Option Discovery in the Absence of Rewards with Manifold Analysis

Amitay Bar [1] **Ronen Talmon** [1] **Ron Meir** [1]

Abstract

Options have been shown to be an effective tool in reinforcement learning, facilitating improved exploration and learning. In this paper, we present an approach based on spectral graph theory and derive an algorithm that systematically discovers options without access to a specific reward or task assignment. As opposed to the common practice used in previous methods, our algorithm makes full use of the spectrum of the graph Laplacian. Incorporating modes associated with higher graph frequencies unravels domain subtleties, which are shown to be useful for option discovery. Using geometric and manifold-based analysis, we present a theoretical justification for the algorithm. In addition, we showcase its performance in several domains, demonstrating clear improvements compared to competing methods.

1. Introduction

Reinforcement learning (RL) has attracted much attention in recent years thanks to its success in solving a broad range of challenging tasks. Options (a.k.a. skills) play an important role in RL (Sutton et al., 1999) and have opened the door to a series of studies demonstrating improvement in both learning and exploration (Vezhnevets et al., 2017; Nachum et al., 2018; Eysenbach et al., 2019; Bellemare et al., 2016; Tang et al., 2017; Mannor et al., 2004; Menache et al., 2002). One important class of options consists of options that are not associated with any specific task and are acquired without receiving any reward. Such generic options often lead to efficient learning in various tasks that are not known a-priori, e.g., Eysenbach et al. (2019).

An effective approach to build such options is based on spectral graph theory, assuming a finite state domain in which each state is regarded as a node of a graph, and the graph edges represent the states' connectivity. In Mahadevan & Maggioni (2007), such an approach led to the introduction of proto-value functions (PVFs), which are the eigenvectors of the graph Laplacian (Chung & Graham, 1997). It was shown that the PVFs establish an efficient representation of the domain. Recently, these PVFs were used for options representation in Machado et al. (2017; 2018). There, eigenoptions were introduced by considering only the dominant eigenvectors (PVFs), where each eigenoption is formed based on a single eigenvector. On the one hand, option discovery with a graph-based representation is a powerful combination, since it facilitates options that are not task or reward-specific, yet it naturally incorporates the geometry of the domain. On the other hand, this method lacks theoretical justification. For example, defining each (eigen)option based only on a single eigenvector, or considering only the dominant eigenvectors while omitting the rest, leave room for improvement.

In this paper, we present a new scheme for defining options, relying on all eigenvectors of the graph Laplacian. More concretely, we form a score function built from the eigenvectors, from which options can be systematically derived. Since the agent acts without receiving reward, it is only natural to discover and analyze the options considering the geometry of the domain. For analysis purposes, we model the domain as a manifold and consequently the graph as a discrete approximation of the manifold, allowing us to incorporate concepts and results from manifold learning, such as the diffusion distance (Coifman & Lafon, 2006a). We show that our options lead to improved performance both in learning and exploration compared to the eigenoptions as well as other option discovery schemes.

Our main contributions are as follows. First, we present a new approach to principled option discovery with a theoretical foundation based on geometric and manifold analysis. Second, this analysis includes novel results in manifold learning involving two key components: the stationary distribution of a random walk on a graph and the diffusion distance. To obtain these results, we employ a new concept in manifold learning, in which the entire spectra of the underlying graph is considered rather than only its leading components. Third, we propose an algorithm for option discovery, applicable in high-dimensional deterministic domains. We empirically demonstrate that the learning performance obtained by our options outperforms competing options on three small-scale domains. In addition, we show extensions

[*]Equal contribution [1]Viterbi Faculty of Electrical Engineering, Technion, Israel Institute of Technology . Correspondence to: Amitay Bar <amitayb@campus.technion.ac.il>.

to stochastic domains as well as to partially known domains.

2. Background

2.1. RL and Options

We use the Markov decision process (MDP) framework to formulate the RL problem (Puterman, 2014). An MDP is a 5-tuple $\langle \mathbb{S}, \mathbb{A}, p, r, \gamma \rangle$, where $\mathbb{S}$ is the set of states, $\mathbb{A}$ is the set of actions, p is the transition probability such that $p\left(s'|s,a\right)$ is the probability of moving from state s to state s' by taking an action a, $r(s,a,s')$ is the reward function and $\gamma \in [0,1)$ is a discount factor. Consider an agent operating sequentially so that at time step n it moves from state s_n to state s_{n+1}, receiving a reward $R_{n+1} = r(s_n, a, s_{n+1})$. Its goal is to learn a policy $\pi : \mathbb{S} \times \mathbb{A} \to [0,1]$ which maximizes the expected discounted return $G_n \triangleq \mathbb{E}_{\pi,p}[\sum_{k=0}^{\infty} \gamma^k R_{n+k+1} | s_n]$.

An option is a generalization of an action (also known as a skill or a sub-goal) (Sutton et al., 1999). Formally, an option o is the 3-tuple $\langle \mathbb{I}, \pi_o, \beta \rangle$ where $\mathbb{I}$ is an initiation set $\mathbb{I} \subseteq \mathbb{S}$ (the states at which the option can be invoked), $\pi_o : \mathbb{S} \times \mathbb{A} \to [0,1]$ is the policy of the option to be followed by the agent, and $\beta : \mathbb{S} \to [0,1]$ is the termination condition. By following an option o the agent chooses actions according to the policy of the option π_o until the option is terminated according to the termination condition β.

2.2. Diffusion Distance

The diffusion distance is a notion of distance between two points in a high-dimensional data set (Coifman & Lafon, 2006a), where the points are assumed to lie on a manifold. It is widely used in many data science applications, e.g., in Mahmoudi & Sapiro (2009); Bronstein et al. (2011); Lafon et al. (2006); Liu et al. (2009); Van Dijk et al. (2018), since it captures well the geometric structure of the data. While the formulation of diffusion distance is typically general, here we describe it directly in the MDP setting.

Consider a graph $G = (\mathbb{S}, \mathbb{E})$, where the finite set of states $\mathbb{S}$ is the node set and the edge set $\mathbb{E} \subset \mathbb{S} \times \mathbb{S}$ consists of all possible transitions between states. Define a random walk on the graph with transition probability matrix $\boldsymbol{W}$, defined by $\boldsymbol{W}_{ij} = p(s_{t+1} = i | s_t = j)$. Let $\boldsymbol{p}_t^{(l)}$ denote the vector of transition probabilities from state l to all states in t random walk steps defined by the lth column of $\boldsymbol{W}^t$. With the above preparation, the diffusion distance is defined by:

$$D_t\left(s, s'\right) \triangleq \|\boldsymbol{p}_t^{(s)} - \boldsymbol{p}_t^{(s')}\|,$$

where $\|\cdot\|$ is the L_2 norm. In contrast to the standard Euclidean distance, the diffusion distance does not depend solely on two individual points, namely, s and s', but takes into account the structure of the entire data sets. See a prototypical demonstration in the supplementary material (SM). Broadly, in short distances it is closely related to the geodesic distance (shortest path) (Portegies, 2016) and in long distances it demonstrates high robustness to noise and outliers (Coifman & Lafon, 2006a). For more details on the advantages of the diffusion distance and its efficient computation using the eigenvectors of the graph Laplacian see the SM.

3. Diffusion Options

In standard, mostly goal-oriented RL, one learns to map states to actions in order to achieve a desired task. In situations with uncertainty (e.g., model uncertainty, reward uncertainty, etc.) exploration is essential in order to reduce uncertainty, thereby improving future actions. Exploration often consists of aspects that are specific to a given task, and aspects that are generic to the domain. For example, in an environment with multiple rooms, one may wish to learn how to reach the door of each room, thereby facilitating learning in later situations where a specific task is given, say, reaching a specific room (or set of rooms). This may also be useful if additional rooms are later added. In both cases (task-based or task-free), options can greatly facilitate the speed of exploration by forming shortcuts (Eysenbach et al., 2019). In this work we present a manifold-based approach to developing generic options that can be later used across multiple task domains.

To encourage exploration, a useful set of options will lead the agent to distant regions, visiting states that the uninformed random walk will seldom lead to. To this end, we exploit the diffusion distance and show that the strength of diffusion distance in the realm of high dimensional data analysis enables us to devise structure-aware options that improve both learning and exploration.

3.1. Algorithm

Initially, we consider discrete and deterministic domains with a finite number of states, where the transitions between states are known to the agent. The proposed algorithm for systematic option discovery consists of two stages. The first stage involves graph construction. Let G be a graph whose node set is the finite set of states $\mathbb{S}$. Let $\boldsymbol{M} \in \mathbb{R}^{|\mathbb{S}| \times |\mathbb{S}|}$ be the symmetric adjacency matrix of the graph, prescribing the possible transitions between states, namely $\boldsymbol{M}_{s,s'} = 1$ if a transition from state s to state s' is possible, and $\boldsymbol{M}_{s,s'} = 0$ otherwise. Based on $\boldsymbol{M}$, define a non-symmetric lazy random walk matrix $\boldsymbol{W} \triangleq \frac{1}{2}(\boldsymbol{I} + \boldsymbol{M}\boldsymbol{D}^{-1})$, where the degree matrix $\boldsymbol{D}$ is a diagonal matrix whose diagonal elements equal the sum of rows of $\boldsymbol{M}$. Applying eigenvalue decomposition to $\boldsymbol{W}$ yields two sets of left and right eigenvectors, denoted by $\{\phi_i\}$ and $\{\tilde{\phi}_i\}$, respectively, and a set of real eigenvalues $\{\omega_i\}$. The sth component of ϕ_i is denoted by $\phi_i(s)$.

The second stage of the algorithm relies on the following function, defined on the set of states $s \in \mathbb{S}$,

$$f_t(s) \triangleq \| \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i \|^2, \quad (1)$$

where $t > 0$ is a scale parameter, representing the diffusion time. By construction, $f_t(s)$ consists of the full spectrum of $\boldsymbol{W}$, including both low and high frequencies, in contrast to common practice. As we show in Proposition 1, $f_t(s)$ is directly related to the average diffusion distance between state s and all other states, making it a promising candidate for an option discovery criterion, as discussed below.

After computing $f_t(s)$, the states at which it gets a local maximum are extracted. We term these states option goal states, and denote them by $\{s_o^{(i)}\}$, where the index i ranges between 1 and the number of local maxima. Each such state is associated with an option, which leads the agent from its current state to the option goal state. The options can start at any state ($\mathbb{I} = \mathbb{S}$), and terminate deterministically once reaching its option goal state, i.e. for option i, $\beta_i\left(s_o^{(i)}\right) = 1$, and $\beta_i(s) = 0 \, \forall s \neq s_o^{(i)}$. In other words, once the agent chooses to act according to an option, it moves to s_o via the shortest path from its current position. We note that the scale parameter t indirectly controls the number of options; since $0 < \omega_i \leq 1$ (Chung & Graham, 1997), the multiplication by ω_i^t in (1) makes $f_t(s)$ smoother as t increases, analogously to a low pass filter effect.

The proposed algorithm for option discovery appears in Algorithm 1. We term the discovered options *diffusion options* because they are built from the eigenvalue decomposition of a discrete diffusion process, i.e., the lazy random walk on the graph. In addition, in section 3.3, we show a tight relation to the diffusion distance. Algorithm 1 exhibits several advantages. First, the algorithm prescribes a systematic way to derive options which are not associated with any particular task or reward. Second, we empirically demonstrate the acceleration of the learning process and more efficient exploration in prototypical domains compared to competing methods for option discovery. Third, the computationally heavy part is performed only once and in advance. Fourth, the scale parameter t enables to control the number of options and facilitates multiscale option discovery.

We remark that the eigenvalue decomposition of $\boldsymbol{W}$ used for the construction of $f_t(s)$ is related to the eigenvalue decomposition of the normalized graph Laplacian $\boldsymbol{N}$, which traditionally forms the spectral decomposition of a graph. See the SM for details.

3.2. Partially Known Model

The exposition thus far focused on domains that are known a-priori. Suppose now that the considered set of states $\mathbb{S}$ is only a subset of the entire set of states, sampled from large or continuous domains. In the SM, we show that the extension of Algorithm 1 to unseen states $s \notin \mathbb{S}$ requires the extension of the eigenvectors ϕ and $\tilde{\phi}$, and the extension of the option policies.

Algorithm 1 Diffusion Options

Input: Adjacency matrix $\boldsymbol{M}$ and scale parameter $t > 0$
Output: K options with policies $\{\pi_o^{(i)}\}_{i=1}^K$

1: Compute the degree matrix $\boldsymbol{D}$ from $\boldsymbol{M}$
2: Compute the random walk matrix $\boldsymbol{W} = \frac{1}{2}(\boldsymbol{I} - \boldsymbol{M}\boldsymbol{D}^{-1})$
3: Apply EVD to $\boldsymbol{W}$ and obtain $\{\phi_i\}$, $\{\tilde{\phi}_i\}$ and $\{\omega_i\}$
4: Construct $f_t(s) = \| \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i \|^2$
5: Find the local maxima of $f_t(\cdot)$ – $\{s_o^{(i)}\}_{i=1}^K$
6: **for** $i \in \{1, \ldots, K\}$ **do**
7: Build an option with policy $\pi_o^{(i)}$ s.t. it leads to $s_o^{(i)}$
8: **end for**

For the purpose of these two extensions, we can take advantage of recent work. Wu et al. (2019) showed a scalable approach for estimating the eigenvectors of the Laplacian based only on a small subset of states. They showed results on continuous control navigation tasks. A different approach was proposed by Machado et al. (2018), which estimated the eigenvectors using deep successor representation. Recent advances in deep manifold learning could also be used (Chui & Mhaskar, 2018; Mishne et al., 2017), where the eigenvectors were extended based on an appropriate regularized objective function. For the extension of the option policy, (Shah & Xie, 2018) showed an approach based on the extension of the Q function.

3.3. Analysis

We start the analysis with our main result relating $f_t(s)$ to the diffusion distance. The proof is provided in the SM.

Proposition 1. *The function $f_t : \mathbb{S} \to \mathbb{R}$ defined as $f_t(s) \triangleq \| \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i \|^2$ is equal to the mean squared diffusion distance between state s and all other states, up to a constant independent on s, namely*

$$f_t(s) = \left\langle D_t^2(s, s') \right\rangle_{s' \in \mathbb{S}} + const, \quad (2)$$

where $\langle g(x) \rangle_{x \in \mathbb{X}}$ represents the average on $\mathbb{X}$:

$$\langle g(x) \rangle_{x \in \mathbb{X}} \triangleq \frac{1}{|\mathbb{X}|} \sum_{s \in \mathbb{X}} g(x).$$

An immediate consequence of Proposition 1 is that

$$\max_s f_t(s) = \max_s \left\langle D_t^2(s, s') \right\rangle_{s' \in \mathbb{S}},$$

implying that the option goal states, $\{s_o^{(i)}\}$, are the farthest states from all other states in terms of average squared diffu-

sion distance. Broadly, moving to such far states encourages exploration as the agent systematically travels through the largest number of states without, for example, the repetitions involved in the uninformed random walk. Additionally, by reaching different option goal states, the agent reaches different and distant regions of the domain, which also benefits exploration. The particular notion of diffusion distance efficiently captures the geometry of the domain and demonstrates important advantages over the Euclidean and even the geodesic distances. See the SM for an illustrative example. The averaging operation $\langle\cdot\rangle$ incorporates the fact that the options are not related to a specific task, and therefore, the start state, the goal state, and the states at which the options are invoked, are all unknown a-priori.

Empirically we will demonstrate that the diffusion distance is related to the domain difficulty (see section 4.4). The larger the average pairwise diffusion distance is, the more difficult the domain is. As a result, when the agent follows options leading to distant states in terms of the diffusion distance, in effect, it reduces the domain difficulty. In addition, we demonstrate that such goal states are typically "special" states such as corners of rooms or bottleneck states such as doors (see Fig. 1(h) and section 4).

Proposition 2 offers an alternative perspective on $f_t(s)$, relating it to the stationary distribution of the graph, denoted by $\boldsymbol{\pi}_0$. The proof is in the SM.

Proposition 2. *$f_t(s)$ can be recast as*

$$f_t(s) = \|\boldsymbol{p}_t^{(s)} - \boldsymbol{\pi}_0\|^2,$$

where $\boldsymbol{\pi}_0$ is the stationary distribution of the lazy random walk $\boldsymbol{W}$ on the graph G. In addition, $f_t(s)$ is bounded from above by

$$f_t(s) \leq \omega_2^{2t}\left(\frac{1}{\boldsymbol{\pi}_0(s)} - 1\right).$$

The first part of Proposition 2 relates $f_t(s)$ to the difference between the transition probability from state s and the stationary distribution. As t grows to infinity, the transition probability approaches the stationary distribution. For a fixed t, the states at which $f_t(s)$ gets a maximum value are the states that their transition probability differ the most from the stationary distribution.

States s for which $\pi_0(s)$ is small are states that are least visited by an agent following a standard random walk. Arguably, these are exactly the states the agent should visit, for example by following options, to improve exploration. Indeed, we observe that the upper bound in Proposition 2 implies that these states allow for large $f_t(s)$ values. We further discuss the relation between $f_t(s)$ and π_0 in a multi-dimensional grid domain in the SM.

Establishing the relation of $f_t(s)$ to the stationary distribution is important by itself because the stationary distribution is a central component in many applications and algorithms. Perhaps the most notable are PageRank (Page et al., 1999) and its variants (Kleinberg, 1999), where the purpose is to discover important web pages that are highly connected and therefore can be considered as network hubs. In the exploration-exploitation terminology, one could claim that PageRank favors exploitation by identifying central pages. Conversely, the diffusion options lead the agent toward states that are least connected (with small stationary distribution values), and therefore, they encourage exploration.

We end this section with a couple of remarks. First, the upper bound in Proposition 2 generalizes a known bound on the convergence of the transition probability, starting from node a in a graph, to the stationary distribution at node b (Spielman, 2018):

$$|p_t(b) - \pi_0(b)| \leq \sqrt{\frac{d(b)}{d(a)}}\omega_2^t,$$

where $d(a)$ and $d(b)$ are the degrees of nodes a and b, respectively.

Second, combining Proposition 1 and Proposition 2 relates the diffusion distance to the distance from the stationary distribution of a random walk. This relation may have consequences in a broader context, when either the diffusion distance or the stationary distribution are used.

3.4. Extension to Stochastic Domains

In the deterministic setting we considered thus far, we assumed that an action definitively leads the agent to a particular state, i.e., given an action a and a state s the probability $p(s'|s,a)$ is concentrated at a single state.

Alternatively, one could consider a setting, where the domain is stochastic, and its stochasticity introduces uncertainty and decouples the action from the transition, namely, $p(s'|s,a)$ can be supported on more than one state. As a result, the agent following a random walk experiences a different number of transitions between states. The corresponding transition probability matrix leads to a non-symmetric normalized graph Laplacian $\boldsymbol{N}$. This poses a challenge since the eigenvalue decomposition of $\boldsymbol{N}$ is not guaranteed to be real, and therefore, the construction of $f_t(s)$ in (1) needs a modification. Note that other settings could lead to an asymmetric Laplacian as well.

Here, we propose a remedy to support such cases. Our solution follows the work presented by Mhaskar (2018), which is based on the polar decomposition. Concretely, consider the polar decomposition of $\boldsymbol{N} = \boldsymbol{R}\boldsymbol{U}$, where $\boldsymbol{R}$ is a positive semi-definite matrix and $\boldsymbol{U}$ is a unitary matrix. Since $\boldsymbol{R}$ is uniquely determined, the spectral analysis applied to $\boldsymbol{N}$ in the deterministic case can be applied to $\boldsymbol{R}$ in a similar manner. As observed in Mhaskar (2018), there exist efficient

algorithms for computing $\boldsymbol{R}$, as in Nakatsukasa et al. (2010). Accordingly, the required modification applied to the option discovery in Algorithm 1 is minimal. After the computation of $\boldsymbol{N}$, its polar decomposition is computed. Then, the eigenvalue decomposition of the positive part $\boldsymbol{R}$ is used for the construction of $f_t(s)$. See the SM for the modified algorithm. In section 4.3, we demonstrate its performance.

4. Experimental Results

We demonstrate empirically that the diffusion options are generic and useful, allowing for improvement in both learning unknown tasks and in exploring domains efficiently. Particularly, using Q learning (Watkins & Dayan, 1992), we show that equipped with the diffusion options, which are computed in a reward-free domain, the agent is able to learn tasks that are unknown a-priori faster and to explore a domain more effectively. In addition, we demonstrate the relation between the diffusion options and the stationary distribution.

We focus on three domains: a Ring domain, which is the 2D manifold of the placement of a 2-joint robotic arm (Verma, 2008), a Maze domain (Wu et al., 2019), and a 4Rooms domain (Sutton et al., 1999). The set of actions are: left, right, up and down. In every domain, we pre-define a single start state and a set of goal states.

The agent performs several trials, where each trial is associated with a different goal state from the set of goal states. In each trial, the agent starts at the same start state and is assigned with the task of reaching the trial goal state. We implement Q learning (Watkins & Dayan, 1992) with $\alpha = 0.1$ and $\gamma = 0.9$ for 400 episodes, containing 100 steps each. The agent follows the Q function at states for which it exists, and otherwise chooses a primitive action or an option with equal probability. In case the agent does not reach the goal state after 100 steps, a default value of 101 is set for the number of steps.

Since options typically consist of multiple steps, for a fair comparison, we take them into account in the total steps count at each episode. Note that this might lead to terminating an option without reaching its option goal state in case the episode reaches 100 steps.

We compare the diffusion options with the eigenoptions presented by Machado et al. (2017). As a baseline, we also show results for a random walk consisting of only primitive actions without options. In the SM, we include comparison to random options as well.

We evaluate the performance using three objective measures. The first measure is the standard learning convergence. We compute the average number of steps to a goal over all learning trials (goal states), where each trial consists of 30 Monte Carlo iterations. The average number of steps is presented as a function of the learning episode. Second, we present the average number of visitations at each state during learning (over all episodes and goal states). Third, to evaluate the exploration efficiency, we compute the number of steps between every two states, as in Machado et al. (2017).

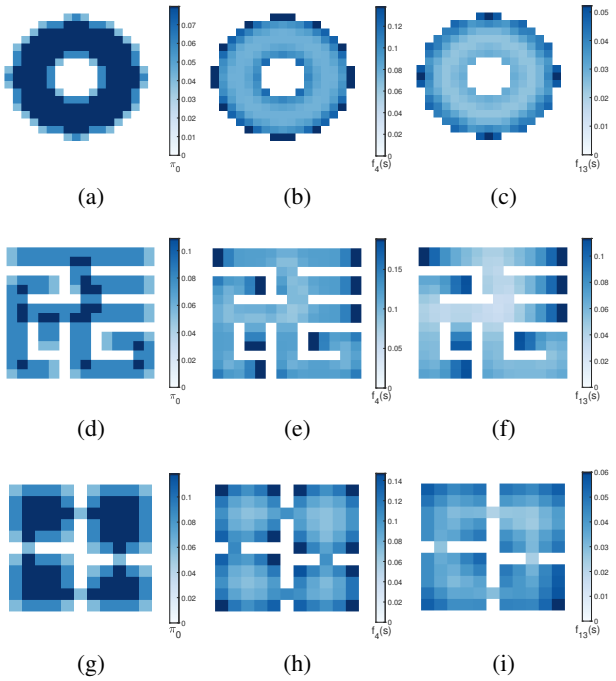


Figure 1. The domains colored according to (a,d,g) the stationary distribution π_0, (b,e,h) the options generating function $f_4(s)$, and (c,f,i) the options generating function $f_{13}(s)$.

The main hyperparameter of the algorithm is t. In our implementation, we set $t = 4$. Our empirical study shows that different values of t lead to similar results. For results using other t values and for a further discussion on the choice of t, see the SM.

4.1. Diffusion Options Generation

In Fig. 1, we plot the options generating function $f_t(s)$ for two values of t as well as the stationary distribution. First, we observe the low pass filter effect obtained by increasing the scale parameter t. Particularly, we see that $f_{13}(s)$ is smoother, containing fewer peaks, than $f_4(s)$. Second, we observe that the minima of the stationary distribution coincide with the local maxima of $f_t(s)$ for some cases, in accordance with Proposition 2. For example, note the corners of the rooms and the doors in the 4Rooms domain (Figs. 1(g) and 1(h)). Nevertheless, we observe that the local minima of the stationary distribution might also capture irrelevant states in evolved domains. For example, in

the Maze domain, in contrast to the stationary distribution, $f_t(s)$ captures the end of the corridors *only* (see Figs. 1(d) and 1(f)), which are important for efficient exploration and learning in this domain.

4.2. Exploration and Learning

Figure 2 presents the results obtained by setting $t = 4$ for all domains.

We observe in the visitation count plots that the diffusion options lead the agent to the goal states through the shortest path, e.g., in the Ring domain, following the inner ring. Importantly, these results are obtained by the diffusion options that were built in advance without access to the location of the start and goal states. Conversely, we observe that the eigenoptions lead the agent less efficiently, for example, in the Ring domain, through both the inner and the outer rings. While both the diffusion options and the eigenoptions result in informed trajectories to the goal, we observe that the naïve random walk tends to concentrate near the start state.

Figure 2 also shows that the diffusion options demonstrate the fastest learning convergence, followed by the eigenoptions and then the random walk. In addition, the diffusion options lead to convergence to shorter paths to a goal compared to the eigenoptions. These convergence results coincide with the visitation count. For example in the Ring domain, by employing the eigenoptions, the agent travels via states at the outer ring which are not on the shortest path to the goal. The significant gap in performance between the diffusion options and the eigenoptions in the Maze domain may be explained by the fact that the option goal states of the diffusion options are located at the end of the corridors (see Fig. 1), leading to efficient exploration, and in turn, to this fast learning convergence. We note that the zero variance in the learning curves at the beginning of the learning implies that the agent did not reach its goal during the episode, so the same default value was set.

For a fair comparison, we use the same number of options in both algorithms with the same Q learning configuration described above. In the SM, we present results, where the number of eigenoptions is tuned to attain maximal performance. Even after tuning, the diffusion options outperform the eigenoptions.

Table 1 shows the number of steps between states. We observe that the diffusion options lead to more efficient transitions between states compared to the eigenoptions and a random walk.

4.3. Stochastic Domains

We revisit the 4Rooms domain with the addition of a stochastic wind blowing downwards. The presence of wind is translated to the probability of $1/3$ that the agent moves down, regardless of its chosen action. As a result, the agent is more likely to visit states at the bottom of the domain, so in principle, the desired options should favor states at the upper parts of the domain.

In Fig. 3(a), we observe that $f_4(s)$ now exhibits high values at the upper part of the rooms, rather than high values at the corners and boundaries as in Fig. 1(h) without the wind. To compare the learning convergence, we adapt the eigenoptions to the stochastic domain by considering the eigenvectors of the positive part of the polar decomposition of the Laplacian as eigenoptions. Figure 3(b), presenting the learning convergence, shows a clear advantage to the use of the diffusion options compared to the eigenoptions in this stochastic setting.

4.4. Diffusion Distance and Domain Difficulty

We empirically show that the diffusion distance is related to the "domain difficulty". We propose to approximate the difficulty by the average diffusion distance between every pair of states, and compare it with two other measures of difficulty: the average time duration required for learning a task using primitive actions, i.e. the learning rate, and the average number of steps between pairs of states. Note that the computation of diffusion distances is intrinsic, i.e., it takes into account only the geometry of the domain. Consequently, it can be computed per domain a-priori without any task assignment or access to rewards. Conversely, the learning rate and the average number of steps are computed in the context of learning particular tasks and rewards, and as a result, convey their difficulties as well.

For each domain, the average diffusion distance between all states is computed. To account for the domain size, we multiply the average diffusion distance by the number of accessible states. In addition, we compute the average of diffusion distance over 100 different scales of t from a regular grid between 1 and 1000.

The results are: 13.6, 20.5, and 8.6 for the Ring, the Maze, and the 4Rooms domains, respectively. We observe that the obtained value in the Maze is higher than the obtained value in the Ring, despite having fewer states. Indeed, the learning convergence in the Maze is slower (see Figs. 2(j) and 2(e)) and the average number of steps between states is higher as well (see Table 1).

The relation between the domain difficulty and the diffusion distance gives another justification to the proposed algorithm. By Proposition 1, acting according to a diffusion option leads the agent to a distant state in terms of the diffusion distance. As a result, it can be seen as a way to effectively reduce the domain difficulty.

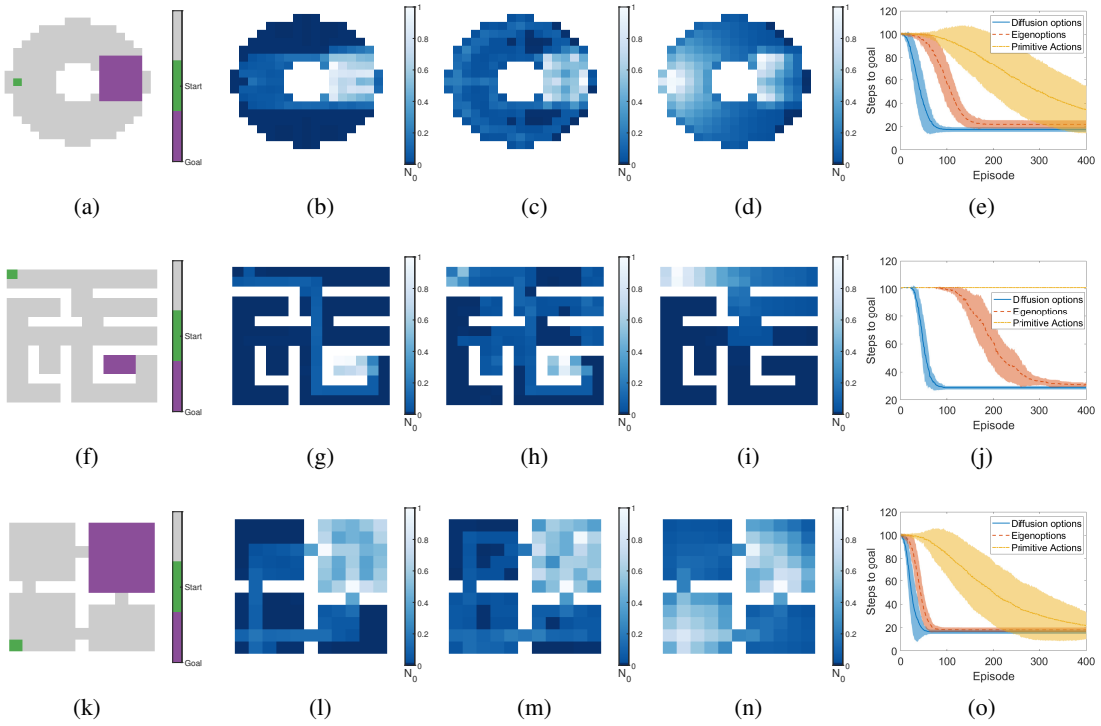


Figure 2. Learning results on the Ring domain (top row), the Maze domain (middle row), and 4Rooms domain (bottom row). (a,f,k) The start state (green) and goal states (purple). (b-d,g-i,l-m) Normalized visitation count N_0 obtained based on (b,g,l) the diffusion options, (c,h,m) the eigenoptions, and (d,i,n) a random walk (d). For visualization purposes, the visitation number is normalized to the range of $[0, 1]$ by dividing by the maximum number of visitations. (e,j,o) The learning convergence depicting the average number of steps to goal for each learning episode. The solid line represents the mean value and the light colors represent the standard deviation.

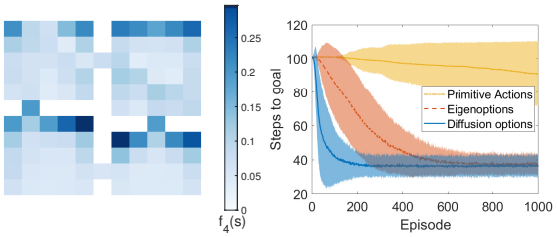


Figure 3. 4Rooms domain with stochastic wind blowing downwards. (a) The domain is colored by $f_4(s)$, where we observe that the local maxima are at the top rooms, compensating for the wind. See Fig. 1 for comparison to the result without wind. (b) The obtained learning convergence.

5. Relation to Existing Work

Option discovery has attracted much interest in recent years, resulting in numerous methods from various perspectives such as information theoretic (Mohamed & Rezende, 2015; Florensa et al., 2017; Hausman et al., 2018), learning hierarchy (Bacon et al., 2017; Vezhnevets et al., 2017), and curiosity Pathak et al. (2017), to name but a few. Discovering options without reward has been a recent active research subject. Combining information theory and skill discovery, Eysenbach et al. (2019) proposed to view skills as mixtures of policies, and to derive policies without a reward using an information theoretic objective function. There, a two-stage approach, similar to the present paper, was presented. In the first stage, the domain is scanned with no reward and the options are computed, and in the second stage, the options are utilized for learning in the context of particular rewards.

The notion of "bottleneck" states has assumed a central role in option discovery. For example, Menache et al. (2002); Şimşek et al. (2005); Mannor et al. (2004) propose to define and to identify bottleneck states using graph and spectral clustering methods. Unfortunately, these approaches fail in domains such as the Ring domain, for which clustering is not well defined. An alternative approach presented by Stolle & Precup (2002) defines bottleneck states as frequently visited states. Recently, Goyal et al. (2019) showed that this definition might lead to the discovery of redundant options in domains such as a T-shaped domain.

Perhaps the closest algorithm to ours for option discovery was presented by Machado et al. (2017). There, the agent

Table 1. Number of steps between any pair of states using options induced by $t = 4$ and by $t = 13$. We report the median value and the interquartile range (IQR) over all pairs. See the SM, for mean and standard deviation.

Domain (#states)	t	#options	Diffusion Options		Eigenoptions		Random Walk	
			Median	IQR	Median	IQR	Median	IQR
Ring (192)	4	32	217	101	301	210	565	160
	13	28	219	110	279	232	565	160
Maze (148)	4	19	282	194	446	573	1280	960
	13	14	249	160	641	781	1280	960
4Rooms (104)	4	20	147	137	160	114	487	104
	13	15	140	96	162	151	487	104

uses a subset of eigenvectors of the graph Laplacian of the domain. Each eigenvector (up to a sign) prescribes a value function assigned to each option (termed eigenoption). The agent follows the eigenvector until it reaches a local extremum, where the option terminates. A natural question that arises is why the extrema of the eigenvectors are good option goal states. Here we offer a plausible answer from a diffusion distance perspective. In Cheng et al. (2019), the diffusion distance to a subset $\mathbb{B} \subset \mathbb{S}$ is defined, and a lower bound is derived. In the present paper notation, the formulation of the bound is as follows. Let $d_{\mathbb{B}_i}(s)$ be the smallest number of steps, such that the random walk starting from state s reaches the subset $\mathbb{B}_i$ with probability greater than $\frac{1}{2}$. Then for $\mathbb{B}_i = \{s \in \mathbb{S} : -\epsilon \leq \psi_i(s) \leq \epsilon\}$ the following holds:

$$d_{\mathbb{B}_i}(s) \log\left(\frac{1}{|1-\nu_i|}\right) \geq \log\left(\frac{|\psi_i(s)|}{\|\psi_i\|_{L^\infty}}\right) - \log\left(\frac{1}{2}+\epsilon\right),$$

where ν_i and ψ_i are a pair of eigenvalue and its associated eigenvector of the normalized graph Laplacian $\boldsymbol{N}$. For small ϵ, the set $\mathbb{B}_i$ is the set of states for which $\psi_i(s)$ is close to zero. By following an eigenoption defined by the eigenvector ψ_i, the agent moves toward states that are distant from the states in $\mathbb{B}_i$. For instance, consider a domain that is comprised of 2 clusters. For such a domain, $\mathbb{B}_2$, derived from ψ_2, is the set of bottleneck states separating the 2 clusters. Thus, the eigenoption leads the agent away from bottleneck states.

In contrast, by Proposition 1, the goal states of diffusion options are states that are distant from *all* states (on average). Diffusion distance, which is closely related to the proposed options via Proposition 1, takes into account the structure of the domain, including bottlenecks. In addition, Proposition 2 also implies on the tight relation between diffusion options and bottleneck states because bottlenecks often lie at the minima of the stationary distribution.

Our options-generating function $f_t(s)$ in (1) is related to recent work in data analysis as well. Similar functions to $f_t(s)$, constructed from the eigenvectors of the graph Laplacian, were proposed for anomaly detection and clustering in Cheng et al. (2018) and Cheng & Mishne (2018), respectively. Particularly, in Cheng & Mishne (2018), a function called spectral norm was introduced, and analyzed, showing that the proliferation of eigenvectors is beneficial for clustering. In this work, we show that the same approach of combining all eigenvectors together, rather than using them separately (as the common practice is, for instance in PCA), is beneficial for option discovery.

6. Conclusions

We presented a method to derive options based on the full spectrum of the graph Laplacian. The main ingredient in the derivation and the subsequent analysis is the diffusion distance, a notion that was introduced in the context of manifold learning primarily for high-dimensional data analysis. We tested our options using Q learning in three domains, demonstrating improved exploration and learning compared to competing options.

We believe that a similar approach with such geometric considerations can be beneficial in other problems. Particularly, in future work we plan to explore its use for state aggregation (Singh et al., 1995; Duan et al., 2019). States that belong to the same partition have the same transition probabilities, and as a consequence, the diffusion distance between them is zero. Therefore, it seems only natural to utilize this notion of distance for this problem. In addition, we will study the possibility to combine model-based state transition learning with the formation of an empirical graph Laplacian.

References

Bacon, P.-L., Harb, J., and Precup, D. The option-critic architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Belkin, M. and Niyogi, P. Convergence of laplacian eigenmaps. In *Advances in Neural Information Processing Systems*, pp. 129–136, 2007.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pp. 1471–1479, 2016.

Bérard, P., Besson, G., and Gallot, S. Embedding riemannian manifolds by their heat kernel. *Geometric & Functional Analysis GAFA*, 4(4):373–398, 1994.

Bronstein, A. M., Bronstein, M. M., Guibas, L. J., and Ovsjanikov, M. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Transactions on Graphics (TOG)*, 30(1):1–20, 2011.

Cheng, X. and Mishne, G. Spectral embedding norm: Looking deep into the spectrum of the graph Laplacian. *Journal on Imaging Sciences (SIAM)*, 2018.

Cheng, X., Mishne, G., and Steinerberger, S. The geometry of nodal sets and outlier detection. *Journal of Number Theory*, 185:48–64, 2018.

Cheng, X., Rachh, M., and Steinerberger, S. On the diffusion geometry of graph laplacians and applications. *Applied and Computational Harmonic Analysis*, 46(3): 674–688, 2019.

Chui, C. K. and Mhaskar, H. N. Deep nets for local manifold learning. *Frontiers in Applied Mathematics and Statistics*, 4:12, 2018.

Chung, F. R. and Graham, F. C. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

Coifman, R. R. and Lafon, S. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006a.

Coifman, R. R. and Lafon, S. Geometric harmonics: a novel tool for multiscale out-of-sample extension of empirical functions. *Applied and Computational Harmonic Analysis*, 21(1):31–52, 2006b.

Duan, Y., Ke, T., and Wang, M. State aggregation learning from markov transition data. In *Advances in Neural Information Processing Systems*, pp. 4488–4497, 2019.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. *International Conference on Learning Representations (ICLR)*, 2019.

Florensa, C., Duan, Y., and Abbeel, P. Stochastic neural networks for hierarchical reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2017.

Fowlkes, C., Belongie, S., and Malik, J. Efficient spatiotemporal grouping using the nystrom method. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pp. I–I. IEEE, 2001.

Goyal, A., Islam, R., Strouse, D., Ahmed, Z., Botvinick, M., Larochelle, H., Levine, S., and Bengio, Y. Infobot: Transfer and exploration via the information bottleneck. *International Conference on Learning Representations (ICLR)*, 2019.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. Learning an embedding space for transferable robot skills. *International Conference on Learning Representations (ICLR)*, 2018.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Kleinberg, J. M. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

Koren, Y. On spectral graph drawing. In *International Computing and Combinatorics Conference*, pp. 496–508. Springer, 2003.

Lafon, S., Keller, Y., and Coifman, R. R. Data fusion and multicue data matching by diffusion maps. *IEEE Transactions on pattern analysis and machine intelligence*, 28 (11):1784–1797, 2006.

Liu, J., Yang, Y., and Shah, M. Learning semantic visual vocabularies using diffusion distance. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 461–468. IEEE, 2009.

Machado, M. C., Bellemare, M. G., and Bowling, M. A laplacian framework for option discovery in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2295–2304. JMLR. org, 2017.

Machado, M. C., Rosenbaum, C., Guo, X., Liu, M., Tesauro, G., and Campbell, M. Eigenoption discovery through the deep successor representation. *International Conference on Learning Representations (ICLR)*, 2018.

Mahadevan, S. and Maggioni, M. Proto-value functions: A Laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 8(Oct):2169–2231, 2007.

Mahmoudi, M. and Sapiro, G. Three-dimensional point cloud recognition via distributions of geometric distances. *Graphical Models*, 71(1):22–31, 2009.

Mannor, S., Menache, I., Hoze, A., and Klein, U. Dynamic abstraction in reinforcement learning via clustering. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 71. ACM, 2004.

Menache, I., Mannor, S., and Shimkin, N. Q-cutdynamic discovery of sub-goals in reinforcement learning. In *European Conference on Machine Learning*, pp. 295–306. Springer, 2002.

Mhaskar, H. N. A unified framework for harmonic analysis of functions on directed graphs and changing data. *Applied and Computational Harmonic Analysis*, 44(3): 611–644, 2018.

Mishne, G., Shaham, U., Cloninger, A., and Cohen, I. Diffusion nets. *Applied and Computational Harmonic Analysis*, 2017.

Mohamed, S. and Rezende, D. J. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 2125–2133, 2015.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 3303–3313, 2018.

Nakatsukasa, Y., Bai, Z., and Gygi, F. Optimizing halley's iteration for computing the matrix polar decomposition. *SIAM Journal on Matrix Analysis and Applications*, 31 (5):2700–2720, 2010.

Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 16–17, 2017.

Portegies, J. W. Embeddings of riemannian manifolds with heat kernels and eigenfunctions. *Communications on Pure and Applied Mathematics*, 69(3):478–518, 2016.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320, 2015.

Shah, D. and Xie, Q. Q-learning with nearest neighbors. In *Advances in Neural Information Processing Systems*, pp. 3111–3121, 2018.

Şimşek, Ö., Wolfe, A. P., and Barto, A. G. Identifying useful subgoals in reinforcement learning by local graph partitioning. In *Proceedings of the 22nd international conference on Machine learning*, pp. 816–823. ACM, 2005.

Singh, S. P., Jaakkola, T., and Jordan, M. I. Reinforcement learning with soft state aggregation. In *Advances in neural information processing systems*, pp. 361–368, 1995.

Spielman, D. Lecture notes. *http://www.cs.yale.edu/homes/spielman/561/syllabus.html*, 2018.

Stolle, M. and Precup, D. Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, pp. 212–223. Springer, 2002.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211, 1999.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, O. X., Duan, Y., Schulman, J., DeTurck, F., and Abbeel, P. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, pp. 2753–2762, 2017.

Van Dijk, D., Sharma, R., Nainys, J., Yim, K., Kathail, P., Carr, A. J., Burdziak, C., Moon, K. R., Chaffer, C. L., Pattabiraman, D., et al. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3): 716–729, 2018.

Verma, N. Mathematical advances in manifold learning. *Technical Report. San Diego: University of California*, 2008.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. Feudal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3540–3549. JMLR. org, 2017.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Williams, C. K. and Seeger, M. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pp. 682–688, 2001.

Wu, Y., Tucker, G., and Nachum, O. The Laplacian in RL: Learning representations with efficient approximations. *International Conference on Learning Representations (ICLR)*, 2019.

Supplementary Material for Option Discovery in the Absence of Rewards with Manifold Analysis

A. Diffusion Distance

The diffusion distance between two points s and s', as defined in section 2.2, can be computed using all eigenvectors of the random walk matrix $\boldsymbol{W}$ (Coifman & Lafon, 2006a):

$$D_t^2(s, s') = \sum_{i \geq 0} \omega_i^{2t} (\tilde{\phi}_i(s) - \tilde{\phi}_i(s'))^2, \tag{3}$$

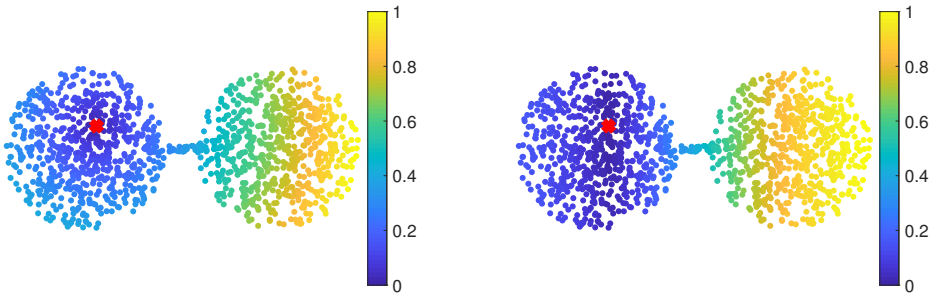
where $\{\tilde{\phi}_i\}$ and $\{\omega_i\}$ are the right eigenvectors and eigenvalues of $\boldsymbol{W}$, respectively. Expressing the diffusion distance using the full spectrum of $\boldsymbol{W}$ is an important property, which is shown to be useful for option discovery, as we show in this work.

Consider the l-dimensional *diffusion maps* embedding, where $l \leq |\mathbb{S}|$, defined by $[\Psi_t(x)]_i \triangleq \omega_i^t \tilde{\phi}_{(i)}(x)$ for $i = 1, \ldots, l$. It was shown in Coifman & Lafon (2006a) that the diffusion distance can be well approximated by the Euclidean distance between the the diffusion maps, i.e.,

$$D_t(s, s') \approx \|\Psi_t(s) - \Psi_t(s')\|$$

where equality holds if all the eigenvectors are used ($l = |\mathbb{S}|$). This approximation prescribes a convenient way to compute the diffusion distance efficiently.

An important property of the diffusion distance is that it captures both the local and the global structure of a data set. A prototypical example of a point cloud with a dumbbell shape is presented in Fig. 4. Consider an arbitrary reference point in the left cluster, marked in red. The remaining points are colored according to their distance to this reference point. In Fig. 4(a) the distance is the Euclidean distance and in Fig. 4(b) the distance is the diffusion distance. We observe that the diffusion distance captures the two-cluster structure of the data set; the diffusion distances of the points residing in the right cluster are larger than the diffusion distances of the points residing in the left cluster, with a sharp transition at the bottleneck. Conversely, the Euclidean distances do not demonstrate such a clear transition between the two clusters.



(a) (b)

Figure 4. Diffusion distance illustration. (a) The Euclidean distances and (b) the diffusion distances from a reference point (marked as a red cross). In contrast to the Euclidean distances, the diffusion distances in the right cluster are larger compared to the diffusion distances in the left cluster.

B. The Normalized Graph Laplacian

In many applications involving graphs, the construction of the normalized graph Laplacian $\boldsymbol{N}$ and its spectral decomposition were shown to be useful. The normalized graph Laplacian is defined by

$$\boldsymbol{N} = \boldsymbol{D}^{-\frac{1}{2}} \left(\boldsymbol{I} - \boldsymbol{M}\right) \boldsymbol{D}^{-\frac{1}{2}}, \tag{4}$$

where $\boldsymbol{M}$ is the symmetric adjacency matrix and $\boldsymbol{D}$ is corresponding diagonal degree matrix.

A symmetric adjacency matrix $\boldsymbol{M}$ leads to a symmetric $\boldsymbol{N}$. Therefore, $\boldsymbol{N}$ has real eigenvalues, denoted by $\{\nu_i\}$, and orthonormal eigenvectors $\{\psi_i\}$.

The eigenvalue decomposition of the lazy random walk matrix $\boldsymbol{W}$, given by $\boldsymbol{W} = \frac{1}{2}(\boldsymbol{I} + \boldsymbol{M}\boldsymbol{D}^{-1})$, is tightly related to the eigenvalue decomposition of the normalized graph Laplacian $\boldsymbol{N}$. First, observe that $\boldsymbol{W}$ can be recast as $\boldsymbol{W} = \boldsymbol{D}^{\frac{1}{2}} \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{N}\right) \boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I} - \frac{1}{2}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{N}\boldsymbol{D}^{-\frac{1}{2}}$. Then, by definition, $\boldsymbol{N}$ and $\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{N}\boldsymbol{D}^{-\frac{1}{2}}$ are similar, and so they share their real eigenvalues and the following holds:

$$\begin{aligned}
\omega_i &= \left(1 - \frac{1}{2}\nu_i\right), \\
\phi_i &= \boldsymbol{D}^{-\frac{1}{2}}\psi_i, \\
\tilde{\phi}_i &= \boldsymbol{D}^{\frac{1}{2}}\psi_i,
\end{aligned} \tag{5}$$

where ω_i are the eigenvalues of $\boldsymbol{W}$ and ϕ_i and $\tilde{\phi}_i$ are its left and right eigenvectors. As a consequence, Algorithm 1 in the paper can be implemented as follows. First, the normalized graph Laplacian $\boldsymbol{N}$ is computed instead of $\boldsymbol{W}$. Then, eigenvalue decomposition (EVD) is applied to $\boldsymbol{N}$. Finally, $\{\omega_i\}$, $\{\phi_i\}$, and $\{\tilde{\phi}_i\}$ are computed based on (5), and the remainder of the algorithm is completed as is.

C. Proofs

C.1. Proof of Proposition 1

The setting and notation follow Spielman (2018).

We consider the lazy random walk matrix, defined by $\boldsymbol{W} = \frac{1}{2}\boldsymbol{I} + \frac{1}{2}\boldsymbol{M}\boldsymbol{D}^{-1}$ for a symmetric adjacency matrix $\boldsymbol{M}$ and the corresponding diagonal degree matrix $\boldsymbol{D}$. Using (4), we can rewrite $\boldsymbol{W}$ as $\boldsymbol{W} = \boldsymbol{D}^{\frac{1}{2}} \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{N}\right) \boldsymbol{D}^{-\frac{1}{2}}$.

The probability distribution after t steps of the lazy random walk is $\boldsymbol{p}_t = \boldsymbol{W}^t\boldsymbol{p}_0 = \boldsymbol{D}^{\frac{1}{2}} \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{N}\right)^t \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{p}_0$, for an initial distribution $\boldsymbol{p}_0$.

Let ν_i and ψ_i be the eigenvalues and eigenvectors of $\boldsymbol{N}$. Using (5), expanding $\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{p}_0$ in the orthonormal basis $\{\psi_i\}$, we get

$$\boldsymbol{p}_t = \boldsymbol{D}^{\frac{1}{2}} \sum_{i\geq 1} \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{N}\right)^t c_i\psi_i = \boldsymbol{D}^{\frac{1}{2}} \sum_{i\geq 1} \omega_i^t c_i\psi_i,$$

where ω_i are eigenvalues of $\boldsymbol{W}$ and $c_i = \psi_i^T \left(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{p}_0\right)$.

Plugging in the explicit form of the coefficients c_i yields

$$\boldsymbol{p}_t = \boldsymbol{D}^{\frac{1}{2}}c_1\psi_1 + \boldsymbol{D}^{\frac{1}{2}} \sum_{i\geq 2} \omega_i^t\psi_i^T \left(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{p}_0\right)\psi_i. \tag{6}$$

Let $\boldsymbol{d}$ denote the degree vector, i.e. $\boldsymbol{d} = \mathrm{diag}(\boldsymbol{D})$. The principal eigenvector ψ_1 can be expressed as $\psi_1 = \frac{\boldsymbol{d}^{\frac{1}{2}}}{\|\boldsymbol{d}^{\frac{1}{2}}\|}$ and c_1 as $c_1 = \frac{1}{\|\boldsymbol{d}^{\frac{1}{2}}\|}$. Consequently, the first term in (6) is given by $\boldsymbol{D}^{\frac{1}{2}}c_1\psi_1 = \frac{\boldsymbol{d}}{\|\boldsymbol{d}^{\frac{1}{2}}\|^2} = \pi_0$, where π_0 is the stationary distribution of $\boldsymbol{M}\boldsymbol{D}^{-1}$ (and also of $\boldsymbol{W}$), namely $\boldsymbol{M}\boldsymbol{D}^{-1}\pi_0 = \pi_0$.

Suppose the initial distribution $\boldsymbol{p}_0$ is completely concentrated at a single state s, i.e., $\boldsymbol{p}_0 = \delta_s$, where δ_s is a vector of all zeros expect the sth entry. The distribution after t steps starting with δ_s is given by

$$\boldsymbol{p}_t^{(s)} = \pi_0 + \boldsymbol{D}^{\frac{1}{2}} \frac{1}{\sqrt{\boldsymbol{d}(s)}} \sum_{i \geq 2} \omega_i^t \psi_i^T \delta_s \psi_i. \tag{7}$$

Using (7), we compute the squared diffusion distance between two states, s and s', after t steps, as follows:

$$D_t^2(s, s') = \|\boldsymbol{p}_t^{(s)} - \boldsymbol{p}_t^{(s')}\|^2 = \|\boldsymbol{D}^{\frac{1}{2}} \sum_{i \geq 2} \omega_i^t \left(\frac{\psi_i(s)}{\sqrt{\boldsymbol{d}(s)}} - \frac{\psi_i(s')}{\sqrt{\boldsymbol{d}(s')}} \right) \psi_i\|^2 = \|\sum_{i \geq 2} \omega_i^t \left(\phi_i(s) - \phi_i(s')\right) \tilde{\phi}_i\|^2$$

using (5). By expanding the norm, we get

$$D_t^2(s, s') = \|\sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i\|^2 + \|\sum_{i \geq 2} \omega_i^t \phi_i(s') \tilde{\phi}_i\|^2 - 2 \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i^T \sum_{j \geq 2} \omega_j^t \phi_j(s') \tilde{\phi}_j.$$

Finally, by Lemma 1, we have

$$\left\langle D_t^2(s, s') \right\rangle_{s' \in \mathbb{S}} = \|\sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i\|^2 + \left\langle \|\sum_{i \geq 2} \omega_i^t \phi_i(s') \tilde{\phi}_i\|^2 \right\rangle_{s' \in \mathbb{S}}$$

where the second term is a constant independent of s.

Lemma 1. *Define $\tilde{\phi}_i$ as in (5). Then, assuming a connected graph, we have*

$$\langle \tilde{\phi}_i(s) \rangle_{s \in \mathbb{S}} = 0 \qquad \forall i \geq 2$$

Proof. The eigenvectors $\{\psi_i\}$ are orthonormal, so $\psi_1^T \psi_i = 0$ for all $i \geq 2$.
Using $\psi_1 = \frac{\boldsymbol{d}^{\frac{1}{2}}}{\|\boldsymbol{d}^{\frac{1}{2}}\|}$ leads to

$$0 = \psi_1^T \psi_i = \frac{\left(\boldsymbol{d}^{\frac{1}{2}}\right)^T}{\|\boldsymbol{d}^{\frac{1}{2}}\|} \psi_i = \frac{\left(\boldsymbol{d}^{\frac{1}{2}}\right)^T}{\|\boldsymbol{d}^{\frac{1}{2}}\|} \boldsymbol{D}^{-\frac{1}{2}} \tilde{\phi}_i = \frac{1}{\|\boldsymbol{d}^{\frac{1}{2}}\|} \mathbf{1}^T \tilde{\phi}_i = \frac{1}{\|\boldsymbol{d}^{\frac{1}{2}}\|} \sum_{s \in \mathbb{S}} \tilde{\phi}_i(s)$$

□

C.2. Proof of Proposition 2

Part 1. Combining (7) and (5) yields

$$\boldsymbol{p}_t^{(s)} - \pi_0 = \boldsymbol{D}^{\frac{1}{2}} \frac{1}{\sqrt{\boldsymbol{d}(s)}} \sum_{i \geq 2} \omega_i^t \psi_i^T \delta_s \psi_i = \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i,$$

and we have

$$f_t(s) \triangleq \|\sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i\|^2 = \|\boldsymbol{p}_t^{(s)} - \pi_0\|^2.$$

Part 2. The concatenation of the eigenvectors $\{\psi_i\}$ as columns of a matrix forms an orthonormal matrix, whose rows are also orthonomal. So,

$$\sum_{i \geq 2} \psi_i^2(s) = 1 - \psi_1^2(s) = 1 - \frac{\boldsymbol{d}(s)}{d(\mathbb{S})} = \frac{d(\mathbb{S}) - \boldsymbol{d}(s)}{d(\mathbb{S})},$$

where $d(\mathbb{S})$ is the degree of the set of all states, $\mathbb{S}$. We therefore have

$$\frac{d(\mathbb{S})}{\boldsymbol{d}(s)} \sum_{i \geq 2} \psi_i^2 = \frac{d(\mathbb{S}) - \boldsymbol{d}(s)}{\boldsymbol{d}(s)} = \frac{d(\mathbb{S})}{\boldsymbol{d}(s)} - 1 = \frac{1}{\pi_0(s)} - 1.$$

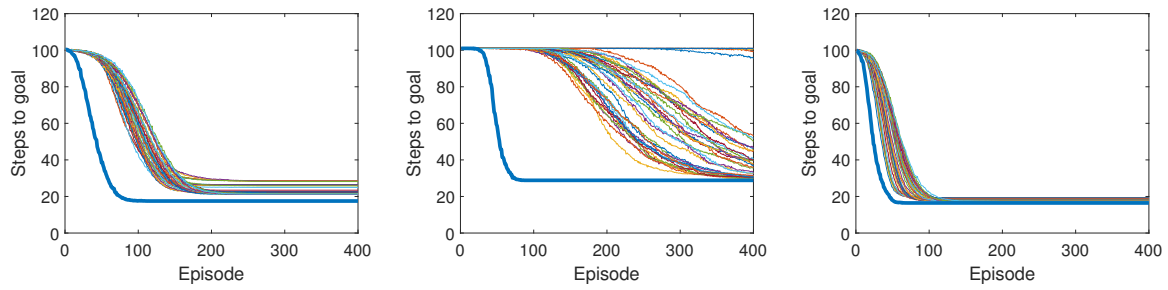


Figure 5. Learning performance of the diffusion options using $t = 4$ (bold line) and different number of eigenoptions ranging between 10 and 50. (a) The Ring domain. (b) The Maze domain. (c) The 4Rooms domain. Due to the multiple curves, the standard deviation is omitted in this figure, but we report that it is similar to the standard deviation reported in Fig. 2 in the paper.

Table 2. Number of steps between any pair of states using options induced by $t = 4$ and by $t = 13$. Here we report the mean value and the stadard deviation.

Domain (#states)	**t**	**#options**	**Diffusion Options**		**Eigenoptions**		**Random Walk**	
			Mean	STD	Mean	STD	Mean	STD
Ring (192)	4	32	228	74	327	139	618	134
	13	28	249	149	326	149	618	134
Maze (148)	4	19	296	271	551	341	1323	639
	13	14	240	134	1393	3260	1323	639
4Rooms (104)	4	20	158	81	183	88	497	80
	13	15	155	97	193	120	497	80

Finally, assuming the eigenvalues ω_i are in descending order, we have

$$
\begin{aligned}
f_t(s) &= \| \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i \|^2 \leq \omega_2^{2t} \| \sum_{i \geq 2} \phi_i(s) \tilde{\phi}_i \|^2 \\
&\leq \omega_2^{2t} \sum_{i \geq 2} \frac{\psi_i^2(s)}{\boldsymbol{d}(s)} \| \boldsymbol{D}^{\frac{1}{2}} \psi_i \|^2 \leq \omega_2^{2t} \sum_{i \geq 2} \frac{\psi_i^2(s)}{\boldsymbol{d}(s)} d(\mathbb{S}) \\
&= \omega_2^{2t} \left(\frac{1}{\pi_0(s)} - 1 \right).
\end{aligned}
$$

D. Additional Experimental Results

D.1. Further Comparison with Eigenoptions

In the paper, we test the learning performance of the diffusion options for $t = 4$, yielding 32, 19, and 20 options for the Ring, Maze and 4Rooms domains, respectively. There, we compare it to the same number of eigenoptions. To complement this comparison, here we compare the performance of the diffusion options to different number of eigenoptions, ranging between 10 and 50, for all three domains. We use the same Q learning setting as described in section 4 in the paper. The results appear in Figure 5. We observe that in all the domains the diffusion options achieve faster convergence compared to the different number of eigenoptions. In other words, the diffusion options are superior even if the number of eigenoptions is tuned to attain maximal performance.

In Table 1 in the paper we reported the median and the interquartile range (IQR) of the number of steps between any pair of states. Here, in Table 2, we complete the picture and present the mean and standard deviation. Broadly, we observe similar trends, where the diffusion options obtain the best results in all three domains.

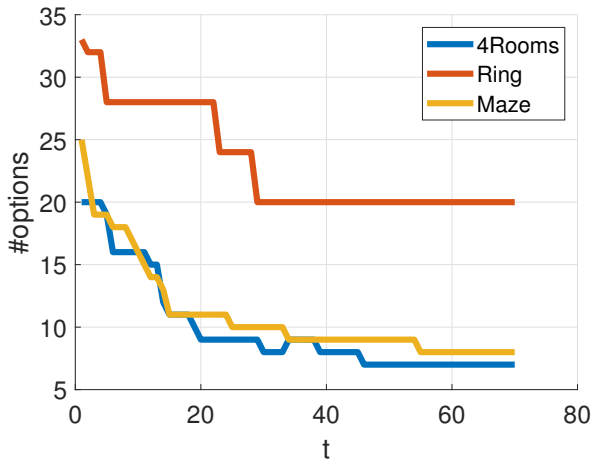


Figure 6. The number of discovered options for different scale parameter t for all 3 domains.

D.2. The Scale Parameter t

The scale parameter t demonstrates a smoothing effect, similar to a low pass filter, on $f_t(s)$. Concretely, $f_t(s)$ displays fewer peaks, which results in a fewer number of diffusion options, as t increases.

Figure 6 shows the number of options derived from the local maxima of $f_t(s)$ as a function of t. First we observe that indeed the number of options is decreasing with t. Second, we observe that above a certain value of t (e.g., $t = 30$ in the Ring domain), the number of options empirically reaches a steady state.

In the paper, we show the performance of the diffusion options derived from $f_t(s)$ by setting $t = 4$. Here, we show results obtained by setting $t = 13$. Figure 6 implies that $t = 13$ results in fewer options than $t = 4$ due to the smoothing effect. Figure 7 presents the results in the Ring, Maze and 4Rooms domains, measured by the learning convergence and the normalized visitation count during the learning process. Broadly, we observe that the obtained results for $t = 13$ are similar to those reported in the paper for $t = 4$, where the diffusion options are superior in comparison to the eigenoptions in the three tested domains. We report that similar results were obtained for other t values as well, suggesting that the algorithm is not very sensitive to the particular value of t.

D.3. Comparison with Random Options

n order to highlight the importance of the option goal states derived in the proposed method, we compare the results to random option goal states. To this end, we first randomly choose states. Then, we generate "random options", where each random option is associated with one random goal state. We implement these random options similarly to the diffusion options, so that their policies lead the agent to their respective random goal state via the shortest path.

We compare the performance of the diffusion options, the eigenoptions, and the random options in the 4Rooms domain, where the agent starts at the bottom left corner and its goal is to reach the top right corner. We implement Q learning (as in the paper) and compute the number of steps until the goal state is reached at each episode during the learning process. We repeat this test for 100 Monte Carlo iterations. At each iteration a different set of random goal states are uniformly chosen, resulting in a different set of random options (while the diffusion options and the eigenoptions remain the same). We use 20 options (derived by setting $t = 4$) from each option kind.

Figure 8 presents the learning performance for 100 episodes, allowing to compare the learning rates, and an inset for 400 episodes, including the learning convergence. We observe that the diffusion options lead to the fastest convergence compared to the eigenoptions and the random options. Random options have a large learning variance and a slow convergence. Even after 300 episodes, the random options' performance demonstrates relatively high variance, suggesting that their performance highly depends on the randomly chosen set of goal states.

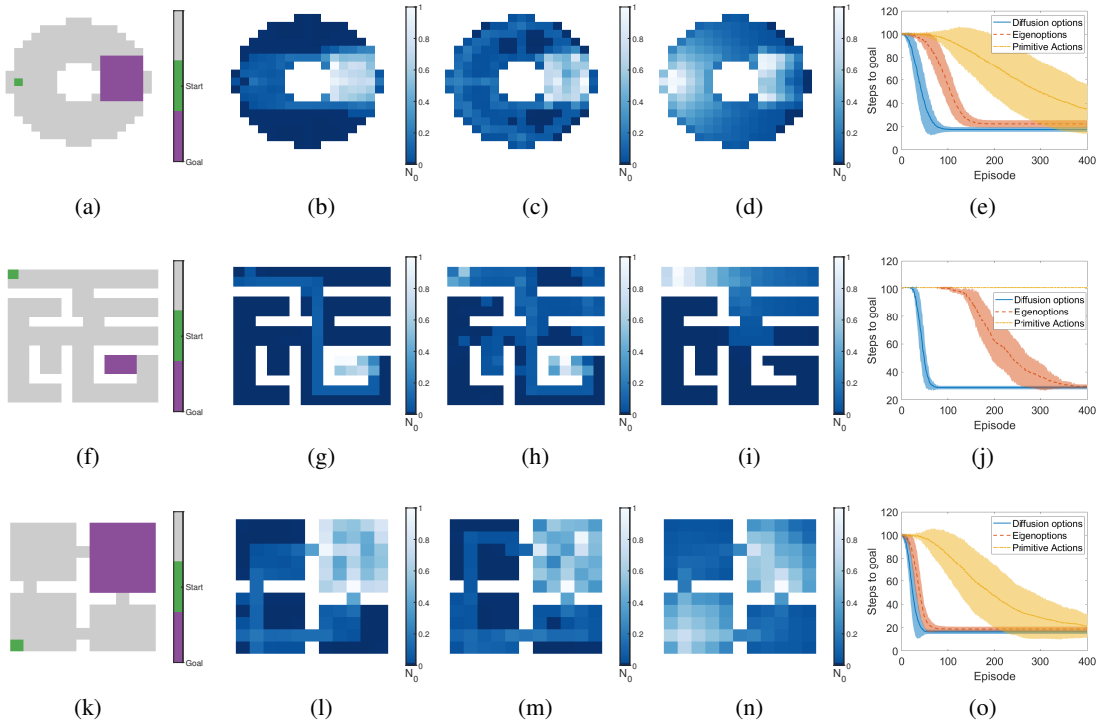


Figure 7. Learning results obtained by setting $t = 13$ in the Ring domain (top row), the Maze domain (middle row), and 4Rooms domain (bottom row). (a,f,k) The start state (green) and goal states (purple). (b-d,g-i,l-m) Normalized visitation count N_0 obtained based on (b,g,l) the diffusion options, (c,h,m) the eigenoptions, and (d,i,n) a random walk (d). For visualization purposes, the visitation number is normalized to the range of $[0, 1]$ by dividing by the maximum number of visitations. (e,j,o) The learning convergence depicting the average number of steps to goal for each learning episode. The solid line represents the mean value and the light colors represent the standard deviation.

D.4. Stochastic Domain

In a deterministic domain, the considered graph is undirected, where the adjacency matrix $\boldsymbol{M}$ merely conveys the (binary) connectivity between states, and thereby, is symmetric. Conversely, in a stochastic domain, the connectivity between states is more evolved, depending on the transition probabilities induced by the stochasticity of the domain. For example, in a domain with wind blowing downward, the agent experiences more transitions from states located at the top to states located at the bottom, than vice versa. As a result, the corresponding graph is directed and the corresponding matrix $\boldsymbol{M}$ is non-symmetric. This requires slight modifications in Algorithm 1 in the paper, which are detailed below.

Let $\boldsymbol{M}$ be the weight matrix of a directed graph. Such a matrix could represent either the true transitions or the empirical transitions experienced by the agent in some exploration phase. Forming the normalized graph Laplacian $\boldsymbol{N} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{M}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the corresponding degree matrix, results in a non-symmetric matrix, to which we cannot apply EVD directly, as prescribed in Algorithm 1 in the paper. Instead, following Mhaskar (2018), we use the polar decomposition $\boldsymbol{N} = \boldsymbol{R}\boldsymbol{U}$, and apply EVD to $\boldsymbol{R}$, obtaining its eigenvectors $\{\tilde{\psi}_i\}$ and eigenvalues $\{\tilde{\nu}_i\}$. Then, by the relations in (5), $\{\phi_i\}$,$\{\tilde{\phi}_i\}$, and $\{\tilde{\omega}_i\}$ are computed, from which $f_t(s)$ is constructed. The entire modified algorithm is presented in Algorithm 2.

We note that this modified version is also applicable as is to deterministic domains with asymmetric transitions between states. We observe that Algorithm 1 in the paper and Algorithm 2 here are very similar, and differ only in the matrix whose spectrum is used: $\boldsymbol{N}$ in Algorithm 1, and $\boldsymbol{R}$ in Algorithm 2. We examine the performance in the 4Rooms domain, making it stochastic by adding a wind blowing downward, as described in section 4.3 in the paper. Figure 9 depicts the normalized visitations at each state during the learning process. The effect of the downward wind on the visitation count is most prominent when using the primitive actions, where we observe that the agent rarely visits the top two rooms. The effect of the wind on the diffusion options and the eigenoptions is milder. Compared to Fig. 2 in the paper, we now observe

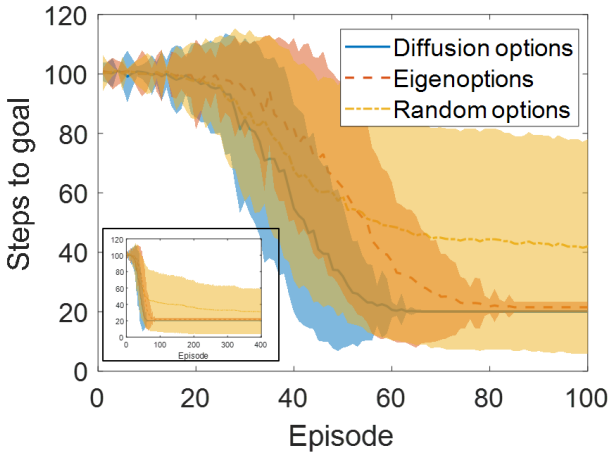


Figure 8. Comparing the learning convergence to random options in the 4Rooms domain. The learning convergence depicting the average number of steps to goal for each learning episode. The solid line represents the mean value and the light colors represent the standard deviation. The start state is the bottom left corner, and the goal state is at the top right corner. The inset presents the same plot for 400 episodes.

Algorithm 2 Diffusion Options for Stochastic Domains

Input: Adjacency matrix $\boldsymbol{M}$ and scale parameter $t > 0$

Output: K options with policies $\{\pi_o^{(i)}\}_{i=1}^K$

Compute the degree matrix $\boldsymbol{D}$ from $\boldsymbol{M}$
Compute the normalized graph Laplacian $\boldsymbol{N}$ using (4)
Compute the polar decomposition $\boldsymbol{N} = \boldsymbol{RU}$
Apply EVD to $\boldsymbol{R}$ and obtain its eigenvectors $\{\tilde{\psi}_i\}$ and eigenvalues $\{\tilde{\nu}_i\}$
Compute $\{\phi_i\}$, $\{\tilde{\phi}_i\}$ and $\{\omega_i\}$ using (5) with $\{\tilde{\psi}_i\}$ and $\{\tilde{\nu}_i\}$
Construct $f_t(s) = \| \sum_{i \geq 2} \omega_i^t \phi_i(s) \tilde{\phi}_i \|^2$
Find the local maxima of $f_t(\cdot)$ – $\{s_o^{(i)}\}_{i=1}^K$
for $i \in \{1, \ldots, K\}$ **do**
    Build an option with policy $\pi_o^{(i)}$ s.t. it leads to $s_o^{(i)}$
end for

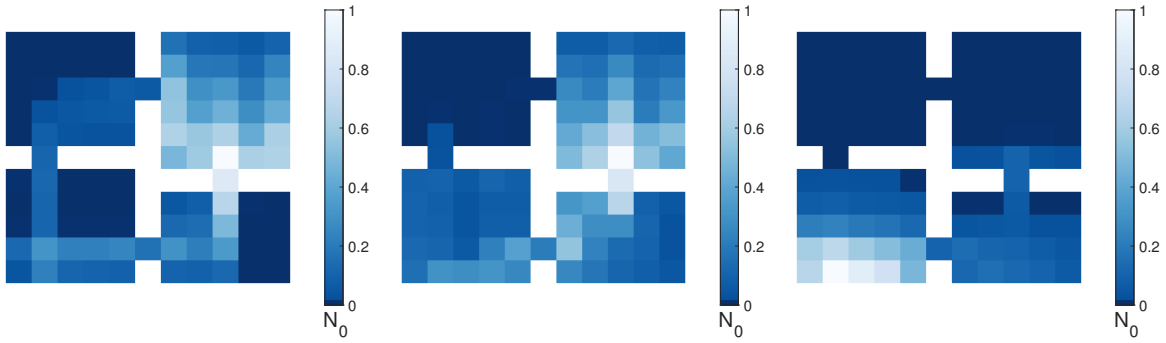


Figure 9. Normalized visitation count N_0 in the 4Rooms domain with stochastic wind blowing downward obtained based on (a) the diffusion options, (b) the eigenoptions, and (c) a random walk.

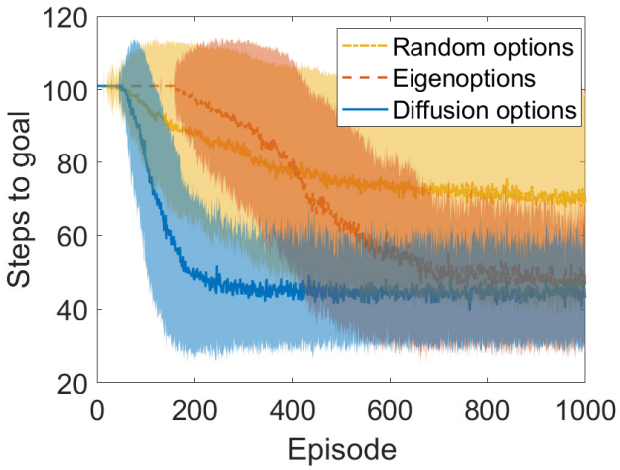


Figure 10. Learning performance in the 4Rooms domain with stochastic wind. The start state is the bottom left corner and the goal state is the top right corner.

a tendency to prefer trajectories that go through the right bottom room rather than the left top room. Still, the advantage of the diffusion options compared to the eigenoptions is evident, where the diffusion options lead the agent to the goal via shorter paths.

Figure 10 shows the learning performance in the 4Rooms domain with the stochastic wind. We observe that the diffusion options obtain the fastest convergence and the smallest standard deviation. In comparison with Fig. 8, we first observe that all the options lead to higher number of steps to goal, coinciding with the increase in the difficulty of the problem. In addition, similar trends appear, where the diffusion options lead to the fastest convergence with the smallest variance.

E. Partially Known Model

Assume that the set of states $\mathbb{S}$, which is known a-priori, is merely a subset of the states in the domain $\bar{\mathbb{S}}$, i.e., $\mathbb{S} \subset \bar{\mathbb{S}}$. Then, the question that arises is how to extend $f_t : \mathbb{S} \to \mathbb{R}$ in equation (1) in the paper from $\mathbb{S}$ to $\bar{\mathbb{S}}$, thereby allowing to discover options on the entire (unseen) domain.

We consider the following extension $g_t : \bar{\mathbb{S}} \to \mathbb{R}$

$$g_t(s) = \left\| \sum_{i \geq 2} \omega_i^t \varphi_i(s) \tilde{\varphi}_i \right\|^2, \forall s \in \bar{\mathbb{S}}, \tag{8}$$

where $\varphi_i, \tilde{\varphi}_i : \bar{\mathbb{S}} \to \mathbb{R}$ are out-of-sample extensions of ϕ_i and $\tilde{\phi}_i$, such that

$$\varphi_i(s) = \phi_i(s), \tilde{\varphi}_i(s) = \tilde{\phi}_i(s), \forall s \in \mathbb{S}.$$

By definition, the extension is consistent in the sense that

$$g_t(s) = f_t(s), \forall s \in \mathbb{S}.$$

There exist many approaches and methods for out-of-sample extension of the eigenvectors of the graph Laplacian, e.g. the classical Nyström extension (Fowlkes et al., 2001; Williams & Seeger, 2001) and geometric harmonics (Coifman & Lafon, 2006b). More recent extension methods also include methods based on neural networks (Chui & Mhaskar, 2018; Mishne et al., 2017), and a scalable approach (Wu et al., 2019) using the spectral graph drawing objective (Koren, 2003) while ensuring orthonormality.

The rationale behind the extension considered in (8) comes from the realm of manifold learning. Suppose that the domain is a manifold and that the set of states is a sample from this continuous space. In such a setting, the graph can be viewed as a discrete proxy of the manifold, and the graph Laplacian as a discrete approximation of the Laplace-Beltrami operator of the manifold (see (Coifman & Lafon, 2006a; Belkin & Niyogi, 2007)). The particular interest in the Laplace-Beltrami operator stems from the fact that it contains all the geometric information on the manifold (Bérard et al., 1994). Under this setting, the two sets of states $\mathbb{S}$ and $\bar{\mathbb{S}}$ are just two samples from the manifold. Since their respective graph Laplacians are approximations of the same Laplace-Beltrami operator, it is reasonable to assume that the eigenvalues remain the same, and the eigenvectors are interpolations of one to the other.

Once $g_t(s)$ is defined using the extended eigenvectors, and options are computed, the remaining question is how to compute the policy of each option, e.g., by computing the option value function $v(s)$ for $s \in \bar{\mathbb{S}}$ or the Q function. Building a policy that leads the agent toward any particular state in the domain can be implemented in multiple ways, e.g., using deep Q learning (DQN) architectures, which have shown success in much more involved tasks (Hessel et al., 2018; Schaul et al., 2015).

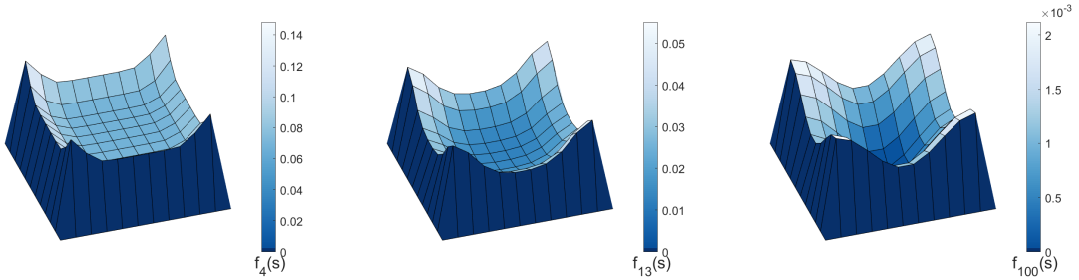
The above description implies that the extension of the proposed method requires the extension of the eigenvectors of the graph Laplacian. As described in section 3.2 in the paper, these extensions in a similar setting have already been considered and tested (Wu et al., 2019; Machado et al., 2018; Chui & Mhaskar, 2018; Mishne et al., 2017). Therefore, in principle, we can use any one of these methods.

While the extension is presented in the context of partially known models, it can also be applied as is for scaling up. The complexity of Algorithm 1 for discovering the diffusion options is governed by the EVD applied to the lazy random walk matrix (or the normalized graph Laplacian). In domains with large state spaces, or even with continuous state spaces, the construction can be based on a small representative set of states and then extended to the entire domain. Such an approach was proposed in Wu et al. (2019); Machado et al. (2018).

F. The Stationary Distribution and $f_t(s)$ in a Multidimensional Grid Domain

We further demonstrate the relations between the stationary distribution π_0 and $f_t(s)$ specifically in a multidimensional grid domain. Consider an n-dimensional grid domain with a set of states $\mathbb{S}$. Such a domain can be represented by the product of n path graphs. Let $G^{(m)}$ be a path graph with a node set $\mathbb{S}^{(m)}$ consisting of N_m nodes. Then, the product graph $G = G^{(1)} \times G^{(2)} \times \cdots \times G^{(n)}$ defined on the node set $\mathbb{S} = \mathbb{S}^{(1)} \times \mathbb{S}^{(2)} \times \cdots \times \mathbb{S}^{(n)}$ represents the n-dimensional grid domain.

The stationary distribution at state s is given by $\pi_0(s) = \frac{\boldsymbol{d}(s)}{d(\mathbb{S})}$ (Spielman, 2018), namely the stationary distribution at a certain state is proportional to its degree. This immediately yields that $\arg\min_s\{\pi_0(s)\} = \arg\min_s\{\boldsymbol{d}(s)\}$, which implies that the minima of the stationary distribution are located at states with minimal degree. In the grid domain, those states are the corner states.



(a) $f_4(s)$ (b) $f_{13}(s)$ (c) $f_{100}(s)$

Figure 11. $f_t(s)$ for different scale parameter t values for the 2D grid domain. As t increases, $f_t(s)$ becomes smoother, yet the corner states remain the maxima. We note that here we use a 3D representation for a better visualization of the maxima.

We demonstrate in a simulation that the maxima of $f_t(s)$ in a 2D grid domain are located at the corners as well. Figure 11 shows $f_t(s)$ for $t = 4$, $t = 13$, and $t = 100$. Indeed, we observe that $f_t(s)$ has maxima at the corners. In addition, we observe that $f_t(s)$ assumes a similar shape for the different t values, and that higher t values lead to a smoother function.

In addition, the corner states of a grid domain admit another property that makes them good option goal states candidates. To show this, we turn to the continuous analogue of the discrete grid domain and focus on 2 dimensions for simplicity. Namely, we present the result in the 2D $[0, L]^2$ domain.

Proposition 3. *The expected distance between a uniformly distributed state to the corners of a grid domain is the largest compared to the expected distance to any other state. Concretely, let $\boldsymbol{x}_s \in [0, L]^2$ sampled uniformly at random. We have*

$$\mathbf{E}_{\boldsymbol{x}_s\in[0,L]^2}\mathbf{E}_{\boldsymbol{x}_c\in\{0,L\}^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_c\|^2\right]=\max_{\boldsymbol{x}_o\in[0,L]^2}\mathbf{E}_{\boldsymbol{x}_s\in[0,L]^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_o\|^2\right].$$

Proposition 3 suggests that by moving to the corners, the agent covers maximal distance. In the discrete counterpart, this could imply that by doing so, the agent visits the maximal number of states, thereby encouraging exploration.

Proof. First, we compute:

$$\mathbf{E}_{\boldsymbol{x}_c\in\{0,L\}^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_c\|^2\right]=x_s^2+y_s^2+L^2-Lx_s-Ly_s$$

Then, the left hand side can be recast as:

$$\begin{aligned}\mathbf{E}_{\boldsymbol{x}_s\in[0,L]^2}\mathbf{E}_{\boldsymbol{x}_c\in\{0,L\}^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_c\|^2\right]&=\int_0^L\int_0^L\frac{1}{L^2}\mathbf{E}_{\boldsymbol{x}_c\in\{0,L\}^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_c\|^2\right]dx_sdy_s\\&=\frac{1}{L^2}\int_0^L\int_0^L\left[x_s^2+y_s^2+L^2-Lx_s-Ly_s\right]dx_sdy_s=\frac{2}{3}L^2.\end{aligned}$$

We continue with the right hand side:

$$\begin{aligned}\mathbf{E}_{\boldsymbol{x}_s\in[0,L]^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_o\|^2\right]&=\int_0^L\int_0^L\frac{1}{L^2}\left(x_o^2+y_o^2+x_s^2+y_s^2-2x_ox_s-2y_oy_s\right)dx_sdy_s\\&=x_o^2+y_o^2-Lx_o-Ly_o+\frac{2}{3}L^2.\end{aligned}$$

This expression gets its maximum value at $x_o = \{0, L\}$ and $y_o = \{0, L\}$, so we have

$$\max_{\mathbf{x}_o}\mathbf{E}_{\boldsymbol{x}_s\in[0,L]^2}\left[\|\boldsymbol{x}_s-\boldsymbol{x}_o\|^2\right]=\frac{2}{3}L^2$$

□