Review of Inverse Reinforcement Learning frameworks for Imitation Learning

Josselin Bonnevie
CentraleSupélec - ENS Paris-Saclay
josselin.bonnevie@student.ecp.fr

Shahine Bouabid
CentraleSupélec - ENS Paris-Saclay
shahine.bouabid@student.ecp.fr

Abstract

In Imitation Learning, an apprentice tries to mimic an expert behavior without knowledge of the objective driving its behavior. This can be addressed with supervised approaches using demonstrations as ground truth. Another approach, commonly known as Inverse Reinforcement Learning, suggests to learn the unknown reward function from expert demonstrations and derive a policy from it. Recent developments in inverse methods have made it increasingly appealing for imitation learning applications. In this paper, we aim at proposing a top-down approach overview of the existing frameworks, their guarantees and the challenges raised.

1 Introduction

In imitation learning, an apprentice tries to mimic an expert behavior in order to reproduce and generalize its decision-making process. Such tasks are usually framed into a Markovian decision process as it narrows the job to learning a reward function that makes the induced behavior closely imitate the observed one.

Imitation learning can be tackled with a passive supervised approach using demonstrations as ground truth. However, such models ignore the influence of the learned policy over next actions thus have poor behavior recovery when exploring new states, with a quadratic loss in the task horizon T [8]. Alternatively, interactive direct policy learning proposes to iteratively solve supervised learning problems constructed from roll out of previously trained policies and involves an expert to teach the apprentice how to recover from past mistakes. Recent approaches in that field can guarantee a loss linear in T [2]. Unfortunately, these solutions require an interactive feedback and querying an expert can be hard without feedback of his own action.

Another approach consists in addressing the imitation learning problem as an Inverse Reinforcement Learning (IRL) one, consistent with the Markovian decision process framework. IRL seeks to recover a reward out of an agent policy or behavior. More specifically, given a set of observed demonstrations and the ability to simulate the environment, it estimates the expert's reward function.

Still, IRL inherently suffers from a solution ambiguity problem : a given set of demonstrations can correspond to several reward functions and different policies can bee optimal for a same reward . The problem being fundamentally ill-posed, measuring the solution's accuracy is a key challenge of IRL.

We will present some of the main attemps to address the IRL problem for Imitation Learning within a common framework of energy maximization in a markov decision process and discuss their guarantees and limitations.

Preprint. Work in progress.

2 Problem Setup

2.1 Outline and notations

We consider the Markov decision process framework (MDP) defined by a tuple $\mathcal{M} = (S, A, P, R, \gamma, I)$ where S is a set of states, A a finite set of actions, P the model's dynamic i.e. the transition probabilities, R a reward function, $\gamma \in [0, 1]$ a discount factor and I an initial state distribution. A policy can either be deterministic $\pi : S \to A$ or stochastic $\pi : S \to \Delta(A)$, $\pi(\cdot \mid s)$ henceforth denoting a probability distribution over actions from state $s \in S$. Let $V^\pi : S \to \mathbb{R}$ the associated value function and its dual equivalent $\rho^\pi : S \times A \to \mathbb{R}$ the occupancy measure function. For the sake of simplicity, we will note R and ρ^π as functions of s only when the policy is deterministic.

In the context of IRL, the reward function considered is the expert's reward function R^* which we have no knowledge of. We hence work with the MDP without reward denoted $\mathcal{M}\backslash R^*$. The learner is provided with demonstrations from the expert $\mathcal{D} = (\zeta_i^*)_{i \in [\![1,n]\!]}$ where each trajectory $\zeta_i^* = \left((s_t^{(i)}, a_t^{(i)})\right)_{t \in [\![0,T]\!]}$ follows a target policy π^* assumed to be near-optimal, T being the time horizon.

2.2 Problem formalization

2.2.1 Preliminary results

Theorem 1. *Let $\pi \in \Delta(A)^S$ be a policy. Then it defines a unique probability distribution over the space of trajectories denoted p_π.*

Given theorem 1, if f is a function over policies we will denote interchangeably without ambiguity $f(\pi)$ and $f(p_\pi)$. Also, if ζ is a trajectory, $\hat{f}(\zeta)$ will be the empirical value of f over the trajectory. Consistently, if $\mathcal{Z}$ refers to a set of trajectories following some policy π, $\hat{f}(\mathcal{Z}) = \frac{1}{|\mathcal{Z}|} \sum_{\zeta \in \mathcal{Z}} \hat{f}(\zeta)$ provides a Monte Carlo estimate of $f(\pi)$.

2.2.2 General IRL framework

The goal of IRL can be to estimate the reward function best explaining the demonstrated behaviour or to recover to its best the expert's policy. The assessment of such performance hence requires a measure. To avoid misweighting single states decisions accordingly to their importance, the choice is made to measure the accuracy in terms of value function.

Formally, let $\mathcal{E}(\pi, R) = \mathbb{E}_{s \sim I}[V^\pi(s)]$. Under this choice of energy function, the IRL paradigm is stated by :

$$\text{Find } R \text{ s.t. } \pi^* \in \arg\max_\pi \mathcal{E}(\pi, R) \tag{1}$$

In what is following, we will consider the existence of a dummy initial state s_0 whose next state distribution is I such that $\mathcal{E}(\pi, R) = V^\pi(s_0)$.

Let's now formulate the reward learning problem. Our best knowledge of π^* is conveyed by the demonstrated trajectories. The latter can be leveraged by considering the empirical probability distribution it induces over the space of trajectories. Furthermore, Theorem 1 ensures there is an associated policy here denoted $\pi_\mathcal{D}$ such that $\mathcal{E}(\pi_\mathcal{D}, R) = \hat{V}^{\pi^*}(s_0)$.

Given this data-driven approximation of the expert's policy, we would want our estimate of the reward function to be the one for which our demonstrated policy performs at its best. We define this notion of performance by an objective function of the demonstrations policy and the estimated reward $J(\pi_\mathcal{D}, R)$. The reward learning problem is hence the one of J maximization modulo a reward regularization Γ

$$\hat{R} \in \arg\max_R J(\pi_\mathcal{D}, R) - \Gamma(R) \tag{$\mathcal{P}$}$$

Various attempts to solve this problem have been presented, differing by their parametrization of the loss, reward and learning method. We focus in this review on a three major optimization approaches : we will first introduce one of the earliest method based on linear programming, then move on to quadratic programming approaches to solve this problem through maximum margin optimization before introducing a probabilistic proposition relying maximum entropy principle.

2.2.3 Feature expectation

Many IRL methods rely on a set of feature functions $\phi = (\phi_i)_{i \in [\![1,d]\!]}$, $\phi_i : S \times A \to \mathbb{R}$ to model the reward function. Such functions are supposed to be provided and designed to reflect an element of the expert's decision making process, thus introducing into the model some prior knowledge.

A common parametrization of the reward function assumes it to be bounded and linear in the feature functions. Explicitely, we suppose the existence of a vector $\theta \in \mathbb{R}^d$ such that $R = \theta^\top \phi = \sum_{i=1}^d \theta_i \phi_i$.

For a given policy π, we denote by $\mu(\pi)$ its expected feature count defined by

$$\mu(\pi) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t \phi(s_t, a_t) \mid \pi\right] \tag{2}$$

Hence, by linearity if the expectation, we verify : $V^\pi(s_0) = \theta^\top \mu(\pi)$

Reducing the problem to the learning of a mapping from features to reward is appealing in terms of problem solving. However, such modelization suggests the features have been chosen carefully to properly embed the environment's relevant facets.

3 A Linear Programming approach

Ng and Russell [6] propose one of the first IRL algorithm. They suggest to favor solutions making step deviation with regard to optimal policy costly, one would hence want to penalize all policies but the demonstrated one under the current choice of reward function. This is done by introducing a penalization function $h = x \longmapsto x\mathbf{1}_{\mathbb{R}_-} + \nu x \mathbf{1}_{\mathbb{R}_+^*}$ with $\nu > 1$ and choosing for J :

$$J(\pi_\mathcal{D}, R) = \sum_\pi h(\mathcal{E}(\pi_\mathcal{D}, R) - \mathcal{E}(\pi, R))$$

where π lies in a finite set of policies.

By iteratively rolling out policies and optimizing the weights of the reward function so that stepping aside of the optimal path is penalizing, Ng. and Russell propose a linear programming algorithm to obtain an estimate of the optimal reward function.

The linear programming problem at step k is stated by :

$$\begin{aligned} \max_\theta \quad & \sum_{j=1}^k h\left(\hat{V}^{\pi^*}(s_0) - \hat{V}^{\pi_i}(s_0)\right) \\ \text{s.t.} \quad & |\theta_i| \leq 1\,,\ \ \forall i \in [\![1,d]\!] \end{aligned} \tag{LP$_k$}$$

The constraint on θ is ensuring the obtained reward function is bounded.

Algorithm 1 Ng and Russell [6]

```
 1: Initialize $k \leftarrow 1$ ; $\theta$ randomly
 2: Initialize $\hat{V}^{\pi^*} \leftarrow \theta^\top \mu(\mathcal{D})$
 3: Initialize base case random policy $\pi_1$ and roll out to collect trajectories $\mathcal{Z}_1$
 4: Initialize $\hat{V}^{\pi_1} \leftarrow \theta^\top \mu(\mathcal{Z}_1)$
 5: while Termination condition not met do
 6:     Solve (LP$_k$) to get $\theta$
 7:     Update $R$ with $\theta$
 8:     Derive $\pi_{k+1} = \arg\max_{\pi} V^\pi(s_0)$ under $R$
 9:     Roll out to collect trajectories $\mathcal{Z}_{k+1}$
10:     Evaluate $\hat{V}^{\pi^*} \leftarrow \theta^\top \mu(\mathcal{D})$
11:     for $j$ in $1 \dots k+1$ do:
12:         Evaluate $\hat{V}^{\pi_j} \leftarrow \theta^\top \mu(\mathcal{Z}_j)$
13:     $k \leftarrow k+1$
    return $\pi_k$
```

A key point of the proposed algorithm resides in the fact that any value estimation can be performed with Monte Carlo following $\mu(\mathcal{Z}_k) = \frac{1}{|\mathcal{Z}_k|} \sum_{\zeta \in \mathcal{Z}_k} \mu(\zeta)$.

The algorithm has been proven to be able to recover the reward function in complex environments. However, a caveat is in order as no convergence guarantee has been provided else than experimental results which may not be replicable to different environments settings.

4 Maximum Margin Optimization

4.1 General outline

Max-margin methods try to tackle the solution ambiguity dilemma by making the supplied demonstrations look better than any other policy by a margin, under the predicted reward. Doing so, we ensure the estimated reward function tends to induce policies similar to the expert's one.

We hence want the energy $\mathcal{E}(\pi_\mathcal{D}, R)$ to be greater than any other energy and also weight in how different a considered policy is from the demonstrated one. To do so, one may introduce a hinge objective function :

$$J(\pi_\mathcal{D}, R) = \mathcal{E}(\pi_\mathcal{D}, R) - \max_{\pi} \left[\mathcal{E}(\pi, R) - \mathcal{L}(\pi, \pi_\mathcal{D})\right]$$

where $\mathcal{L}$ defines a non-negative measure of closeness between policies. The reward function linearity in the feature functions is also supposed, leading to $\mathcal{E}(\pi, R) = \theta^\top \mu(\pi)$.

With such assumptions, ($\mathcal{P}$) can be rewritten as the constrained maximum margin problem stated by :

$$\begin{aligned} \max_{\eta\, \theta} \quad & \sum_{\pi} \eta_\pi - \Gamma(\theta) \\ \text{s.t.} \quad & \theta^\top \mu(\pi_\mathcal{D}) - \eta_\pi \geq \theta^\top \mu(\pi) - \mathcal{L}(\pi, \pi_D)\,, \ \ \forall \pi \end{aligned} \tag{MM}$$

In practice, π lies in a finite set of policies . We present in the following two approaches that have been suggested to solve this problem.

4.2 A Quadratic Programming approach

Abbeel and Ng [1] bring down the problem to the one of expected feature matching : the goal is to find π so that $\mu(\pi) \approx \mu(\pi^*)$. Indeed, if the reward function is constrained with $\|\theta\|_1 \leq 1$, then to get an ε precision on the value function difference, it suffices to do so on the expected feature count :

$$\forall \pi \in \Pi\ \forall \varepsilon > 0,\ \ \|\mu(\pi) - \mu(\pi^*)\|_2 \leq \varepsilon \Rightarrow \|V^\pi(s_0) - V^{\pi^*}(s_0)\|_2 \leq \varepsilon \tag{3}$$

Such choice ensures the complexity of the used samples only depends on the feature functions one. If specified with an L^2 regularisation on θ, it can be solved by rewriting (MM) as :

$$\begin{aligned} \max_{\eta\,\theta} \quad & \eta \\ \text{s.t.} \quad & \theta^\top \mu(\pi_{\mathcal{D}}) - \eta \geq \theta^\top \mu(\pi_i) \ \ \forall i \in [\![1, k]\!] \\ & \|\theta\|_2 \leq 1 \end{aligned} \tag{SVM$_k$}$$

The latter quadratic problem can be solved as a SVM. It is also assumed that our feature functions now take value in [0,1].

Algorithm 2 Abbeel and Ng [1]

```
 1: Initialize $\eta \leftarrow 0$ ; $k \leftarrow 1$ ; $\theta$ randomly
 2: Initialize $\hat{\mu}^* \leftarrow \mu(\mathcal{D})$
 3: Initialize base case random policy $\pi_1$ and roll out to collect trajectories $\mathcal{Z}_1$
 4: Initialize $\hat{\mu}_1 \leftarrow \mu(\mathcal{Z}_1)$
 5: while $\eta > \varepsilon$ do
 6:     Solve (SVM$_k$) to get $\theta$ and $\eta$
 7:     Update $R$ with $\theta$
 8:     Derive $\pi_{k+1} = \arg\max_\pi V^\pi(s_0)$ under $R$
 9:     Roll out to collect trajectories $\mathcal{Z}_{k+1}$
10:     Compute $\hat{\mu}_{k+1} \leftarrow \mu(\mathcal{Z}_{k+1})$
11:     $k \leftarrow k + 1$
    return $\pi_k$
```

Theorem 2. *Let $\mathcal{M}\backslash R$ be a MDP without reward, $\phi = (\phi_i)_{i \in [\![1,d]\!]}$, $\phi_i : S \times A \to [0,1]$ a set of feature functions and any $\varepsilon > 0$, $\delta > 0$ given. Let m the number of Monte Carlo sample used in Algorithm 2. Then, for the algorithm to terminate in a finite number of steps K with probability at least $1 - \delta$ and return a policy $\hat{\pi}$ so that for any true reward function $R^* = \theta^{*\top}\phi$ ($\|\theta\|_1 \leq 1$) we have $|V^{\pi^*}(s_0) - V^{\hat{\pi}}(s_0)| \leq \varepsilon$, it suffices that :*

$$m \geq \frac{2d}{(1-\gamma)^2\varepsilon^2} \log \frac{2d}{\delta}$$

Plus, we verify $K = \mathcal{O}\left(\frac{d}{(1-\gamma)^2\varepsilon^2} \log \frac{d}{(1-\gamma)\varepsilon}\right)$

Theorem 2 hence provides us with guarantees in terms of convergence and accuracy. However, Abbeel and Ng stress out that relying on the expected feature count creates a high dependence on the quality of the policy estimation.

4.3 Max-margin Planning

Ratliff et al. [7] suggest another way to formalize the structured prediction problem over policies. But unlike Abbeel and Ng [1], their proposition is designed to allow the demonstrations to come from different MDPs. Hence, multiple feature maps and start/goal states can be introduced, bolstering the algorithm's ability to generalize.

Formally, we define a set of MDPs $(\mathcal{M}_i)_{i \in [\![1,n]\!]}$, $\mathcal{M}_i = (S_i, A_i, P_i, R^*_i, \gamma_i, I_i)$ endowed with their own set of feature functions $(\phi^{(i)}_j)_{j \in [\![1,d]\!]}$. It is suggested that each trajectory $\zeta^*_i \in \mathcal{D}$ could be driven by a different policy π^*_i specific to $\mathcal{M}_i$. Having one MDP and policy per demonstrated trajectory allows us to get rid of the strong near-optimal behavior assumption

Each MDP also has its own loss vector $\ell_i \in \mathbb{R}^{|S_i| \times |A_i|}$ inducing a non-negative loss function $\mathcal{L}_i = \pi \longmapsto \ell_i^\top \rho^\pi$. The latter conveys a notion of closeness with the i-th demonstrated policy. It hence seems natural to define ℓ_i as a payoff if occupancy measure vector fails to match expert's occupancy measure. Ratliff et al. proposes to do so by counting the number of state-action pairs visited by the apprentice that were not by the expert and point out that performances are improved when this is smoothed to allow nearby paths too.

The problem formulation is framed in Taskar et al. [9] structured large margin learning :

$$\begin{aligned} \min_{\eta\,\theta} \quad & \frac{\lambda}{2}\|\theta\|_2^2 + \frac{\kappa}{n}\sum_{i=1}^{n} \beta_i \eta_i^q \\ \text{s.t.} \quad & \theta^\top \hat{\mu}(\zeta_i^*) + \eta_i \geq \max_{\zeta} \theta^\top \hat{\mu}(\zeta) + \ell_i^\top \hat{\rho}^\zeta \ \ \forall i \in [\![1, n]\!] \end{aligned} \tag{MMP$_k$}$$

where λ is a regularization parameter, κ scales constraints violation penalty, β_i are normalizing scalars and $q \in \{1, 2\}$ allows L^1 and L^2 slack variables penalty.

Note that unlike the previous problem, (MMP$_k$) cannot be solved as a SVM and requires a full quadratic program solving method. However, the number of constraints being proportional to $|\mathcal{D}|$ times the cardinality of state-action pairs, Ratliff et al. suggest an efficient optimization through a subgradient approach.

Algorithm 3 Ratliff et al. [7]

```
1: Initialize hyperparameter $\lambda$, learning rate $\alpha_t$, horizon $T$
2: Initialize $t \leftarrow 1$ and $\theta \leftarrow 0$
3: while $t \leq T$ do
4:     for $i$ in $1 \dots n$ do:
5:         Update $R_i$ with $\theta$
6:         Derive $\hat{\pi}_i^* = \arg\max_{\pi} \mathcal{E}(\pi, R_i) + \mathcal{L}_i(\pi)$
7:     Update $\theta \leftarrow \theta - \alpha_t \cdot$ subgradient$(\hat{\pi}_1^*, \dots, \hat{\pi}_n^*)$
8:     $t \leftarrow t + 1$
   return $\theta$
```

The solution being inherently agnostic about a real underlying MDP, the yielded weight vector θ is prone to extrapolate to new feature maps and goals. Moreover, Ratliff et al. provide guarantee of a linear and sublinear convergence in the number of steps for respectively batch and online settings.

5 Maximum Entropy Optimisation

5.1 Expected Feature counts and occupancy measure matching

Abbeel and Ng [1] have shown that in the case of a reward function that is linear in some state-action features, expected feature matching allows us to recover a policy that will perform as well as the expert's policy for any (linear) reward.

This setting also includes the general case of a reward of any shape. Indeed the expected reward can always be decomposed using occupancy measures as follows.

$$\mathcal{E}(\pi_{\mathcal{D}}, R) = \sum_{(s,a)} \rho^\pi(s, a) R(s, a)$$

And occupancy measure can be considered as the expected feature count associated with the state-action indicators.

$$\rho^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t \mathbf{1}_{(s_t, a_t) = (s, a)} \right]$$

Therefore in the general case of a reward of any shape, feature matching corresponds to occupancy measure matching. In the rest of the section, expected feature count can refer to both occupancy measures or expected feature counts, unless explicitly specified.

However the feature matching problem is ill-posed since several policies can match a given feature count. This is true in particular if the demonstrated policy is not optimal for any reward.

To solve this under-constrained problem, [12] proposed a method based on maximum entropy which allows to recover both a reward and a policy imitating the behavior of the expert.

5.1.1 Deterministic MDP :

We first consider the case of a deterministic MDP with known dynamics. And we consider a given policy π through its corresponding induced probability distribution on trajectories p_π. The maximum entropy principle applied to inverse reinforcement learning suggests to chose the trajectory distribution with maximal entropy under the constraint of feature count matching.

$$\max_{p_\pi} H(p_\pi) \quad \text{s.t.} \quad \mu(p_\pi) = \hat{\mu}(\mathcal{D}) \tag{4}$$

This allows to recover a policy that matches the performance of the demonstrated policy while containing minimal additional information. As a consequence, in the case of a deterministic MDP, such a policy gives the same probability to paths that have the same cumulative reward.

What's more, the problem of entropy maximization under expected feature matching is the dual of the problem of maximum likelihood in the exponential family. Therefore the solution is a distribution that associates to each trajectory a probability exponentially proportional to its cumulative reward. The parameter θ is estimated through maximum likelihood.

$$\hat{p}(\zeta) = \frac{e^{\theta^\top \mu(\pi_\mathcal{D})}}{Z(\theta)} \quad \text{with } Z(\theta) = \int e^{\theta^\top \mu(\zeta)} d\zeta \tag{5}$$

This distribution assigns a strictly positive probability to all the trajectories.

This way to approach the formulation the formulation does not make any hypothesis about the behaviour of the agent with respect to a true unknown reward. We just imitate the demonstrated policy in terms of feature count, with the guarantee that the recovered policy will perform exactly like the demonstrated policy for every reward. But the problem can equivalently be thought of in the following way, assuming a certain behaviour of the expert with respect to the unknown reward.

$$\text{Find } R \ \text{ s.t. } \ \pi_\mathcal{D} \in \arg\max_\pi H(p_\pi) + \mathcal{E}(\pi, R) \tag{6}$$

Which assumes that the demonstrating agent behaves near optimally with respect to the reward R^*. This also allows to model noise in the demonstrations. By solving problem (4), the recovered reward $\hat{R}$ and the recovered policy $\hat{\pi}$ do correspond to the original reward R^* and the original policy $\pi_\mathcal{D}$ when the estimation of the feature count is perfect ($\hat{\mu}(\mathcal{D}) = \mu(\pi^*)$).

5.2 Non deterministic MDPs

In a non deterministic MDP, the transition probabilities between states introduce some randomness to the trajectories. However the idea of the method is to maximize the entropy coming from the policy itself and not from the MPD.

5.2.1 Conditioned Entropy

In a first attempt to extend the maximum entropy principle to non deterministic MDPs, Ziebart et al. [12] suggested maximizing the entropy of the trajectory distribution conditioned on the transition distribution. Let $\mathcal{T}$ be the space of possible action outcome and o an outcome sample. This means that o specifies the outcome of every action for every timestep. Therefore $\mathcal{M}$ is deterministic given o. By applying the maximum entropy principal to the conditional trajectory distribution $p(\zeta \mid o)$ and by neglecting the effect of the transition randomness so as to make the expression tractable, we get the following expression.

$$\begin{aligned} p(\zeta \mid \theta) &= \sum_{o \in \mathcal{T}} P(o) \frac{e^{\theta^\top \hat{\mu}(\zeta)}}{Z(\theta, o)} \mathbf{1}_{\zeta \in o} \\ &\approx \frac{e^{\theta^\top \hat{\mu}(\zeta)}}{Z(\zeta)} \prod_{t \geq 0} P(s_{t+1} \mid a_t, s_t) \end{aligned} \tag{7}$$

Similarly θ is estimated by maximum likelihood using gradient ascent.

It should be noted that this method fails to assign the same probability to all trajectories yielding the same global reward. Which motivates the introduction of the concept of Causal Entropy.

5.2.2 Causal entropy

In the case of a non-deterministic MDP, the previous method does not yield equal probabilities of occurrence for equally rewarded trajectories. We were maximizing the entropy of the trajectory distribution marginalized over future states. Ziebart et al. [13] argue that this method does not take into account the sequential structure of the data. By conditioning on future states, we let future information that is not yet available at time t influence the actions of the agent at time t. They present a generalization of the concept of entropy maximization that takes into account the sequential revelation of side information (here the realization of the transitions in the MDP). The concept was initially introduced for finite time horizons and was extended to infinite time horizon MDPs by Zhou et al. [11].

The γ-discounted causal entropy for a factor $\gamma \in (0, 1)$ is defined as :

$$\tilde{H}^{\gamma}(\pi) = \mathbb{E}\left[-\sum_{t\geq 0}\gamma^t \log \pi(a_t \mid s_t)\right] \tag{8}$$

In the case of linear rewards, the corresponding Maximum Causal Entropy Inverse Reinforcement learning can be formulated as follows.

$$\hat{\pi} = \arg\max_{\pi} \tilde{H}^{\gamma}(\pi) \;\text{ s.t. }\; \mu(\pi) = \hat{\mu}(\mathcal{D}) \tag{9}$$

It should be noted that in the case of a deterministic MDP, this problem is equivalent to the problem of Maximum Entropy Inverse Reinforcement Learning from 5.1.1.

5.2.3 Mathematical analysis

When strong duality holds in MCE IRL (9), which is the case in most cases we will encounter, the solution has the form of a soft-Bellman policy defined in the following way :

$$\hat{\pi}(a \mid s) = \exp\left(Q^{soft}_{\hat{\theta}}(s,a) - V^{soft}_{\hat{\theta}}(s)\right) \tag{10}$$

Where $Q^{soft}_{\hat{\theta}}$ and $V^{soft}_{\hat{\theta}}$ verify the soft Bellman equations

$$\begin{aligned} Q^{soft}_{\hat{\theta}}(s,a) &= \theta^{\top}\phi(s,a) + \beta \sum_{s'\in\mathcal{S}} P(s' \mid s,a) V^{soft}_{\hat{\theta}}(s') \\ V^{soft}_{\hat{\theta}}(s) &= \operatorname*{softmax}_{a\in A} Q^{soft}_{\hat{\theta}}(s,a) = \log\left(\sum_{a\in A}\exp(Q^{soft}_{\hat{\theta}}(s,a))\right) \end{aligned} \tag{11}$$

The parameter $\hat{\theta}$ is estimated by maximum likelihood on the demonstrations. The problem being concave it can be computed as previously through gradient ascent.

Theorem 3. *If the expert policy π^* is known only through it's exact expected feature count $\mu(\pi^*)$. A maximum discounted causal entropy policy $\hat{\pi}$ minimizes the worst-case discounted log-loss when predicting the actions taken by π^*.*

$$\hat{\pi} \in \arg\min_{\pi}\sup_{\tilde{\pi}} \mathbb{E}_{\tilde{\pi}}\left[\sum_{t\geq 0} -\beta^t \log \pi(a_t \mid s_t)\right]$$

Those guarantees do not hold when maximizing the entropy conditioned on future states as exposed in section 5.2.1.

5.3 Resolution

The gradient of the demonstration's loglikelihood can be expressed as follows.

$$\nabla_\theta L(\theta) = \hat{\mu}(\mathcal{D}) - \mu(\pi_\theta) = \hat{\mu}(\mathcal{D}) - \sum_{(s,a)\in S\times A} \rho^\theta(s,a)\phi(s,a) \tag{12}$$

To compute this gradient, it is necessary to first solve the MDP and compute the policy associated with the current value of parameter θ. And then to estimate $\mu(\pi_\theta)$, or equivalently ρ^θ. One of the main drawbacks of those techniques is the computational cost of solving a RL problem in a loop.

5.3.1 Solving the MDP

Two main methods have been proposed to solve the MDP efficiently :

- Ziebart et al. [13] proposed a recursive dynamic programming algorithm which can be seen as an adaptation of value algorithm to Soft Bellman equations (11). It requires the knowledge of the MDP's dynamics.
- Zhou et al. [11] propose an adaptation of Q-learning to Soft Bellman equations which enables to estimate the policy with no knowledge of the MDP's dynamics.

5.3.2 Estimating the occupancy measures

Two main methods have been proposed to compute the corresponding occupancy measures :

- Ziebart et al. [13] proposed another dynamic programming based algorithm which recovers occupancy measures for a given policy when the MDP's dynamics are known.
- An other approach often used in model free settings is to estimate occupancy measures by Monte Carlo

5.3.3 Regularization

The expected feature count of the expert's policy is estimated from a finite set of demonstrations, therefore $\hat{\mu}(\mathcal{D})$ is only an approximation of $\mu(\pi^*)$ and exact expected feature count matching would lead to over fitting. For this reason it is necessary to regularize.

The results of Dudík and E. Schapire [3] about regularization in entropy maximization problems given bounded uncertainty on feature expectations can be extended to maximum causal entropy. They especially recommend using L1 regularization.

5.4 Deep Inverse Reinforcement Learning by maximum entropy

Methods assuming the linearity of the reward in some features $\phi(s,a)$ strongly rely on the quality of those features. Therefore a lot of effort needs to be put in engineering the set of features. However estimating individually the reward of each state-action pair is not possible either. It would not be tractable in MDPs with large state action spaces.

We still would like similar state-actions to have similar rewards so as to enable generalization. Wulfmeier et al. [10] propose restricting the space of possible rewards to a space of complex function in some raw state-action features, implemented by a neural network. Those features can be high dimensional, for instance raw visual input in the case of a robotic task. The choice of the network's architecture allows to add some prior information into the problem while requiring less engineering work than hand crafted features.

Let $\mathbf{f}_(s,a)$ be the raw, possibly high dimensional features representing the state-action couple (s,a). Deep Maximum Entropy Inverse Reinforcement Learning recovers a policy which induces a trajectory distribution expressed in the following way.

$$\hat{p}(\zeta) \propto \exp\left(\sum_{t\geq 0} \gamma^t R_\theta(\mathbf{f}(s_t,a_t))\right) \tag{13}$$

The parameters θ of the network are estimated in the same way as previously by back propagating the gradient to the weights of the network. In the original paper, different sort of regularization was used (L1, L2 and dropout).

The main advantage of this method is that it overcomes the need for manually engineered features. It proves especially efficient in cases where such good features are hard to design. For tasks where very good handcrafted features exist, Maximum Entropy Deep Inverse Reinforcement Learning requires more demonstrations data than Linear Maximum Entropy Inverse Reinforcement Learning.

6 Generative Adversarial Imitation Learning

Most of the time in Imitation Learning, the true quantity of interest is the policy recovered after running RL on the recovered reward. Starting from this observation,Ho and Ermon [5] developed an IRL based method to directly recover the policy without explicitly computing a reward. Their problem formulation unifies several of the methods exposed in this paper and draws parallels between the inverse reinforcement learning and the Generative Adversarial Network framework. The two main advantages of the method are the following :

- Contrarily to many methods, it does not require solving a full MDP in a loop
- It is model free : it does not require knowledge of the MDP's dynamics
- It takes advantage of the expressive power of neural networks as linear reward are limiting for previously mentioned reasons

6.1 The General Framework

The general framework introduced in [5] adopts and generalizes maximum causal entropy IRL. Let's use the following IRL formulation, with $\mathcal{C}$ a given class of functions.

$$\min_{R \in \mathcal{C}} \left(\max_{\pi \in \Pi} \hat{H}^\gamma(\pi) + \mathbb{E}_\pi[R(s,a)] \right) - \mathbb{E}_{\pi^*}[R(s,a)] \tag{14}$$

When $\mathcal{C}$ is the set of linear functions in the features $\phi(s,a)$, this is equivalent to maximum causal entropy IRL with linear rewards (expected feature count matching). When $\mathcal{C}$ is the set of all possible functions. We recover the more general case of maximum causal entropy IRL with occupancy measure matching, which is not desirable for two main reasons :

- When the number of state-action pairs is large this won't be tractable
- The demonstration set is limited and exact occupancy measure matching would result in over-fitting (only visiting the states visited by the expert during the demonstration)

This is why we introduce a general regularizer ψ and consider the new problem.

$$\min_{R \in \mathcal{C}} \left(\max_{\pi \in \Pi} \hat{H}^\gamma(\pi) + \mathbb{E}_\pi[R(s,a)] \right) - \mathbb{E}_{\pi^*}[R(s,a)] \underbrace{+ \psi(R)}_{\text{Regularization}} \tag{15}$$

We also consider the corresponding RL problem which we will use to recover a policy from the recovered reward.

$$RL(R) = \arg\max_{\pi} H(\pi) + \mathbb{E}_\pi[R(s,a)] \tag{16}$$

The policy recovered by running RL after IRL in this setting is characterized as follows.

Theorem 4. *Denoting ψ^* the convex conjugate of ψ*

$$RL \circ IRL_\psi(\pi^*) = \arg\max_{\pi} \hat{H}^\gamma(\pi) - \psi^*(\rho^\pi - \rho^{\pi^*}) \tag{17}$$

Running RL after IRL recovers a policy with large causal entropy which occupancy measure is close to the expert's as quantified by ψ^*. Different choices of ψ or equivalently ψ^* lead to different problems.

6.2 Generative Adversarial Imitation Learning

Ho and Ermon [5] introduce the following regularization function :

$$\begin{aligned} \psi_{GA}(R) &= \begin{cases} \mathbb{E}_{\pi^*}[g(R(s,a))] & \text{if } R > 0 \\ +\infty & \text{otherwise} \end{cases} \\ \text{With } & g(R) = R - \log(1 - e^{-R}) \text{ for all } R > 0 \end{aligned} \tag{18}$$

This regularization is an expectation over the expert's policy and therefore is able to adjust to the expert's demonstration. It penalizes heavily policies that assign an almost null reward to the expert's trajectories while also gently penalizing rewards that are too high. What's more it enables to recover a penalization on occupancy measure that corresponds to the Jensen-Shannon divergence up to a constant. This divergence corresponds to a symmetric version of the Kullback Leibler divergence.

$$\begin{aligned} \psi^*_{GA}(\rho^\pi - \rho^{\pi^*}) &= \max_{D \in (0,1)^{S \times A}} \mathbb{E}_\pi[\log D(s,a)] + \mathbb{E}_{\pi^*}[\log(1 - D(s,a))] \\ &= \underbrace{D_{KL}\left(\rho^\pi \| (\rho^\pi + \rho^{\pi^*})/2\right) + D_{KL}\left(\rho^{\pi^*} \| (\rho^\pi + \rho^{\pi^*})/2\right)}_{D_{\text{JS}}} + \text{constant} \end{aligned} \tag{19}$$

And the final formulation of Generative Adversarial Imitation Learning can be formulated as follows.

$$\text{Find } \hat{\pi} \in \arg\min_\pi D_{\text{JS}}(\rho^\pi, \rho^{\pi^*}) - \lambda \tilde{H}^\gamma(\pi) \tag{20}$$

Or equivalently,

$$\hat{\pi} \in \arg\min_\pi \max_{D \in (0,1)^{S \times \mathcal{A}}} \mathbb{E}_\pi[\log(D(s,a)] + \mathbb{E}_{\pi^*}[\log(1 - D(s,a)] - \lambda \tilde{H}^\gamma(\pi) \tag{21}$$

This formulation can be interpreted as a causal entropy regularized version of Generative Adversarial Networks [4], with $\lambda > 0$, a parameter controlling the strength of the regularization. We simultaneously train a discriminating classifier D to distinguish the expert's demonstration from trajectories generated by other policies; and we train the policy $\hat{\pi}$ to both confuse the discriminator D and maximize causal entropy.

In practice, D and π are implemented by neural networks and the expectancies are estimated by empirical means. The min max formulation is solved by alternating gradient step on D and π.

7 Conclusion

To conclude, since the works of Ng and Russell in 2000, IRL has proven to be a brilliant alternative to direct policy learning in Imitation Learning tasks. Its ability to mitigate the dependency on an expert that can be queried during training makes it worth being explored. Several works have been tackling its resolution and formalization, especially when it comes to solving its inherent ambiguity dilemma. In particular, links established with generative methods provide an appealing approach of IRL in the wake recent developments in that field.

We let for future work, comparison of these algorithms at performing similar tasks.

References

[1] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04,

pages 1–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015430. URL `http://doi.acm.org/10.1145/1015330.1015430`.

[2] A. Attia and S. Dayan. Global overview of imitation learning. 2018. URL `https://arxiv.org/abs/1801.06503v1`.

[3] Miroslav Dudík and Robert E. Schapire. Maximum entropy distribution estimation with generalized regularization. pages 123–138, 09 2006. doi: 10.1007/11776420_12.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

[5] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *NIPS*, 2016.

[6] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-707-2. URL `http://dl.acm.org/citation.cfm?id=645529.657801`.

[7] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 729–736, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143936. URL `http://doi.acm.org/10.1145/1143844.1143936`.

[8] Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `http://proceedings.mlr.press/v9/ross10a.html`.

[9] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 896–903, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: 10.1145/1102351.1102464. URL `http://doi.acm.org/10.1145/1102351.1102464`.

[10] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. 2015.

[11] Zhengyuan Zhou, Michael Bloem, and Nicholas Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. *IEEE Transactions on Automatic Control*, 63: 2787–2802, 2018.

[12] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438, 2008.

[13] Brian D. Ziebart, J. Andrew Bagnell, and Anind K. Dey. Modeling interaction via the principle of maximum causal entropy. In *ICML*, 2010.

A Proofs of Theorems

A.1 Proof of Theorem 1

Proof. Let $\zeta = ((s_t, a_t))_{t\geq 0}$ and $\zeta_t = ((s_u, a_u))_{u\in [\![0,t]\!]}$

Then,

$$\forall t \geq 0,\ p(\zeta_{t+1}) = p(s_{t+1}, a_{t+1} \mid \zeta_t)p(\zeta_t) = \pi(a_{t+1} \mid s_{t+1})p(\zeta_t)$$

By induction, we hence prove that any policy induces a probabilty distribution over trajectories defined by $p(\zeta) = \prod_{t\geq 0} \pi(a_t \mid s_t)p_I(s_0)$.

Let's now show that any such distribution is induced by a unique policy. Let $q \in \Delta(A)^S$ a policy, we denote p_π a distribution induced by π.

Keeping the same notations as previously,

$$\begin{aligned}
D_{KL}(p_\pi \| p_q) &= \sum_\zeta p_\pi(\zeta) \log \frac{p_\pi(\zeta)}{p_q(\zeta)} \\
&= \sum_\zeta \sum_{t\geq 0} p_\pi(\zeta) \log \frac{\pi(a_t \mid s_t)}{q(a_t \mid s_t)} \\
&= \sum_{t\geq 0} \sum_{s_t,a_t} \rho^\pi(s_t,a_t) \log \frac{\pi(a_t \mid s_t)}{q(a_t \mid s_t)} \\
&= \sum_{t\geq 0} \sum_{s_t} \rho^\pi(s_t) \sum_{a_t} \pi(a_t \mid s_t) \log \frac{\pi(a_t \mid s_t)}{q(a_t \mid s_t)} \\
&= \sum_{t\geq 0} \mathbb{E}_{s_t\sim\rho^\pi(\cdot)} [\underbrace{D_{KL}(\pi(\cdot \mid s_t) \| q(\cdot \mid s_t))}_{\geq 0}]
\end{aligned}$$

By nonnegativity of the divergence, $p_\pi = q \Leftrightarrow \forall t \geq 0,\ \forall s_t,\ \pi(\cdot \mid s_t) = q(\cdot \mid s_t) \Leftrightarrow \pi = q$ which proves the stated result. $\square$

A.2 Proof of Theorem 2

Given a set of policies Π, let $M(\Pi) = \text{Conv}\{\mu(\pi), \pi \in \Pi\}$ the convex set of feature expectations. In the following, for the sake of simplicity, we only consider the case where $\hat{\mu}^* \in M(\Pi)$. This is not necessarily true and one can find an extended version of the proof in Abbeel and Ng [1].

Similarly, at iteration k of Algorithm 2, let $M_k = \text{Conv}\{\mu(\pi_j), j \in [\![1,k]\!]\}$ the convex set of feature expectations of policies up to step k. Then, the feature expectation estimate returned by the max-margin algorithm $\hat{\mu}_k$ is the orthogonal projection of $\hat{\mu}^*$ on convex set M_k.

Lemma 5 (Single step improvement)**.** *Let $k \in \mathbb{N}^*$ and $\hat{\mu}_k \in M_k$.*

Let π_{k+1} be the optimal policy of $\mathcal{M}\backslash R$ augmented with reward function $R = (\hat{\mu}^* - \hat{\mu}_k)^T\phi$ i.e

$$\pi_{k+1} = \arg\max_\pi (\hat{\mu}^* - \hat{\mu}_k)^T \mu(\pi) \tag{22}$$

and $\hat{\mu}_{k+1}$ be the orthogonal projection of $\hat{\mu}^*$ on M_{k+1}. Then

$$\frac{\|\hat{\mu}^* - \hat{\mu}_{k+1}\|_2}{\|\hat{\mu}^* - \hat{\mu}_k\|_2} \leq \frac{d}{\sqrt{d^2 + (1-\gamma)^2\|\hat{\mu}^* - \hat{\mu}_k\|_2^2}} \tag{23}$$

Proof. Without loss of generality, let's make $\hat{\mu}_k$ the origin of our coordinate system.

Let $\mu_{k+1} = \mu(\pi_{k+1})$ and $\tilde{\mu}_{k+1}$ be the projection of $\hat{\mu}^*$ onto the line through and $\hat{\mu}_k$ and μ_{k+1} i.e

$$\tilde{\mu}_{k+1} = \frac{(\hat{\mu}^* - \hat{\mu}_k)^T(\mu_{k+1} - \hat{\mu}_k)}{\|\mu_{k+1} - \hat{\mu}_k\|_2^2}(\mu_{k+1} - \hat{\mu}_k) + \hat{\mu}_k = \frac{\hat{\mu}^{*T}\mu_{k+1}}{\|\mu_{k+1}\|_2^2}\mu_{k+1}$$

Let $\lambda = \frac{\hat{\mu}^{*T}\mu_{k+1}}{\|\mu_{k+1}\|_2^2}$, so that $\tilde{\mu}_{k+1} = \lambda\mu_{k+1} + (1-\lambda)\hat{\mu}_k$

Since (22), $\hat{\mu}^{*T}\mu_{k+1} \geq \|\mu_{k+1}\|_2^2 > 0 \Rightarrow \lambda \geq 0$. Plus, we also get with Cauchy Schwartz inequality : $\lambda \leq \frac{\|\hat{\mu}\|_2}{\|\mu_{k+1}\|_2} \leq 1$. $\tilde{\mu}_{k+1}$ is hence as convex combinaison of μ_{k+1} and $\hat{\mu}_k$ and as they both are in M_{k+1}, $\tilde{\mu}_{k+1} \in M_{k+1}$.

Given that $\hat{\mu}_{k+1}$ is the orthogonal projection of $\hat{\mu}^*$ on M_{k+1}, it entails that

$$\|\hat{\mu}^* - \hat{\mu}_{k+1}\|_2 \leq \|\hat{\mu}^* - \tilde{\mu}_{k+1}\|_2 \tag{24}$$

Furthermore,

$$\frac{\|\hat{\mu}^* - \tilde{\mu}_{k+1}\|_2^2}{\|\hat{\mu}^*\|_2^2} = \frac{\left\|\hat{\mu}^* - \frac{\hat{\mu}^{*T}\mu_{k+1}}{\|\mu_{k+1}\|_2^2}\mu_{k+1}\right\|_2^2}{\|\hat{\mu}^*\|_2^2} \tag{25}$$

$$= 1 - \frac{(\hat{\mu}^{*T}\mu_{k+1})^2}{\|\hat{\mu}^*\|_2^2\|\mu_{k+1}\|_2^2} \tag{26}$$

$$= \frac{\|\mu_{k+1}\|_2^2 - \frac{(\hat{\mu}^{*T}\mu_{k+1})^2}{\|\hat{\mu}^*\|_2^2}}{\|\mu_{k+1}\|_2^2} \tag{27}$$

$$\leq \frac{\|\mu_{k+1}\|_2^2 - 2\hat{\mu}^{*T}\mu_{k+1} + \|\hat{\mu}^*\|_2^2}{\|\mu_{k+1}\|_2^2} \tag{28}$$

$$= \frac{\|\hat{\mu}^* - \mu_{k+1}\|_2^2}{\|\hat{\mu}^* - \mu_{k+1}\|_2^2 + \|\hat{\mu}^*\|_2^2 + 2(\mu_{k+1} - \hat{\mu}^*)^T\hat{\mu}^*} \tag{29}$$

$$\leq \frac{\|\hat{\mu}^* - \mu_{k+1}\|_2^2}{\|\hat{\mu}^* - \mu_{k+1}\|_2^2 + \|\hat{\mu}^*\|_2^2} \tag{30}$$

$$\leq \frac{d^2/(1-\gamma)^2}{d^2/(1-\gamma)^2 + \|\hat{\mu}^*\|_2^2} \tag{31}$$

25 : replacing $\tilde{\mu}_{k+1}$ by its fomula 26 : expanding 27 : mutiply by $\|\mu_{k+1}\|_2^2/\|\mu_{k+1}\|_2^2$ 28 : $\left(a^Tb - \|b\|_2^2\right)^2 \geq 0 \Leftrightarrow \frac{-\left(a^Tb\right)^2}{\|b\|_2^2} \leq -2a^Tb + \|b\|_2^2$ 29 : rewritting 30 : $\pi_{k+1} = \arg\max_{\pi} \hat{\mu}^{*T}\mu(\pi)$ as $\hat{\mu}_k$ is the origin and $\mu_{k+1} = \mu(\pi_{k+1})$, hence $\hat{\mu}^{*T}\mu_{k+1} \geq \|\hat{\mu}^*\|_2^2$ 31 : All points lie in $M(\Pi) = [0, \frac{1}{1-\gamma}]^d$, so their norms are bounded by $d/1-\gamma$. Adding up with (24) provides the desired result.

□

Proof of Theorem 2. Let's first show that the algorithm terminates w.r.t the Monte Carlo estimate of the optimal expected feature count $\hat{\mu}^* = \mu(\mathcal{D})$.

Let $k \in \mathbb{N}^*$, $\varepsilon > 0$ our threshold and suppose the stopping criterion has not been met, $\|\hat{\mu}^* - \hat{\mu}_k\|_2 \geq \varepsilon$

Given $\hat{\mu}_k$, we derive π_{k+1}, build M_{k+1} out of $\mu_{k+1} = \mu(\pi_{k+1})$ and project $\hat{\mu}^*$ on M_{k+1} to obtain $\hat{\mu}_{k+1}$ which satisfies (5).

Hence

$$\frac{\|\hat{\mu}^* - \hat{\mu}_{k+1}\|_2}{\|\hat{\mu}^* - \hat{\mu}_k\|_2} \leq \frac{d}{\sqrt{d^2 + (1-\gamma)^2\varepsilon^2}} \tag{32}$$

$$\Rightarrow \|\hat{\mu}^* - \hat{\mu}_k\|_2 \leq \left(\frac{d}{\sqrt{d^2 + (1-\gamma)^2\varepsilon^2}}\right)^k \|\hat{\mu}^* - \hat{\mu}_0\|_2 \tag{33}$$

$$\Rightarrow \|\hat{\mu}^* - \hat{\mu}_k\|_2 \leq \left(\frac{d}{\sqrt{d^2 + (1-\gamma)^2\varepsilon^2}}\right)^k \frac{\sqrt{d}}{1-\gamma} \tag{34}$$

$$\Rightarrow \|\hat{\mu}^* - \hat{\mu}_k\|_2 \leq \left(\frac{\sqrt{d}}{\sqrt{d + (1-\gamma)^2\varepsilon^2}}\right)^k \frac{\sqrt{d}}{1-\gamma} \tag{35}$$

$$\tag{36}$$

34 : maximum distance in $M(\Pi) = [0, \frac{1}{1-\gamma}]^d$, maximum distance is $d/(1-\gamma)$ 35 : complete proof in Abbeel and Ng [1] supplementary material

Hence, we verify $\|\hat{\mu}^* - \hat{\mu}_k\|_2 \leq \varepsilon$ if

$$k \geq \log\left(\frac{\sqrt{d}}{(1-\gamma)\varepsilon}\right) \log\left(\frac{\sqrt{d+(1-\gamma)^2\varepsilon^2}}{\sqrt{d}}\right)^{-1} = \mathcal{O}\left(\frac{d}{(1-\gamma)^2\varepsilon^2} \log \frac{d}{(1-\gamma)\varepsilon}\right)$$

' which proves the algorithm terminates.

Let's now show that if the number of Monte Carlo sample m is high enough, this is still true for the optimal expected feature count $\mu^* = \mu(\pi^*)$.

We know that $\forall \mu \in M(\Pi),\ \ (1-\gamma)\mu_i \in [0,1]\ \forall i \in [\![1,d]\!]$

Hence, Hoeffding inequality guarantees that $\forall \tau > 0\ \ \mathbb{P}((1-\gamma)|\mu_i^* - \hat{\mu}_i| > \tau) \leq 2e^{-2\tau^2 m}$

It follows that

$$\begin{aligned}
\mathbb{P}((1-\gamma)\|\mu^* - \hat{\mu}^*\|_\infty > \tau) &= \mathbb{P}\left(\bigcup_{i=1}^{d} \{(1-\gamma)|\mu_i^* - \hat{\mu}_i| > \tau\}\right) \\
&\leq \sum_{i=1}^{d} \mathbb{P}((1-\gamma)|\mu_i^* - \hat{\mu}_i| > \tau) \\
&\leq 2de^{-2\tau^2 m} \\
\Rightarrow \mathbb{P}((1-\gamma)\|\mu^* - \hat{\mu}^*\|_\infty \leq \tau) &\geq 1 - 2de^{-2\tau^2 m}
\end{aligned}$$

If we choose $\tau = \frac{(1-\gamma)\varepsilon}{2\sqrt{d}}$, as $\|\cdot\|_2 \leq \sqrt{d}\|\cdot\|_\infty$ for d-dimensional spaces, it comes $\mathbb{P}(\|\mu^* - \hat{\mu}^*\|_2 \leq \frac{\varepsilon}{2}) \geq 1 - 2de^{-2\tau^2 m}$.

Hence, by setting $m \geq \frac{2d}{(1-\gamma)^2\varepsilon^2} \log \frac{2d}{\delta}$ we get

$$\mathbb{P}(\|\mu^* - \hat{\mu}^*\|_2 \leq \frac{\varepsilon}{2}) \geq 1 - \delta \tag{37}$$

Finally, using the previous terminaison result, after a sufficient number of iterations K $\|\hat{\mu}^* - \hat{\mu}_K\|_2 \leq \frac{\varepsilon}{2}$ and we verify :

$$\|\mu^* - \hat{\mu}_K\|_2 \leq \|\mu^* - \hat{\mu}^*\|_2 + \|\hat{\mu}^* - \hat{\mu}_K\|_2 \leq \varepsilon$$

Hence with Cauchy Schwarz and as $\|\theta^*\|_2 \leq \|\theta^*\|_1 \leq 1$, we have with probability $1-\delta$:

$$|V^{\pi^*}(s_0) - V^{\pi_K}(s_0)| = |\theta^{*T}(\mu^* - \hat{\mu}_K)| \leq \|\theta^*\|_2 \|\mu^* - \hat{\mu}_K\|_2 \leq \varepsilon$$

$\square$