice there are many data-driven strategy for choosing a good window:

- Plug-in. An asymptotic approximation of the MISE is given by

$$\text{AMISE} = n^{-1}|H|^{-1/2}R(K) + 1/4m_2(K)^2(\text{vec}H)^\top \Psi_4(\text{vec}H).$$

  Here $R(K) = \int K(x)^2 dx$ and $\int xx^\top K(x)dx = m_2(K)I_d$. Ψ_4 is a $d^2 \times d^2$ matrix of integrated fourth order partial derivatives of f. After plugging in an estimator for this moment quantity we can minimize a sample version AMISE to get an estimated window.

- Cross validation. We can use smoothed cross validation (SCV) to obtain a working H. Here SCV is a subset of a larger class of cross validation techniques[Hall et al., 1992].

In practice the direct application of kernel density estimation can suffer from curse of dimensionality. Consider some data points in $\mathbb{R}^{10}$. Suppose we require at least 10 data points in each dimension, this might lead to a total demand of 10^{10} points, which is obviously forbidden in practice. Hence when learning rewards we first perform dimension reduction on the original data. For tasks in moderate dimensions, we use principal component analysis(PCA) to transform data into low dimensional principal components representations, which are then utilized to learn the underlying density.

4 Theory

In this section, we will establish the connection between explicit variance evaluation and the expert trajectories distribution. Ideally, the evaluated variance should reflect the distribution of action and state space. If a action and state tuple appears rarely, this should be corresponded to relatively large variance. We establish the following theory to provide the theoretical insights for our explicit variance evaluation method. This reveals the geometric connection between the expert density space and the explicit variance value space.

Theorem 4.1. *(Bidirectional Asymptotics) Suppose that we have n export trajectories $\{a_1^{(i)}, s_1^{(i)}, a_2^{(i)}, s_2^{(i)}, \cdots, a_T^{(i)}, s_T^{(i)}\}_1^n$ with same length T. Assume that each trajectory is independent and form a strictly stationary seires with β-mixing coefficients satisfy that $\beta(p) \leq c_1 \exp(-c_2 p)$ for some constants $c_1, c_2 > 0$. Let's denote the expert policy's true density as $q_E(s, a)$. By using the explicit variance evaluation technique in section 3.1 to extract the reward function $\hat{r}(s, a)$, we can prove that as either $n \to \infty$ or $T \to \infty$, with probability goes towards 1, we have $[\hat{r}(s_1, a_1) - \hat{r}(s_2, a_2)][q_E(s_1, a_1) - q_E(s_2, a_2)] \geq 0$ for any (s_1, a_1) and (s_2, a_2) tuples.*

Since the original dataset is not ideally i.i.d but different trajectories of time sequences instead, we need to establish the assumption for the stationary condition. This assumption enables us to establish the limiting distribution of our test when $p \to \infty$. We require $\beta(p)$ to decay exponentially with respect to p. When $\{Z_t\}_t$ forms a Markov chain, it is equivalent to the geometric ergodicity condition [Bradley, 2005]. This condition is commonly imposed in the time series literature. We show the detailed proof in the appendix.

5 Experiments

In this section, we present the extensive numerical results for our proposed explicit variance evaluation algorithm to demonstrate its advantage over RED method. We firstly brief introduce the experiment environments, model architecture settings and hyperparameters selections. Next, we compare the difference of the constructed reward function between explicit variance evaluation and RED with several graph visualization. Then we present the evaluation performance of these two methods on reinforcement learning algorithm. In addition, we also explore the kernel density estimation based method with implemenations in several experiments.

5.1 Experiment Setting

We evaluate the performance of explicit variance evaluation based support estimation on the several control tasks from the Mujoco environment including Hopper, HalfCheetah, Walker2d and etc. We compare our proposed method against RED to demonstrate the advantages. Similiar to the experiments in [Wang et al., 2019], we consider learning with 4 trajectories of expert demonstration generated by an expert policy trained with RL. To make sure, All RL algorithms terminate within 5M environmen steps. We repeat several seeds for each experiment to rule the effect of randomness. All the simulation was run on NVIDIA Tesla K80 GPU and the average computation time for each experiment replication is around 15 minutes, which is only half time of the RED algorithm. The results suggest that the explicit varaince evaluation method has more stable performance compared with RED method and can also achieve higher average reward.

5.2 Comparison of Extracted Reward

To begin with, we firstly present some comparisions of the extracted reward function under the Hopper-v2 environment as an example. This can demonstrate the difference between Explicit Variance Based Evaluation Method with RED. Figure 1 shows the scatter plot and heatmap for the density distribution of the expert action space in first two dimensions under the Hopper-v2 environment with different smoothing parameters. We can see that the action space of expert policy is not evenly distributed and concentrates on two centers. Therefore, the reinforcement algorithm should ideally mimic the distribution of the expert action space.

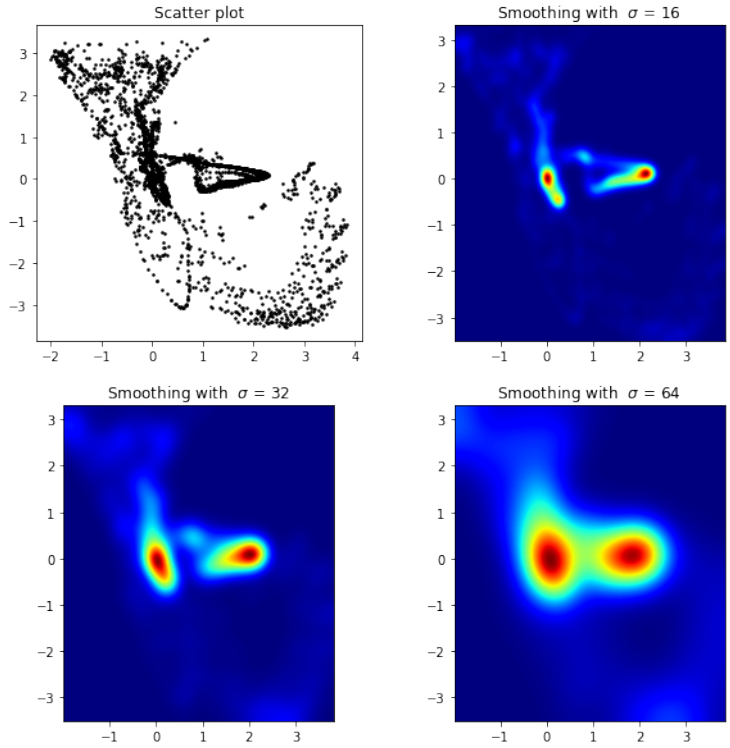

Figure 1: Scatter plot and heatmap for the density distribution of the expert action space in first two dimensions under the Hopper-v2 environment with different smoothing parameters.

We also present the heatmap of the extracted reward value of the expert action space in first two dimensions by using RED and Explicit Variance Based Evaluation (EVB) under the Hopper-v2 environment in Figure 2. From the graphs, we can see that compared with our proposed EVB method, RED has a more clear and sharp boundary. As for EVB, the boundary disappear gradually. This difference will be discussed in the next subsection that leads to the stationarity of our proposed method compared with RED.

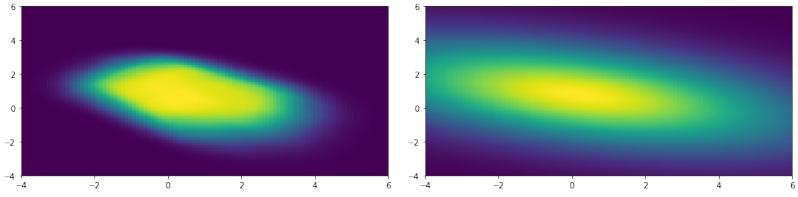

Figure 2: The heatmap of the extracted reward value of the expert action space in first two dimensions by using RED (on the left side) and Explicit Varaince Based Evaluation (EVB, on the right side) under the Hopper-v2 environment.

The distribution of the expert state space is also explore under these two methods, as shown in Figure 3. In the same way, we can see that even though the RED and EVB has the similar center for the extracted reward value, the EVB

has a wider ranger and more moderate boundary. It might be correlated to the natural property of the explicit variance estimation method. To further explore the difference of the extracted reward function, we also generate the function curve of the extracted reward value with these two different methods. The plots are presented in Figure 4. We can observe that our proposed method seems to yield smoother reward curve while the original RED reward curve possesses less smoothness but more strict boundaries. These plots imply some inferiority of direct linear-model based variance estimation, since it leads to over-smooth thus less decisive reward curve.

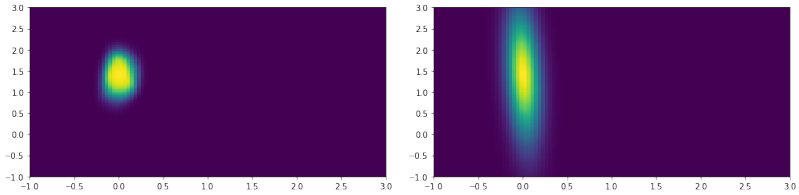

Figure 3: The heatmap of the extracted reward value of the expert state space in first two dimensions by using RED (on the left side) and Explicit Varaince Based Evaluation (EVB, on the right side) under the Hopper-v2 environment.

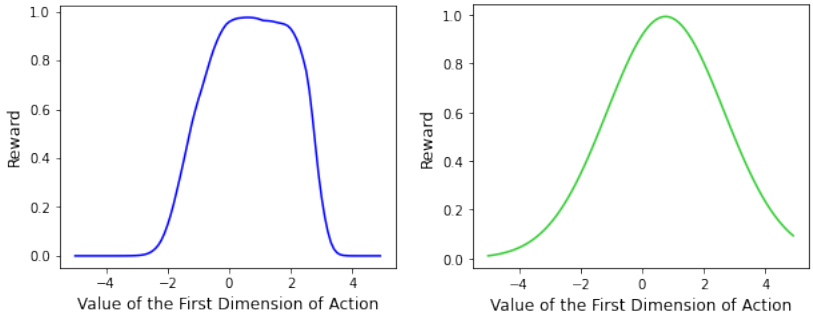

Figure 4: The curve of the extracted reward value of the expert action space in first two dimensions by using RED (on the left side) and Explicit Varaince Based Evaluation (EVB, on the right side) under the Hopper-v2 environment.

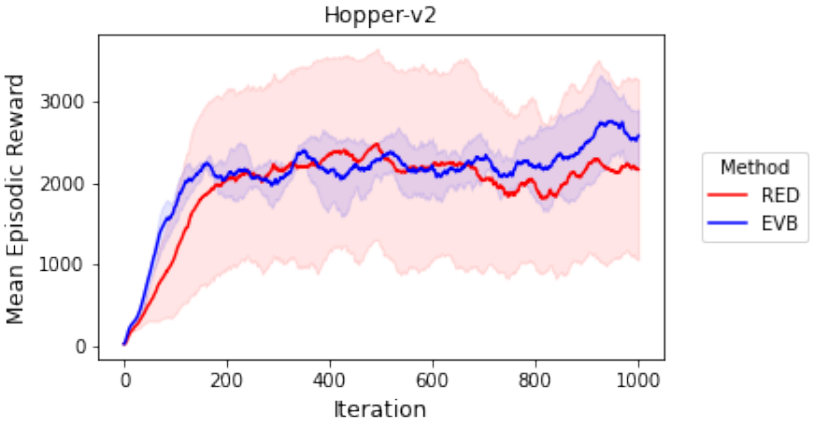

Figure 5: The true mean episodic reward curve of RED and EVB (explicit variance based evaluation) method over the iteration steps under the Hopper-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve.

5.3 Explicit Variance Evaluation

To evaluate the performance of our proposed explicit variance based estimation method, we implement it on several reinforcement learning environments and compare it with RED method. We start to take a look at the Hopper-v2

environment. Hopper-v2 is the task to make a two-dimensional one-legged robot hop forward as fast as possible. The result is shown in Figure 5. From the plots, we can the explicit variance based evaluation method can achieve a little better results compared with the RED method. In addition, the more important observation is that the standard deviation of the explicit variance based evaluation method is much smaller than the RED curve. We can clearly find out that the curve of RED method is very unstable over different seeds. With explicit variance evaluation technique, we can successfully make the reinforcement learning algorithm more robust and less insensitive to the random variations.

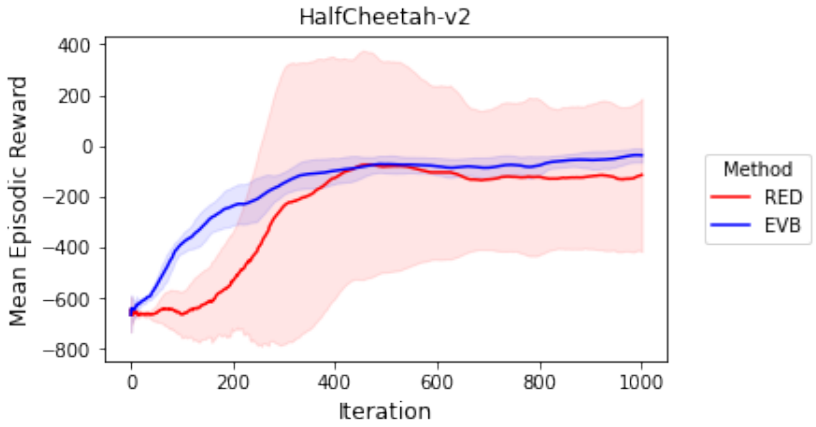

Figure 6: The true mean episodic reward curve of RED and EVB (explicit variance based evaluation) method over the iteration steps under the HalfCheetah-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve.

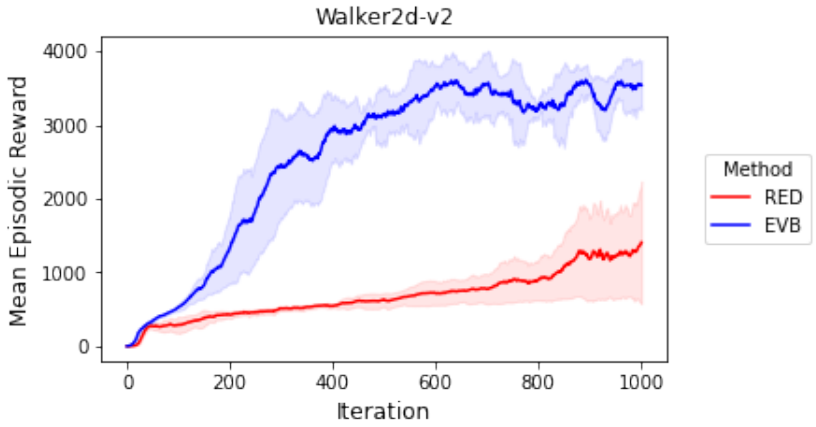

Figure 7: The true mean episodic reward curve of RED and EVB (explicit variance based evaluation) method over the iteration steps under the Walker2d-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve.

We also consider the HalfCheetah-v2 and Walker2d-v2 environment to compare the performance of explicit variance based evaluation and RED method. The results are presented in the Figure 6 and Figure 7 respectively. We can get the similar observations that EVB can either get higher reward values over iterations or much smaller variance. This further support the claim that EVB can provide a more robust and less sensitive reward function estimation.

5.4 Kernel Density Estimation

As a byproduct, we explore the numerical performance of the kernel density estimation method. The true mean episodic reward curve of RED and PCA based kernel density estimation method over the iteration steps under the HalfCheetah-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve. The graph is presented in Figure 8. The Walker2d-v2 environment is also considered in Figure 9. These results suggest that the kernel density based estimation method failed to extract more efficient reward function even though with much small variance and more stable estimation.

In addition, to further explore the kernel density based estimation method, we also present the extracted reward function curve in Figure 10. We can obviously observe that the reward curve here is not very smooth and it doesn't have a relatively flat central part. This could make the boundary of the decision not very clear and hurt the performanc of reinforcement leanring algorithm.

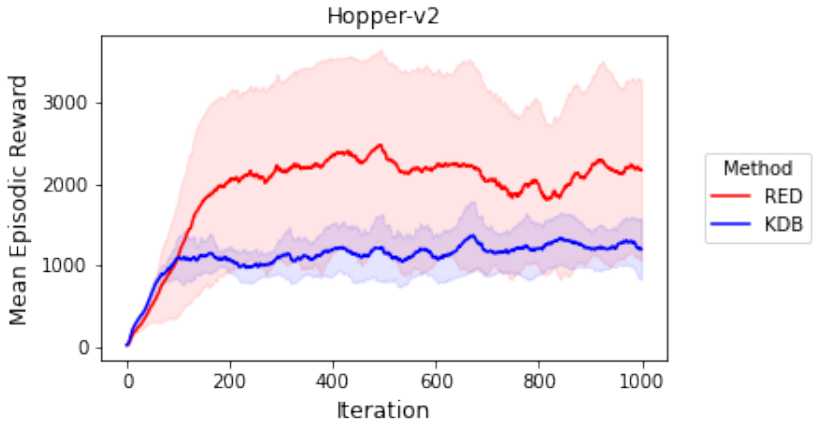

Figure 8: The true mean episodic reward curve of RED and PCA based kernel density estimation. method over the iteration steps under the HalfCheetah-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve.

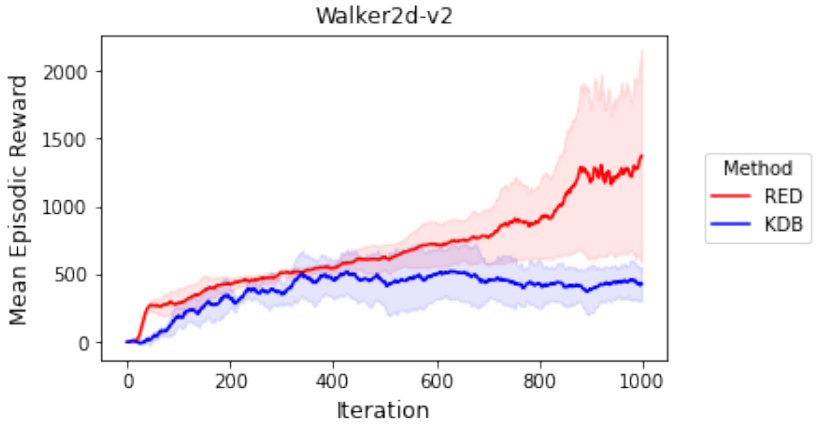

Figure 9: The true mean episodic reward curve of RED and PCA based kernel density estimation. method over the iteration steps under the HalfCheetah-v2 environment. Then standard deviation region is also plotted to visualize the variation of the reward curve.

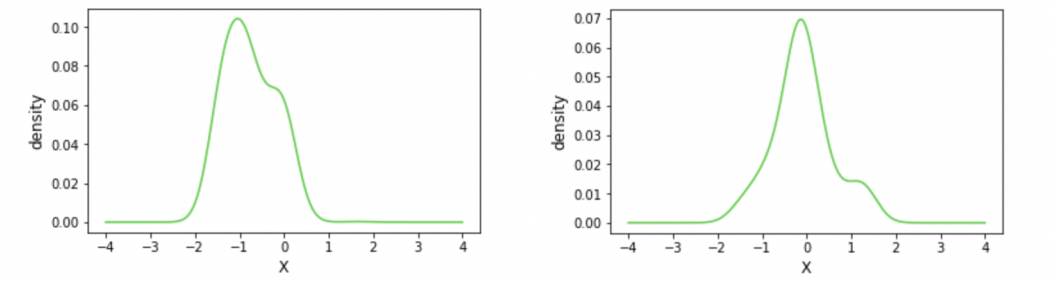

Figure 10: Learned density for the first PC: Hopper-v2(left) and Walker2d-v2(right) via kernel density and PCA.

6 Conclusions and Future Works

In this work we discussed the bias dominance and procedural variability problems regarding the practice of applying RND for reward learning purposes. As a solution, we propose to utilize an explicit variance estimation based expert policy support learning framework as a surrogate, which circumvents the theoretical issues and demonstrates great flexibility and numerical stability among a lot of reinforcement learning tasks. Besides, the proposed method can incorporate many powerful variance or density estimation schemes, such as linear model, random forest, neural networks and kernel estimations, to explore the hidden support information behind the datasets. We demonstrate the advantage of our proposed method with extensive numerical experiments. We observe that our proposed method can achieve higher rewards and also smaller variance compared with RED method. In addition, our method only requires very low computation cost, which further illustrate the efficiency.

The simulation results are actually quite inspiring. The original RED method use random network distillation with the tool of complex neural network structure for modeling. However, we turn our direction to more traditional statistical models including sieve basis model, random forest, and etc. Although these methods look relatively simple, they achieve promising performance than we have expected. We not only greatly improve the performance of the RED method, but also increase the calculation effacy because of the more simple structure. We also establish the theoretical guarantee for the proposed methods. To our knowledge, this is the first work to successfully connect the reward function extraction with expert data distribution in rigorous theories.

For future works, combining different approaches of recovering the expert's reward function appears to be a promising idea. Besides, extending the reward learning framework to higher dimensional datasets is another interesting land to conquer. Especially, we want to further improve the performance the kernel density estimation since the simulation performance in our experiments is not very promising. In addition, we also want to explore the possibility of combining the representation learning technique to extract important features from the dataset, and then pass it to reward function learning algorithm. This can have strong potential to solve high dimensional problems.

References

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3 (1):88–97, 1991.

Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338, 2019.

Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, page 103500, 2021.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

Ruohan Wang, Carlo Ciliberto, Pierluigi Vito Amadori, and Yiannis Demiris. Random expert distillation: Imitation learning via expert policy support estimation. In *International Conference on Machine Learning*, pages 6536–6544. PMLR, 2019.

Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas. A modern take on the bias-variance tradeoff in neural networks. *arXiv preprint arXiv:1810.08591*, 2018.

Durga L Shrestha and Dimitri P Solomatine. Machine learning approaches for estimation of prediction interval for the model output. *Neural Networks*, 19(2):225–235, 2006.

Stefan Wager, Trevor Hastie, and Bradley Efron. Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *The Journal of Machine Learning Research*, 15(1):1625–1651, 2014.

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

Grace Wahba. Optimal convergence properties of variable knot, kernel, and orthogonal series methods for density estimation. *The Annals of Statistics*, pages 15–29, 1975.

Peter Hall, JS Marron, and Byeong U Park. Smoothed cross-validation. *Probability theory and related fields*, 92(1): 1–20, 1992.

Simon J Sheather and Michael C Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):683–690, 1991.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Ziyi Huang, Henry Lam, and Haofeng Zhang. Quantifying epistemic uncertainty in deep learning. *arXiv preprint arXiv:2110.12122*, 2021.

Christopher J Paciorek. Bayesian smoothing with gaussian processes using fourier basis functions in the spectralgp package. *Journal of statistical software*, 19(2):nihpa22751, 2007.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

Richard C. Bradley. Basic properties of strong mixing conditions. A survey and some open questions. *Probab. Surv.*, 2: 107–144, 2005. ISSN 1549-5787. doi:10.1214/154957805100000104. Update of, and a supplement to, the 1986 original.

Jérôme Dedecker and Sana Louhichi. Maximal inequalities and empirical central limit theorems. 01 2002. doi:10.1007/978-1-4612-0099-4_3.

A Group Contribution

Lei was in charge of exploring the experiments of the kernel density estimation and also establish rigorous mathematical framework for the theoretical guarantee of explicit variance estimation method. Yunzhe worked on implementing the proposed EVB method on several different reinforcement learning settings to compare the performance with RED.

B Proof

Proposition B.1. *According to Lemma 4.1 in Dedecker and Louhichi [2002], suppose we have a sequence of stationary time series $U = (Z_1, Z_2, ..., Z_n)$ with common marginal distribution, where $Z_i \in \mathcal{R}^d$. If the $\beta-$mixing coefficient of this sequence satisfies that $\exists\, c$, s.t. $\beta(p) \leq c_1 e^{-c_2 p}$. Then there exists a sequence $U_i^0 = (Z_{ip+1}^0, Z_{ip+2}^0, ..., Z_{ip+p}^0)$ for $i \geq 0$ such that*

1. *U_i^0 has the same distribution as $U_i = (Z_{ip+1}, Z_{ip+2}, ..., Z_{ip+p})$*
2. *The sequence $\{U_{2i}^0\}_{i\geq 0}$ is i.i.d and so is $\{U_{2i+1}^0\}_{i\geq 0}$*
3. *For any $i \geq 0$, $P(U_i \neq U_i^0) \leq c\beta(p)$*

If we denote the empirical process as $G_n(U) := \sqrt{T}(\frac{1}{T}\sum_{i=1}^T f(Z_i) - \mathbb{E}f)$, without loss of generality, we assume that T is divisible by $2p$ and then we can get that

$$|G_n(U) - G_n(U^0)| \leq \frac{2||f||_\infty}{\sqrt{T}} \sum_{i=1}^T \mathbf{1}_{Z_i \neq Z_i^0} \tag{12}$$

So we have

$$P(|G_n(U) - G_n(U^0)| \neq 0) \leq P(\frac{2||f||_\infty}{\sqrt{T}} \sum_{i=1}^T \mathbf{1}_{Z_i \neq Z_i^0} \neq 0) \tag{13}$$

$$\leq \frac{T}{p} P(U_i \neq U_i^0) \tag{14}$$

$$\leq \frac{cT}{p}\beta(p) \tag{15}$$

Proposition B.2. *For a linear basis model $y_i = \alpha + (x_i - \bar{x})^\top \beta + \epsilon_i, i = 1, \cdots, n,\ \epsilon_i \overset{i.i.d.}{\sim} N(0, \sigma^2)$, the predicted variance of $E\{y \mid x\}$ at a new point x is given by*

$$\hat{V} = \frac{1}{n} + \frac{1}{n}(x - \bar{x})^\top S_{xx}^{-1}(x - \bar{x})$$

where

$$S_{xx} = \frac{1}{n}\sum_{i=1}^n (x_i - \bar{x})^\top (x_i - \bar{x}).$$

Proof. Let $\tilde{x}_i = x - \bar{x}_i$ and $\tilde{X} = (\tilde{x}_1, \cdots, \tilde{x}_n)^\top$. Also aggregate y_i into one vector $y = (y_1, \cdots, y_n)^\top$. Let $\mathbf{1}_n$ be the vector in $\mathbb{R}^n$ with all ones. The OLS estimates $(\hat{\alpha}, \hat{\beta})$ is given by

$$\hat{\alpha} = \bar{y},\ \hat{\beta} = (\tilde{X}^\top \tilde{X})^{-1}\tilde{X}^\top y.$$

Note that they are normally distributed, with covariance given by

$$\begin{aligned}&(\tilde{X}^\top \tilde{X})^{-1}\tilde{X}^\top(\sigma^2 I_n)(n^{-1}\mathbf{1}_n)\\ =&n^{-1}\sigma^2(\tilde{X}^\top \tilde{X})^{-1}\tilde{X}^\top \mathbf{1}_n = 0.\end{aligned}$$

The last equality is due to the fact that $\tilde{X}$ is zero-centered.

Therefore $\hat{\alpha} \perp \hat{\beta}$. The prediction for the conditional expectation $E(y \mid x)$ is

$$\hat{E}(x) = \hat{\alpha} + \hat{\beta}^\top (x - \bar{x}),$$

which has variance

$$\begin{aligned}\text{Var}(\hat{E}(x)) &= \text{Var}(\hat{\alpha}) + (x-\bar{x})^\top \text{Var}(\hat{\beta})(x-\bar{x}) \\ &= \frac{1}{n} + (x-\bar{x})^\top (\tilde{X}^\top \tilde{X})^{-1}(x-\bar{x}) \\ &= \frac{1}{n} + \frac{1}{n}(x-\bar{x})^\top (S_{xx})^{-1}(x-\bar{x}).\end{aligned}$$

□

Theorem B.1. *(Bidirectional Asymptotics) Suppose that we have n export trajectories $\{a_1^{(i)}, s_1^{(i)}, a_2^{(i)}, s_2^{(i)}, \cdots, a_T^{(i)}, s_T^{(i)}\}_1^n$ with same length T. Assume that each trajectory is independent and form a strictly stationary seires with β-mixing coefficients satisfy that $\beta(p) \leq c_1 \exp(-c_2 p)$ for some constants $c_1, c_2 > 0$. Let's denote the expert policy's true density as $q_E(s,a)$. By using the explicit variance evaluation technique in section 3.1 to extract the reward function $\hat{r}(s,a)$, we can prove that as either $n \to \infty$ or $T \to \infty$, with probability goes towards 1, we have $[\hat{r}(s_1,a_1) - \hat{r}(s_2,a_2)][q_E(s_1,a_1) - q_E(s_2,a_2)] \geq 0$ for any (s_1,a_1) and (s_2,a_2) tuples.*

Proof. With Proposition B.1 above, assume $N := n * T$, we can construct i.i.d sequece $x_1, x_2, \cdots, x_N$ with a cost of only $1/N$ rate loss of the probability. That is with probability at least $1 - c/N$, $x_1, x_2, \cdots, x_N$ should have the same marginal distribution with the original sequence. Then we turn to consider the explcit variance estimation. Let's firstly focus on the sieve basis model mentioned in section 3.2. With Proposition B.1 above, we can guarantee that as N goes towards infinity, the probability of $[d(s_1,a_1) - d(s_2,a_2)][\hat{\Omega}(s_1,a_1) - \hat{\Omega}d(s_2,a_2)] \geq 0$ will go to 1. Here, we define $d(s,a)$ as the Euclidean distance between the test point and the central point of the given fixed sample. According to the definition of the sample density, it should be negative correlated to the Euclidean distance between the test point and the central point. Thus, combing the two relations we have got, this can complete the proof. The similar proof procedure applies to other explicit variance evaluation methods. □