# Unbiased Efficient Feature Counts for Inverse RL

**Gerard Donahue**
University of New Hampshire
gerard.donahue@unh.edu

**Brendan Crowe**
University of New Hampshire
brendan.crowe@unh.edu

**Dr. Marek Petrik**
University of New Hampshire
mpetrik@cs.unh.edu

**Daniel Brown**
UC Berkeley
dsbrown@berkeley.edu

**Soheil Gharatappeh**
University of New Hampshire
soheil.gharatappeh@unh.edu

## Abstract

Feature counts play a crucial role when computing good reward weights in inverse reinforcement learning. Despite their importance, little work has focused on developing better methods for estimating feature counts. In this work, we propose a new method for estimating feature counts for scenarios with a small number of long demonstrations. Most existing algorithms perform poorly in this scenario. In particular, we propose two new algorithms, E-DLS and E-SLS, which can efficiently use a small number of long demonstrations to estimate feature counts. We show that E-SLS estimates are unbiased, which is the first such estimation algorithm. Our experimental results on benchmark problems demonstrate better learned reward weights when feature counts are estimated with E-DLS and E-SLS compared to other popular methods.

## 1 Introduction

In imitation learning, an agent aims to learn desirable behavior by observing demonstrations from an expert that consist of sequences of states (and sometimes, corresponding actions). A key challenge in IRL is to generalize the expert's behavior to states for which no expert action has been observed. A popular and efficient generalization method is to assume that the unknown rewards can be represented as a linear combination of a given set of features [2, 9]. Using this linearity assumption, many popular IRL algorithms are based on feature count matching, where the goal is to construct a policy that has feature counts as similar as possible to the empirical feature counts observed in the demonstrations [1, 2, 9, 10]. If the estimated feature counts are inaccurate, it is likely that the learned policy will not match the policy of the expert. To compute the feature counts, most prior work estimates the empirical discounted feature counts [6]. This approach may be sufficient when the objective tasks are small or there is an abundance of data. However, when the tasks are long and only a few demonstrations are available, existing methods are inefficient due to the discount factor diminishing the value of observations later in the episodes. In this paper we look to formulate new methods of estimating feature counts that are more robust to the limitations of discounting long episodes.

Related prior work that shares our goal of estimating feature counts from limited demonstrations is *Least Squares Temporal Difference Learning for μ* (LSTD-μ) [4]. LSTD-μ considers the well-known LSTD for value function approximation [8] and adapts it to estimating feature counts. LSTD-μ is specifically effective when there are a few long demonstrations, making it a preferred method of feature count estimation when obtaining expert samples is expensive. The algorithm is based on temporal difference (TD) methodologies and the linear least squares algorithm. TD uses bootstrapping, which does not provide guarantees for finite-sample generality, and may experience a significant amount of bias if given a set of demonstrations with less diversity in state visitations. Also, LSTD-μ

Preprint. Under review.

requires an additional set of feature-count features, which may be difficult to construct. If designed improperly, these additional features can introduce significant error into the process.

As the main contribution of this paper, we introduce two new methods for estimating feature counts when data is limited and the episodes are long. The first method, which we call *Estimation by Deterministic Length Segments* (E-DLS), produces a more efficient feature count estimation in scenarios with a small number of long demonstrations compared to existing methods [1]; which are inefficient for long demonstrations because discounting makes observations in the future less important. The main idea of E-DLS is to divide up expert episodes into *segments* beginning at initial starting states and continuing until the end of the episode. This ensures that observations later in the episode have sufficient presence in the numerical computation of the estimation.

Building on E-DLS, our second method, *Estimation by Stochastic Length Segments* (E-SLS), preserves the efficiency of E-DLS but also guarantees that the feature count estimates are unbiased. A common reason for the bias of existing methods [1] is that the demonstrated episodes are of a fixed length. Clipping the episodes such that their length is deterministic can introduce significant bias. The main idea of E-SLS is to (1) compute the expert occupancy frequency without a discount factor, and (2) use the notion of probabilistic termination when determining the length of segments. While each one of these modifications alone introduces potential bias in the expert feature count estimation, we tune the methods to extract impressive results in Section 5 and show in Section 4 that E-SLS is an unbiased estimator for feature counts.

## 2 Preliminaries and Framework

We first summarize the notation in this paper. Random variables and matrices are denoted with capital letters. Vectors, such as $\boldsymbol{x}$, are types using bold font, and their elements, such as $x_i$, are typeset regularly. We use $\Delta^S$ to denote the probability simplex over $\mathcal{S}$.

We model the problem as a discounted infinite-horizon Markov Decision Process (MDP) [7]. An MDP is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \alpha)$ comprising a finite set of states, $\mathcal{S} = \{1, \ldots, S\}$, a finite set of actions $\mathcal{A} = \{1, \ldots, A\}$, transition probabilities, $P : \mathcal{S} \times \mathcal{A} \to \Delta^S$, rewards, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and initial state distribution $\boldsymbol{\alpha} \in \Delta^S$. We define $\mathcal{S}_\alpha = \{s \in \mathcal{S} : \alpha_s > 0\}$ as the set of states with positive initial probability. A standard assumption in IRL is that rewards can be expressed as a linear combination of features $\boldsymbol{\phi} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^k$ where $k$ is the number of features. Hence, $r(s, a) = \boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(s, a)$ for each $s \in \mathcal{S}, a \in \mathcal{A}$ and some unknown weights $\boldsymbol{w} \in \mathbb{R}^k$.

Occupancy frequencies play an important role in many IRL algorithms. We use $u^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ to denote the state-action occupancy frequency for each policy $\pi$. In MDPs with large state and action spaces, computing the exact occupancy frequencies is difficult. Instead, one can compute *feature counts* $\boldsymbol{\mu}^\pi \in \mathbb{R}^k$ [3] to generalize occupancy frequencies to MDPs with large state and action spaces. Feature counts are defined as:

$$\boldsymbol{\mu}^\pi = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} \cdot \boldsymbol{\phi}(S_t, A_t) \mid S_1 \sim \alpha, A_t \sim \pi(S_t)\right] . \tag{1}$$

Feature counts are a good substitute for occupancy frequencies when one assumes that rewards can be expressed in terms of the features $\boldsymbol{\phi}$. This is because the return $\rho^\pi \in \mathbb{R}$ of a policy $\pi$ satisfies that

$$\rho^\pi = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} u^\pi(s, a) \cdot r(s, a) = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \sum_{j=1}^{k} u^\pi(s, a) \cdot \phi_j(s, a) \cdot w_j = \sum_{j=1}^{k} \mu_j^\pi \cdot w_j = \boldsymbol{w}^\mathsf{T} \boldsymbol{\mu}^\pi .$$

A crucial aspect of IRL is to learn to mimic an unknown policy $\pi_E : \mathcal{S} \to \mathcal{A}$ from a fixed set of expert observations $\mathcal{D}_E = (\tau_1, \tau_2, \ldots, \tau_L)$ where each episode, $\tau_i = (\langle \hat{s}_{i,1}, \hat{a}_{i,1} \rangle, \ldots, \langle \hat{s}_{i,|\tau_i|-1}, \hat{a}_{i,|\tau_i|-1} \rangle)$, is a finite set of sequential state-action pairs such that $\hat{a}_{i,t} = \pi_E(\hat{s}_{i,t})$. Given the data described above, the objective in many classical IRL methods has been to use feature count matching [1, 2, 9],

$$\min_{\pi \in \Pi} \|\boldsymbol{\mu}^\pi - \boldsymbol{\mu}^E\| + \psi(\pi), \tag{2}$$

where $\boldsymbol{\mu}^E = \boldsymbol{\mu}^{\pi_E}$, $\|\cdot\|$ is some vector norm, and $\psi : \Pi \to \mathbb{R}$ is some regularization function. Clearly, the precise value of $\boldsymbol{\mu}^E$ is unknown and must be estimated from $\mathcal{D}_E$. The *standard* method for estimating $\tilde{\mu}^E \approx \mu^E$ is to compute

$$\tilde{\boldsymbol{\mu}}^E \;=\; \frac{1}{|\mathcal{D}_E|} \sum_{\tau \in \mathcal{D}_E} \sum_{t=1}^{|\tau|} \gamma^{t-1} \cdot \boldsymbol{\phi}(\hat{s}_{\tau,t}, \hat{a}_{\tau,t}) \,. \tag{3}$$

Recall that $\hat{s}_t$ and $\hat{a}_t$ are the observed state and action at time-step $t$ in episode $\tau$.

A significant limitation of the standard estimation in (3), is that it makes a single sequential pass through each episode. As a result of discounting, later terms in the episode give little to negligible impact on the value of $\tilde{\boldsymbol{\mu}}^E$. In the remainder of the paper, we propose two algorithms that alleviate this weakness.

## 3 E-DLS: Efficient Feature Counts

This section introduces our first algorithm—*Estimation by Deterministic Length Segments* (E-DLS). E-DLS computes improved feature count estimates $\tilde{\boldsymbol{\mu}}^E$. Its main idea is straightforward. We take each given expert episode and split it into multiple new, but shorted, episodes which we call segments. We take care to create new segments only that begin with states that are known to be initial states and extend each segment until the end of the original episode from which they are created.

Next, we describe E-DLS formally. To enhance clarity, we consider only demonstrations $\mathcal{D}_E = \{\hat{\tau}\}$ consisting of a single long episode $\hat{\tau}$; however the algorithm easily extends to demonstration sets consisting of multiple episodes. We also omit $\hat{\tau}$ from the notation in this section when there is no ambiguity. Also shown algorithmically in *Algorithm 1*, the E-DLS estimate of the feature counts is computed as

$$\tilde{\boldsymbol{\mu}}^E \;=\; \sum_{j \in \mathcal{S}_\alpha} \sum_{i=1}^{\hat{N}(j)} \sum_{t=\hat{f}(j,i)}^{|\tau|} \frac{\alpha_j \cdot \gamma^{t-\hat{f}(j,i)}}{\hat{N}(j)} \cdot \boldsymbol{\phi}\left(\hat{s}_t, \hat{a}_t\right) \,. \tag{4}$$

Recall that $\mathcal{S}_\alpha$ is the set of states with positive initial probabilities. The functions $\hat{N}(j)$ and $\hat{f}(j,k)$ represent the number of occurrences and the index of the $k$-th occurrence of the state $j$ in the episode $\hat{\tau}$ respectively. The functions are formally defined as $\hat{f}(j,0) = 0$ and

$$\hat{f}(j,i) \;=\; \min\{l \mid |\hat{\tau}| \geq l > \hat{f}(j,i-1), \hat{s}_l = j\}, \qquad \hat{N}(j) \;=\; \max\{l \mid \hat{f}(j,l) \leq |\hat{\tau}|\} \,.$$

It is important to note that (4) weighs each segment by the initial probability, $\alpha_j$, of the state that starts the segment. This additional weight ensures that segments with differing initial states will be counted with correct proportions. We recognize that our algorithm results in different summations overlapping certain portions of $\hat{\tau}$, which may, in turn, result in some additional bias in the estimation. However, we found that when we extend all segments until the end of the episode we observe the most impressive results. This may be due to overlapping segments contributing additional value toward the estimation.

## 4 E-SLS: Unbiased Efficient Feature Counts

In this section, we refine the E-DLS algorithm in order to get a statistically *unbiased* estimate of feature counts with a new algorithm called *Estimation by Stochastic Length Segments* (E-SLS). E-DLS algorithm is biased because of the deterministic nature of segment lengths. We eliminate this bias by allowing segment lengths to fit a distribution based on *probabilistic termination*. This approach is motivated by the equivalence between the discounting by $\gamma$ and a total return undiscounted problem with $1-\gamma$ probability of termination in each step [7].

For E-SLS, we use the same scenario that we used for E-DLS, such that we are assuming access to only one longer episode, $\hat{\tau}$. E-SLS builds on E-DLS but segments do not continue until the end of the episode. The length $C_i$ of the $i$-th segment is randomly sampled from a *geometric distribution* with success probability $(1-\gamma)$. Intuitively, this means that each segment terminates with the probability of termination being $(1-\gamma)$. E-SLS proceeds as follows. Sample $C_i \sim \mathrm{Geom}(1-\gamma)$ for sufficiently

| *Error Norm* | *x = 1 ($L_1$)* | *x = 2 ($L_2$)* |
|---|---|---|
| Standard | $4.91 \pm 2.11$ | $6.41 \pm 2.09$ |
| LSTD-µ | $2.94 \pm 0.00$ | $4.34 \pm 0.00$ |
| E-DLS | $\mathbf{1.08 \pm 0.02}$ | $\mathbf{2.08 \pm 0.01}$ |
| E-SLS | $\mathbf{1.17 \pm 0.00}$ | $\mathbf{2.29 \pm 0.00}$ |

Table 1: Feature count estimation error $\|\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} - \boldsymbol{\mu}^{\boldsymbol{E}}\|_x$

| *Method* | *MM-IRL* | *LPAL* |
|---|---|---|
| Standard | $2.508 \pm 0.522$ | $2.297 \pm 0.756$ |
| LSTD-µ | $0.772 \pm 0.000$ | $0.778 \pm 0.000$ |
| E-DLS | $\mathbf{3.166 \pm 0.000}$ | $\mathbf{3.230 \pm 0.000}$ |
| E-SLS | $\mathbf{3.051 \pm 0.001}$ | $\mathbf{3.230 \pm 0.000}$ |

Table 2: Policy return after using $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ to estimate an expert reward function with IRL

many $i = 1, 2, \ldots$. Here, $\mathrm{Geom}$ is the geometric distribution. The feature counts are then estimated according to

$$\tilde{\boldsymbol{\mu}}^E = \sum_{j \in \mathcal{S}_\alpha} \sum_{i=1}^{\hat{N}(j)} \sum_{t=\hat{f}(j,i)}^{\hat{f}(j,i)+C_i} \frac{\alpha_j}{\hat{N}(j)} \cdot \boldsymbol{\phi}\left(\hat{s}_t, \hat{a}_t\right) \ . \tag{5}$$

The functions $\hat{N}(j)$ and $\hat{f}(j,i)$ represent the number of occurrences and the index of the $i$-th occurrence of the state $j$ in the episode $\hat{\tau}$ respectively. These functions are defined the same as they were in Section 3.

An essential difference between (5) and (4) is that (5) lacks the multiplicative discount factor component. The discount rate is instead handled implicitly using the stochastic episode length. The seemingly minor modification of E-DLS to use a stochastic length sequence makes the algorithm unbiased. We state this property formally next, and it is proven in Appendix C.

**Theorem 1.** *E-SLS algorithm* (5) *computes an unbiased estimator of feature counts:* $\mathbb{E}\left[\, \tilde{\boldsymbol{\mu}}^E \,\right] = \boldsymbol{\mu}^E$ *assuming that* $\hat{\tau}$ *is sufficiently long.*

To the best of our knowledge, E-SLS is the first unbiased method of estimating occupancy frequencies and feature counts.

## 5 Empirical Results

In this section, we present experimental results on a benchmark problem to indicate the experimental promise of E-DLS and E-SLS. This benchmark problem is called the *River Swimmer*.

For the River Swimmer problem, we use 10 states. The actions for this MDP are simple, you can either swim forward or swim backward. The initial state probability for this MDP gives a 50% chance of starting in state 0 (beginning of the river) and a 50% chance of starting in state 9 (the end of the river). Each state in the River Swimmer MDP represents progress as the agent looks to swim up the river. As such, the transition probabilities for this MDP tend to force the agent backwards as the current would suggest in a real-life river. The rewards for this MDP is unknown to the IRL agent, however the expert acts under the true rewards in a simulated means of extracting one long episode of river swimming. The true optimal policy is to always swim forward, as the rewards at higher states are greater.

The results for the River Swimmer experiment are shown in Table 1 and Table 2. The rows of both tables represent the estimation method of estimating Feature Counts, while the columns represent the metric we use after using that estimation method. We ran 200 individual experiments where long episodes of length 3000 were used. Each estimation method was used on each episode, and a collection of metrics were used to evaluate their effect on performance. For Table 1, we use both $\mathrm{L}_1$ and $\mathrm{L}_2$ error norms to show different error measurements. $L_1$ is computed by $\|\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} - \boldsymbol{\mu}^{\boldsymbol{\pi}}\|_1$ and $L_2$ is computed by $\|\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} - \boldsymbol{\mu}^{\boldsymbol{\pi}}\|_2$. While estimation error is important, it does not necessarily indicate that we will achieve better policy return after using our estimated $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ in IRL algorithms. Because of this, we show in Table 2 the return accomplished after value iteration when we use IRL methods to estimate a reward function. For IRL methods we use *Max-Margin* (MM-IRL) [1] and *Linear Programming Apprenticeship Learning* (LPAL) [9]. E-DLS, E-SLS, LSTD-µ, and Standard are the $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ estimation method that we used for both Table 1, and Table 2.

As noted in Section 1, LSTD-µ looks to tackle the same scenario as E-DLS and E-SLS; when our agent is given a singular long expert episode. Shown in Table 1, LSTD-µ predicts a $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ that is closer to the true expert feature counts, $\boldsymbol{\mu}^{\boldsymbol{E}}$ than the Standard. This is expected, as the goal of LSTD-μ is to outperform previous methods when given longer episodes. However, it is noted that LSTD-μ performs the worst when using subsequent IRL algorithms. We theorize that this is due to the temporal difference methodologies and the effect that bootstrapping has on generalizing the feature count estimation for states not visited. Further analysis of LSTD-μ's inefficiency with respect to policy return is needed to gain better theory on these results.

The emboldened results in Table 1 highlight the estimation accuracy of our new methods in practice. The $L_1$ error for E-DLS and E-SLS is roughly a fourth of the Standard method's, and a third of LSTD-μ's. These results indicate that the element-wise error of E-DLS and E-SLS estimations are significantly lower than previous methods. The $L_2$ error for E-DLS and E-SLS is roughly a third of the Standard method's, and a half of LSTD-μ's. These error results indicate that the estimated feature count vector is much closer to the true feature count vector in direction and magnitude.

The emboldened results in Table 2 highlight the practical promise of our new methods with respect to policy return after obtaining a learned policy with IRL algorithms. When given long episodes, it is clear that segmenting the episodes based on starting states with E-DLS and E-SLS allows for more informative estimates in the IRL algorithm. It is interesting to note that, while E-SLS is an unbiased estimation method, E-DLS performs better with MM-IRL. For LPAL, E-DLS and E-SLS result in the same policy return. Further analysis is needed to determine the reason for these results. It is also important to note that LSTD-μ, E-DLS, and E-SLS, all produce results with very low variance over the 200 experiments. The Standard method has relatively high variance in general. A potential reason for this is that the Standard method is dependent on the behavior earlier in the episode because the discount factor causes diminishing value in later behaviors, which causes the estimated feature counts to vary with the stochastic behavior earlier in the episodes. Segmentation in E-DLS and E-SLS allows the algorithm to refresh the discount factor and have better insight into behaviors that follow starting states that appear throughout the long episode, and makes it less dependent on behaviors observed earlier in the episode.

## 6 Conclusion

We proposed two new methods for estimating feature counts. Feature count estimation is an important component of many IRL algorithms. On the theoretical side, we propose the first unbiased method for estimating feature counts. Our limited empirical results indicate the promise of the proposed methods. The future work should focus on deepening the theoretical understanding of these new methods as well as on more comprehensive empirical results.

## 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, page 1, New York, NY, USA, 2004. Association for Computing Machinery.

[2] Jonathan Ho and Stefan Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, pages 7461–7472, 2016. ISSN: 10495258.

[3] Jessie Huang, Fa Wu, Doina Precup, and Yang Cai. Learning safe policies with expert guidance. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 9105–9114. Curran Associates, Inc., 2018.

[4] Edouard Klein, Matthieu Geist, and Olivier Pietquin. Batch, Off-Policy and Model-Free Apprenticeship Learning. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Scott Sanner, and Marcus Hutter, editors, *Recent Advances in Reinforcement Learning*, volume 7188, pages 285–296. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[5] Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5:1–16, 2016.

[6] Bilal Piot, Matthieu Geist, and Olivier Pietquin. Bridging the gap between imitation learning and inverse reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 28(8):1814–1826, August 2017. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[7] Martin L Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. Wiley-Interscience, 2005.

[8] R S Sutton and A G Barto. *Reinforcement learning: an introduction*. The MIT Press, 2 edition, 2018. arXiv: 1603.02199 ISSN: 1045-9227.

[9] Umar Syed, Michael Bowling, and Robert E. Schapire. Apprenticeship learning using linear programming. *International Conference on Machine Learning (ICML)*, pages 1032–1039, 2008. ISBN: 9781605582054.

[10] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum Entropy Inverse Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, 2008.

| Discounted weight | $1$ | $\gamma$ | $\gamma^2$ | $\gamma^3$ | $\gamma^4$ | $\gamma^5$ | $\gamma^6$ | $\gamma^7$ | $\gamma^8$ | $\gamma^9$ | $\dots$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Demonstration ($\hat{\tau}$) | $s_0$ | $s_5$ | $s_3$ | $s_5$ | $s_3$ | $s_1$ | $s_2$ | $s_5$ | $s_0$ | $s_4$ | $\dots$ |
| Segment 1 | $s_0$ | $s_5$ | $s_3$ | $s_5$ | $s_3$ | $s_1$ | $s_2$ | $s_5$ | $s_0$ | $s_4$ | $\dots$ |
| Segment 2 | $s_1$ | $s_2$ | $s_5$ | $s_0$ | $s_4$ | $\dots$ | | | | | |
| Segment 3 | $s_0$ | $s_4$ | $\dots$ | | | | | | | | |

Table 3: An example of a segmented demonstration.

## A Example

Consider the following simple scenario to illustrate how E-DLS works. We have an IRL system with initial states $\mathcal{S}_\alpha = \{s_0, s_1\}$ and given an expert episode $\hat{\tau}$ depicted in the top row of Table 3. In order to estimate $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ we traverse through $\hat{\tau}$ sequentially, and initialize new segments as shown in Table 3. Each segment begins with $s_0$ or $s_1$, one of the initial states. Creating segments allows us to restart the discount factor when observing the start state, which will give the starting states a higher valued estimation upon each re-occurrence. We argue that this gives value to the estimation of $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ by exposing it to segments that show varying trajectories following a starting state.

## B Algorithms

```
Algorithm 1: E-DLS
Data: $\hat{\tau}, \gamma, \boldsymbol{\alpha}, \phi$
Result: $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ //Estimated feature counts given the expert data
$\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} \leftarrow \mathbf{0}$; //Vector of zeros existing in $\mathbb{R}^k$
for $t = 1 \dots |\hat{\tau}|$ do
    counts $\leftarrow 0$;
    for $j = 1 \dots t$ do
        counts $\leftarrow$ counts $+ \alpha_{\hat{s}_j} \cdot \gamma^{t-j}$;
    end
    $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} \leftarrow \phi(\hat{s}_t, \hat{a}_t) \cdot$ counts;
end
return $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$;
```

```
Algorithm 2: E-SLS
Data: $\hat{\tau}, \gamma, \boldsymbol{\alpha}, \phi$
Result: $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$ //Expected feature counts given the expert data
$\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} \leftarrow \mathbf{0}$; //Vector of zeros existing in $\mathbb{R}^k$
$t = 1$;
while $t \leq |\hat{\tau}|$ do
    if $\alpha_{\hat{s}_t} > 0$ then
        $c \leftarrow \mathrm{Geom}(1 - \gamma)$;
        for $j = t \dots t + c$ do
            $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} \leftarrow \tilde{\boldsymbol{\mu}}^{\boldsymbol{E}} + \alpha_{\hat{s}_t} \cdot \phi(\hat{s}_j, \hat{a}_j)$;
        end
    end
    $t \leftarrow t + 1$;
end
return $\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}$;
```

## C Proofs

*Proof of Theorem 1.* Before commencing the proof, we state two well-known identities that we will need to derive our result. First, one can easily show by algebraic manipulation that:

$$\sum_{i=1}^{\infty}\sum_{j=i}^{\infty} f(i,j) = \sum_{1\le i\le j<\infty} f(i,j) = \sum_{j=1}^{\infty}\sum_{i=j}^{\infty} f(i,j)\,. \tag{6}$$

Second, using the analytical expressions for the geometric sum and the series, we get from algebraic manipulation that

$$\begin{aligned}\sum_{l=t}^{\infty}\gamma^l(1-\gamma) &= \left(\sum_{l=0}^{\infty}\gamma^l - \sum_{l=0}^{t-1}\gamma^l\right)(1-\gamma) = 1-\left(\sum_{l=0}^{t-1}\gamma^l\right)(1-\gamma)\\ &= 1-\left(\frac{1-\gamma^t}{1-\gamma}\right)(1-\gamma) = \gamma^t\,.\end{aligned} \tag{7}$$

Here we prove that $\mathbb{E}\left[\,\tilde{\boldsymbol{\mu}}^{\boldsymbol{E}}\,\right] = \boldsymbol{\mu}^{\boldsymbol{E}}$. Given an expert episode, $\hat{\tau}$, and considering (5), we get that:

$$\begin{aligned}
\mathbb{E}\left[\,\tilde{\boldsymbol{\mu}}\,\right] &= \mathbb{E}\left[\sum_{j\in\mathcal{S}_\alpha}\sum_{i=1}^{\hat{N}(j)}\sum_{t=\hat{f}(j,i)}^{\hat{f}(j,i)+C_i}\frac{\alpha_j}{\hat{N}(j)}\cdot\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_{\hat{f}(j,i)}=j\right]\\
&= \sum_{j\in\mathcal{S}_\alpha}\sum_{i=1}^{\hat{N}(j)}\mathbb{E}\left[\sum_{t=\hat{f}(j,i)}^{\hat{f}(j,i)+C_i}\frac{\alpha_j}{\hat{N}(j)}\cdot\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_{\hat{f}(j,i)}=j\right] && \text{Move expectation inwards}\\
&= \sum_{j\in\mathcal{S}_\alpha}\sum_{i=1}^{\hat{N}(j)}\frac{\alpha_j}{\hat{N}(j)}\mathbb{E}\left[\sum_{t=1}^{C_i}\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_{\hat{f}(j,i)}=j\right] && \text{Take out constants}\\
&= \sum_{j\in\mathcal{S}_\alpha}\alpha_j\cdot\mathbb{E}\left[\sum_{t=1}^{C_i}\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_{\hat{f}(j,i)}=j\right] && \text{Remove } \hat{N}(j)\\
&= \sum_{j\in\mathcal{S}_\alpha}\alpha_j\cdot\sum_{l=1}^{\infty}P(C_i=l)\cdot\sum_{t=1}^{l}\mathbb{E}\left[\,\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_1=j\,\right] && \text{Expand expectation based on } \mathrm{Geom}(1-\gamma)\\
&= \sum_{j\in\mathcal{S}_\alpha}\alpha_j\cdot\sum_{t=1}^{\infty}\mathbb{E}\left[\,\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_1=j\,\right]\cdot\sum_{l=t}^{\infty}P(C_i=l) && \text{Re-arrange summation using (6)}\\
&= \sum_{j\in\mathcal{S}_\alpha}\alpha_j\cdot\sum_{t=1}^{\infty}\mathbb{E}\left[\,\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_1=j\,\right]\cdot\sum_{l=t}^{\infty}\gamma^{l-1}\cdot(1-\gamma) && \text{Substitute } P(C_i=l) \text{ from } \mathrm{Geom}(1-\gamma)\\
&= \sum_{j\in\mathcal{S}_\alpha}\alpha_j\cdot\sum_{t=1}^{\infty}\mathbb{E}\left[\,\boldsymbol{\phi}\left(\hat{s}_t,\hat{a}_t\right)\ \middle|\ \hat{s}_1=j\,\right]\cdot\gamma^{t-1} && \text{From (7)}
\end{aligned}$$

This proves the desired equality. □