THE TERMINAL REPRESENTATION IN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Representation learning is a powerful tool for spatio-temporal abstraction within reinforcement learning (RL). Two well established approaches are through the successor representation (SR) and the default representation (DR). The SR encodes states by the future trajectories they induce, capturing information flow decoupled from reward. The DR builds on this by weighting trajectories with reward, integrating credit-assignment structure into the representation. Both representations, as well as their eigenvectors, have been used to support a range of downstream tasks—including option discovery, reward shaping, transfer learning, and exploration. We introduce a structurally distinct formulation: the *terminal representation* (TR). The TR encodes reward-weighted trajectories similarly to the DR, but can be learned as a lower-dimensionality object, and can be used directly for the mentioned applications without eigenvector computations. Eigenvector methods also rely on reversible transition dynamics, which the TR can bypass. In this work we develop the theoretical foundations of the TR: its derivation, convergence of two learning algorithms, its use for zero-shot compositionality, and equivalences between alternative reward formulations. We further show the TR is embedded in the top DR eigenvector, allowing it to capture the same underlying knowledge without eigendecomposition. Additionally, we provide empirical evidence of the TR as a viable alternative to existing representations in subsidiary applications, while requiring less computational overhead to learn, store, and use.

1 INTRODUCTION

Learning features for mapping problems onto more tractable forms is a priority across machine learning [4]. In Reinforcement Learning (RL), this is termed *representation learning*, and typically involves transformation of the state space.

Previous work analyses two related state representations: the *successor representation* (SR) [5] and the *default representation* (DR) [18]. In the SR, states are represented via the future visitation trajectories they induce, capturing underlying transition dynamics of the environment. The DR extends this idea to reward-weighted trajectories, and thus exploits knowledge of the credit-assignment structure.

We propose a novel approach, the *terminal representation* (TR), related to but distinct from the DR. The TR represents reward-weighted visitations of terminal states and has the following advantages:

- **Compactness** - The TR is lower dimensionality than the SR and DR, aiding more efficient learning and storage.
- **No Additional Computation** - The DR/SR rely on eigendecomposition for downstream tasks like reward shaping and option discovery, while the TR can be used directly as is.
- **Handling Asymmetry** - The above eigenvector approach relies on reversible transition dynamics, while the TR does not make such assumptions.
- **Terminal Knowledge** - The TR directly encodes information about the reachability of terminal states in the environment, weighted by the non-terminal rewards along this path—facilitating zero-shot transfer to any choice of terminal rewards.

In this work, we formally derive the TR and analyse its structure to arrive at these benefits. We prove the convergence of two learning algorithms under appropriate conditions, illustrate how the TR can

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

be used to achieve zero-shot compositionality, demonstrate structural overlap between the TR and the top eigenvector of the DR, and handle equivalence between varying reward formulations.

These theoretical properties are supplemented by empirical results, where the TR in its base form is competitive with the SR and DR across four areas: 1) *option discovery*, 2) *reward shaping*, 3) *transfer learning*, and 4) *count based exploration*. Through this empirical analysis we maintain an important thread: the TR achieves this comparable performance while requiring less computationally intensive methods.

2 BACKGROUND

Markov decision process: A *terminating* Markov decision process (MDP) can be expressed as $\mathcal{M} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ and $\mathcal{T}$ are respectively finite, *non-empty* sets of *non-terminal* and *terminal* states. The full set of states is denoted by $\mathcal{S}^+ = \mathcal{S} \cup \mathcal{T}$. We use $S = |\mathcal{S}|$ and $T = |\mathcal{T}|$ to denote the number of non-terminal and terminal states, respectively, and $S^+ = |\mathcal{S}^+|$ to denote the total number of states. $\mathcal{A}$ is the finite action set. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S}^+)$ represents the transition dynamics, while $\mathcal{R} : \mathcal{S}^+ \to \mathbb{R}$ is a *reward function*. We assume that $\mathcal{R}$ is state-dependent and that $\mathcal{R}(s) < 0$ for all $s \in \mathcal{S}$.

At each time-step t, the learning agent receives a state $s_t \in \mathcal{S}$ and executes an action $a_t \in \mathcal{A}$. The environment sends back a reward $r_t = \mathcal{R}(s_t)$, and moves the agent to a new state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$.

The agent aims to compute a stochastic *policy* $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes its expected future reward. Each π induces a *transition kernel* $\mathcal{P}^\pi : \mathcal{S} \to \Delta(\mathcal{S}^+)$ which consolidates all transition effects such that $\mathcal{P}^\pi(s' \mid s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) \cdot \mathcal{P}(s' \mid s, a)$. We adopt the convention that $\mathcal{P}^\pi(s' \mid \tau) = 0$ for all $s', \tau \in \mathcal{S}^+ \times \mathcal{T}$, i.e. terminal states do not induce any transition probability. Each policy is assumed to be *proper*, i.e. guaranteed to reach a terminal state. That is, each π induces trajectories of the form $(s_1, a_1, \dots a_{N-1}, s_N)$ such that $s_N \in \mathcal{T}$, where N is a discrete random variable representing episode length. In communicating MDPs, this is achieved by assigning non-zero probability to each action.

Another central quantity is the *value function* of π, which we define in each non-terminal state $s \in \mathcal{S}$ as $v^\pi(s) = \mathbb{E}\big[\sum_{t=1}^{N-1} \mathcal{R}(s_t) + \mathcal{R}(s_N) \mid s_1 = s\big]$, where the expectation is over the stochastic choice of next state $s_{t+1} \sim \mathcal{P}^\pi(\cdot \mid s_t)$ at each time t, and the time N it takes for the episode to end. Since $\mathcal{R}(s) < 0$ for each $s \in \mathcal{S}$, the value $v^\pi(s)$ has a well-defined upper bound. We extend the value function to each terminal state $\tau \in \mathcal{T}$ by defining $v(\tau) \equiv \mathcal{R}(\tau)$. The optimal policy is $\pi^* = \arg\max_\pi v^\pi$, and its corresponding optimal value function satisfies the Bellman optimality equations $v^*(s) = \mathcal{R}(s) + \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}^+} \mathcal{P}(s'|s, a) v^*(s')$ for all $s \in \mathcal{S}$. Note the non-linear $\max$ operation present in this equation, which limits tractability of solution methods.

Linearly-solvable Markov decision process: We consider a related MDP formulation, with properties that aid tractability. A *linearly-solvable Markov decision process* (LMDP) [23] can be defined as $\mathcal{L} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mu)$, i.e. an MDP augmented with a *default policy* $\mu : \mathcal{S} \to \Delta(\mathcal{A})$, representing the uncontrolled way of acting in this environment. All other definitions are inherited from the MDP setting. We additionally assume that any transition kernel $\mathcal{Q} : \mathcal{S} \to \Delta(\mathcal{S}^+)$ defined for each $s, s' \in \mathcal{S} \times \mathcal{S}^+$ as $\mathcal{Q}(s' \mid s) = w(s')\mathcal{P}^\mu(s' \mid s)/Z$ is *realizable*, i.e. there exists a policy π such that $\mathcal{P}^\pi = \mathcal{Q}$. Here, $w(s') > 0$ is an arbitrary weight of state s' and Z is a normalization term. For example, this condition is satisfied for deterministic MDPs.

The immediate reward of an LMDP is $\mathcal{R}(s_t, \pi) = \mathcal{R}(s_t) - \lambda \cdot \mathrm{KL}(\mathcal{P}^\pi(\cdot|s_t) \| \mathcal{P}^\mu(\cdot|s_t))$. Hence, the agent can choose $\pi(\cdot \mid s_t)$ freely, but is penalized by the KL-divergence between the induced kernel $\mathcal{P}^\pi(\cdot \mid s_t)$ and the default kernel $\mathcal{P}^\mu(\cdot \mid s_t)$. The strength of this penalty is regulated by the temperature parameter λ.

Expressing our optimal value function as v, the Bellman optimality equations can be written as

$$\begin{aligned}\frac{1}{\lambda} v(s) &= \frac{1}{\lambda} \max_\pi \left[\mathcal{R}(s, \pi) + \mathbb{E}_{s' \sim \mathcal{P}^\pi(\cdot|s)} v(s')\right] \\ &= \frac{1}{\lambda}\mathcal{R}(s) + \max_\pi \mathbb{E}_{s' \sim \mathcal{P}^\pi(\cdot|s)} \left[\frac{1}{\lambda} v(s') - \log \frac{\mathcal{P}^\pi(s'|s)}{\mathcal{P}^\mu(s'|s)}\right] \quad \forall s.\end{aligned}$$

We introduce the notation $z(s) = e^{v(s)/\lambda}$ for each $s \in \mathcal{S}^+$, and will often refer to $z(s)$ simply as the value of s. In this format, we can resolve the maximization operation inside the Bellman equations

analytically, to achieve

$$z(s) = e^{\mathcal{R}(s)/\lambda} \sum_{s'} \mathcal{P}^{\mu}(s'|s) z(s'). \tag{1}$$

The optimal control is given by a transition kernel $\mathcal{P}^*$ defined for each $s, s' \in \mathcal{S} \times \mathcal{S}^+$ as

$$\mathcal{P}^*(s' \mid s) = \frac{\mathcal{P}^{\mu}(s' \mid s) z(s')}{\sum_{s''} \mathcal{P}^{\mu}(s'' \mid s) z(s'')}.$$

By assumption, there exists a policy π such that $\mathcal{P}^{\pi} = \mathcal{P}^*$. In practice, the learner directly maintains an estimate of $\mathcal{P}^*$ from which it samples the next state, and hence never needs to compute π.

We define the following vectors and matrices. Let R be an $S^+ \times S^+$ diagonal matrix with elements $R(s,s) = e^{\mathcal{R}(s)/\lambda}$, and let P be an $S^+ \times S^+$ matrix with elements $P(s,s') = \mathcal{P}^{\mu}(s' \mid s)$. P_S is the $S \times S$ restriction of P that considers only transitions between non-terminal states, while P_T is the $S \times T$ restriction that considers transitions from non-terminal to terminal states. $R_S \in \mathbb{R}^{S \times S}$ is the non-terminal restriction of R, and y is a T-dimensional vector with elements $e^{\mathcal{R}(\tau)/\lambda}$. We let z be an S-dimensional vector with elements $e^{v(s)/\lambda}$.

Successor representation: The successor representation (SR) [5] encodes a state as the expected number of visits to its successor states, under a given policy π. This is built on the intuition that similar states lead to similar outcomes. The undiscounted SR, $\mathbf{\Psi}^{\pi} \in \mathbb{R}^{S^+ \times S^+}$, is defined as

$$\mathbf{\Psi}^{\pi} = \sum_{t=0}^{\infty} (\mathbf{P}^{\pi})^t = (\mathbf{I} - \mathbf{P}^{\pi})^{-1}. \tag{2}$$

This quantity is well defined in the undiscounted setting under the previous assumptions of our framework: policies are proper and terminal states do not induce transition probability. The SR can be learned online using temporal difference (TD) methods [5]. Once obtained, the SR for a policy π can be used to express the value function as $v^{\pi}(s) = \sum_{s' \in \mathcal{S}^+} \mathbf{\Psi}^{\pi}(s,s') \mathcal{R}(s')$.

Default representation: The SR is defined without use of the reward function, isolating the policy-dependent transition dynamics. We move now to a related concept, the default representation (DR), defined in the LMDP setting [18; 25]. The DR is 'reward-aware' in that it encodes states by the reward-weighted trajectories they induce.

Denoted $\mathbf{Z} \in \mathbb{R}^{S^+ \times S^+}$, the DR can be computed in closed form by

$$\mathbf{Z} = (R^{-1} - P)^{-1} = (I - RP)^{-1} R. \tag{3}$$

Given knowledge of transition dynamics, the DR can be used to compute the optimal value function for any configuration of terminal rewards [18; 25]. We have $z = \mathbf{Z}_S P_T y$ where $\mathbf{Z}_S = (R_S^{-1} - P_S)^{-1}$ is the $S \times S$ restriction of $\mathbf{Z}$. The DR can be learned either using a dynamic programming (DP) formulation (in the case of known dynamics) or through TD learning.

3 Related work

The terminal representation is related to the SR [5] and the DR [18; 25], both mechanically and semantically. We explore these connections throughout, but essentially the DR extends the SR's transition encodings with reward knowledge and the TR extends the DR with terminal state knowledge. The TR can also be interpreted as a special case of *default features* [25], where the choice of features indexes over terminal states. The use of representations that carry underlying information flow in MDPs is influenced by *proto value function* (PVF) theory [16], which led subsequently to the idea of *eigenoptions* which highlight this information flow in the context of temporally extended behaviour [12; 13; 15]. Machado et al. [15] in particular frame this as a cycle in which the representation informs option discovery and the resulting options refine the representation.

The thread of value function composition in LMDPs that we study can be linked to the work of Todorov [24], but our direct application of the TR in this vein is related to Infante et al. [8] who apply compositionality in a hierarchical setting. This hierarchical use of LMDPs has a longer history: Jonsson & Gómez [10] formulate each task in a hierarchy as its own LMDP, while Saxe et al. [20] exploit concurrent compositionality to blend many subtasks in parallel rather than serially. Infante

et al. [9] extend the hierarchical LMDP setting to average reward, partitioning the state space into subtasks and using compositionality to represent the high-level value function exactly.

Compositionality has overlap with the idea of transfer learning, and our investigation here can be compared to the *successor features* approach [2]. Subsequent work extends this line in relevant directions: Barreto et al. [3] combine skills in the space of cumulants to synthesise new options without learning, Alegre et al. [1] characterise when combined policies are optimal rather than merely improved, and Kuric et al. [11] learn a policy basis over subproblems that can be recombined to solve tasks specified by an automaton. These share our interest in a reusable basis that is invariant up to task specification, though they operate over reward features rather than terminal structure.

4 Terminal representation for LMDPs

This section forms the core of our work. We derive the terminal representation within the LMDP framework, and discuss how it can be learned recursively—through both a dynamic programming approach under the assumption of known dynamics, and in purely sample-based fashion.

We exploit a feature of the LMDP setting: our linear Bellman optimality equation can be decomposed into two terms, corresponding respectively to the non-terminal and terminal states:

$$z(s) = e^{\mathcal{R}(s)/\lambda} \sum_{s' \in \mathcal{S}} \mathcal{P}^{\mu}(s' \mid s) z(s') + e^{\mathcal{R}(s)/\lambda} \sum_{\tau \in \mathcal{T}} \mathcal{P}^{\mu}(\tau \mid s) e^{\mathcal{R}(\tau)/\lambda}, \quad \forall s \in \mathcal{S}.$$

This leads to matrix equations in the form

$$z = R_S P_S z + R_S P_T y \iff R_S^{-1} z = P_S z + P_T y \iff z = (R_S^{-1} - P_S)^{-1} P_T y.$$

Jointly considering the influence of environment transitions and rewards, we let $D_S = R_S P_S$ be the $S \times S$ non-terminal matrix, and $D_T = R_S P_T$ be its $S \times T$ equivalent.

Then, the matrix $M = (R_S^{-1} - P_S)^{-1} P_T = (I_S - R_S P_S)^{-1} R_S P_T = (I_S - D_S)^{-1} D_T$

is the terminal representation (TR) of the LMDP $\mathcal{L}$. Given any configuration of terminal rewards, we can use the TR to zero-shot recover the optimal z-values across all non-terminal states: $z = My$.

Learning the terminal representation: We can reformat the expression for M as a recursive dynamic programming update rule:

$M = (I_S - D_S)^{-1} D_T = \left(\sum_{i=0}^{\infty} (D_S)^i\right) D_T = D_T + D_S M.$

Theorem 4.1 *Let $M_0 = D_T$. The update rule*

$$M_{k+1} = D_T + D_S M_k \tag{4}$$

converges to the TR: $\lim_{k \to \infty} M_k = M$.

Proof. Found in Section A.1 of the appendix. ■

To handle transitions into terminal states in our vector updates, we define an extended mapping $M^+ : \mathcal{S}^+ \times \mathcal{T} \to \mathbb{R}$ such that

$$M^+(s, \tau_i) = \begin{cases} M(s, \tau_i) & \text{if } s \in \mathcal{S}, \\ \mathbb{1}_{s=\tau_i} & \text{if } s \in \mathcal{T}. \end{cases}$$

For row-wise operations, we adopt the convention that $M^+(s) = M^+(s, \cdot) \in \mathbb{R}^T$.

This leads to an online update rule for learning rows of M from samples of the form (s_t, a_t, r_t, s_{t+1}), drawn under the default policy distribution $\mathcal{P}^{\mu}$:

$$\widehat{M}(s_t) \leftarrow (1 - \alpha_t) \widehat{M}(s_t) + \alpha_t e^{\frac{r_t}{\lambda}} \widehat{M}^+(s_{t+1}) \tag{5}$$

$\widehat{M}$ denotes the current estimate of M and $\widehat{M}^+$ its extended domain equivalent.

Theorem 4.2 *Assume the learning rate α_t is such that $\sum_t^{\infty} \alpha_t = \infty$ and $\sum_t^{\infty} \alpha_t^2 < \infty$. The sample-based update rule in* (5) *will converge to M.*

Proof: Found in Section A.2 of the appendix. ■

5 STRUCTURE OF THE TERMINAL REPRESENTATION

In this section, we present a more detailed analysis of the structural properties exhibited by the terminal representation, particularly their contrasts with those of the default representation, and the consequent benefits.

Dimensionality: A straightforward advantage of the TR is compactness, allowing in practice for faster learning and more efficient storage compared to the other discussed representations.

Consider the TR expressed in the form $M = (R_S^{-1} - P_S)^{-1} P_T = \mathbf{Z}_S P_T$, where $\mathbf{Z}_S$ is the non-terminal restriction of the DR. It is simple to verify that $M \in \mathbb{R}^{S \times T}$. In our setting—with $\mathcal{S}, \mathcal{T} \neq \emptyset$—it is necessary that $S, T < S^+$. Thus M is guaranteed to be smaller than the $S^+ \times S^+$ DR.

While not a requirement, often the number of terminal states T is much smaller than S. Thus we can also say that the TR tends to be lower-dimensional than the $S \times S$ non-terminal restriction of the DR that is used in applications like zero-shot value function recovery.

Eigenvectors: The DR (as well as the SR) has been used in applications like reward shaping and option discovery. This is done through manipulation of the top eigenvector. Typically, performing a full eigendecomposition is $O((S^+)^3)$ [6], while methods like power iteration that return only the top eigenvector can cost $O((S^+)^2)$ per iteration [19]. For reward shaping, the DR and its top eigenvector are computed offline under the assumption of known dynamics. The eigenvector is then queried at each Q-learning iteration to generate an augmented learning signal. For option discovery, the DR is learnt in parallel with the value function. At each iteration we 1) update the representation, 2) compute its top eigenvector, 3) use this to generate options which are added to the agent's action set, and 4) repeat the process with this newly extended action space. If k iterations of the algorithm run, this leads to an eigenvector cost of at least $O(k \cdot (S^+)^2)$.

The TR can be used for the same applications. However, we gain a powerful advantage—the columns of the TR are directly extracted and used equivalently without additional computation. Alternative methods like *slow feature analysis* [21] learn eigenvectors directly through an online algorithm, but this in turn sacrifices access to the full predictive matrix that we require for applications like zero-shot value recovery. Learning the TR online provides access to a rich predictive matrix, while also capturing the utility of eigenvector extractions—without needing to compute eigenvectors. We formalise this relationship in the following theorem:

Theorem 5.1 *Let $\mathbf{Z}$ be the DR over $\mathcal{S}^+$ and $\mathbf{Z}_S$ its restriction to non-terminal states. Let $\tau^* = \operatorname{argmax}_{\tau \in \mathcal{T}} \mathcal{R}(\tau)$, $\mathbf{e}_{\tau^*}$ be a one-hot vector of length T that encodes τ^*, and $M_{\cdot,\tau^*}$ be the corresponding column vector of the TR M. Suppose $e^{\mathcal{R}(\tau^*)/\lambda} > \rho(\mathbf{Z}_S)$, where $\rho(\cdot)$ denotes spectral radius, which holds in particular whenever $e^{\mathcal{R}(\tau^*)/\lambda} > \frac{\gamma}{1-\gamma}$ for $\gamma := \max_{s \in \mathcal{S}} e^{\mathcal{R}(s)/\lambda} < 1$. Then the top eigenvector of $\mathbf{Z}$ is*

$$\mathbf{v}^* = \begin{bmatrix} \sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}} \mathbf{Z}_S^k M_{\cdot,\tau^*} \\ \mathbf{e}_{\tau^*} \end{bmatrix}$$

with corresponding eigenvalue $\beta^* = e^{\mathcal{R}(\tau^*)/\lambda}$.

Proof: Found in Section A.3 of the appendix. ■

So, the TR appears as a central component inside the definition of the DR's top eigenvector. Using the TR columns directly captures the same underlying information, while avoiding any eigendecomposition. This reduces complexity per vector query from $O((S^+)^2)$ or worse to an $O(1)$ operation.

Further analysis (Appendix A.3) shows that the non-terminal component $\mathbf{v}_S^*$ of $\mathbf{v}^*$ can be interpreted as a discounted reward-aware transition term over trajectories of varying lengths, weighted by transitions to the most desirable terminal state.

Continuous states: We introduce our setting with discrete state space $\mathcal{S}$, however this is not inherently required for the construction of M. For continuous $\mathcal{S}$ and finite $\mathcal{T}$, D_S and D_T become operators on bounded functions, $(D_S z)(s) = e^{\mathcal{R}(s)/\lambda} \int_{\mathcal{S}} z(s')\, \mathcal{P}^{\mu}(ds' \mid s)$ and $(D_T y)(s) = e^{\mathcal{R}(s)/\lambda} \sum_{\tau \in \mathcal{T}} \mathcal{P}^{\mu}(\tau \mid s)\, y(\tau)$. Then $\|D_S\|_\infty < 1$, the series $M = \sum_{i \geq 0} D_S^i D_T$ converges, $M(s, \tau)$ is defined pointwise, and $z = My$ still recovers the optimal values (see Appendix Section A.4).

Each row-analogue $M(s, \cdot)$ remains a vector in $\mathbb{R}^T$, making the TR a natural target for parameterised function approximation of the form $f_\theta : \mathcal{S} \to \mathbb{R}^T$. Additionally, the TR recursion $M = D_T + D_S M$ has the sample target $e^{r_t/\lambda} M^+(s_{t+1})$, which directly estimates reward-weighted terminal occupancy and never has to compute an estimate of the terminal transition kernel $\mathcal{P}^\mu(\tau \mid s)$ from $s \in \mathcal{S}$ to $\tau \in \mathcal{T}$—which in practice may be difficult or impossible to do over large or continuous spaces.

Symmetry: The underlying information structure within MDPs has been studied using the graph Laplacian $L = D - W$, where W is a symmetric weight matrix encoding state adjacency and D is the corresponding degree matrix [16]. This includes the notion of *proto-value functions* (PVF), basis functions that are derived from the eigenvectors of L. Eigendecomposition of the DR (or SR) exploits equivalence to the PVF eigenvectors through an identity of the form $P^\pi = D^{-1}W$, which requires that the distribution induced by any policy π used to compute the SR/DR is reversible: there must exist another distribution ν such that $\nu(s)\,\mathcal{P}^\pi(s' \mid s) = \nu(s')\,\mathcal{P}^\pi(s \mid s')$ for all $s, s' \in \mathcal{S}^+$.

Reversibility ensures P^π is similar to a symmetric matrix, guaranteeing real eigenvalues for the DR. Without it, the DR's eigendecomposition may yield complex eigenvalues and non-orthogonal eigenvectors, undermining its use as a structural representation of the environment. This is a strong condition. Policies are typically directional, biasing the agent toward certain states and away from others. Such policies are incompatible with the balanced flow of behaviour that reversibility demands.

The TR is constructed in a way that avoids this issue. It is of shape $S \times T$ and maps non-terminal to terminal states, making it inherently asymmetric and avoiding the need for reversibility. Its columns encode the same structural information that the DR's top eigenvector captures under reversibility (Theorem 5.1), but M is well-defined and directly usable under arbitrary non-reversible dynamics.

Hierarchy: We believe the TR will become especially useful in hierarchical RL, where an overarching decision problem is decomposed into a set of subproblems. Whether handcrafted or learned, these subproblems are naturally defined using terminal states whose values may not be known at the outset of learning. Further, two or more subproblems may share both dynamics and terminal states yet assign different terminal rewards. In this setting, estimating the TR allows zero-shot learning of subproblem value functions given a set of terminal rewards. The SR and DR do not exhibit this property, and subproblem value functions may only be encoded after additional work to estimate the transition kernel $\mathcal{P}^\mu(\tau \mid s)$ from $s \in \mathcal{S}$ to $\tau \in \mathcal{T}$.

Limitations: The TR is defined such that terminal states and their reward/transition dynamics are segmented from non-terminal states. This relies on the assumption of 1) an environment with clearly demarcated terminal states and 2) some structural knowledge of these states. The DR, defined over the full state space, does not require these conditions to be satisfied. This is a clear limitation of the TR, and restricts the environments where it can be applied. However, it is still important to acknowledge certain mitigating factors:

- The presence of terminal or 'goal' states is a common feature in RL environments, and does not constitute a fringe setting.
- Given such states, encoding direct knowledge of their relative desirability and reachability helps facilitate efficient learning.
- Thus far we have been using the language of terminal states, but it is more accurate to say the TR requires a set of demarcated states, which do not necessarily have to induce termination.

The third point is worth emphasising. The TR theory can function with arbitrarily selected 'relevant' states—high or low reward regions, bottleneck states that lead from one segment of the environment to another, boundary states. Pivoting to this view does not reproduce the flexibility offered by the DR, but we can relax the need for traditionally defined goals. Future work should formalise this extension, along with methods that automatically discover important states during learning.

6 Additional Properties: Compositionality and Alternative Reward Formulations

Reward: The LMDP framework was introduced with state-dependent rewards. It is often the case that MDPs and LMDPs are defined with rewards that depend either on the transition (s, s') or the state-action pairing (s, a), which may require a modified analysis. We deal with both these

cases by showing 1) that every LMDP with transition-dependent reward can be mapped to one with state-dependent reward, and that 2) the TR can be extended to the case of state-action rewards.

Theorem 6.1 *Consider an LMDP of the form* $\widetilde{\mathcal{L}} = \left(\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \widetilde{\mathcal{R}}, \widetilde{\mu}\right)$*, with transition dependent reward* $\widetilde{\mathcal{R}} : \mathcal{S} \times \mathcal{S}^+ \to \mathbb{R}$*. We can always transform* $\widetilde{\mathcal{L}}$ *to an equivalent LMDP*

$$\mathcal{L} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mu)$$

with state-dependent rewards $\mathcal{R} : \mathcal{S}^+ \to \mathbb{R}$*, such that* $\mathcal{L}$ *and* $\widetilde{\mathcal{L}}$ *have the same optimal value function.*

Proof: Found in section A.5 of the appendix. ■

Owing to the equivalence of $\mathcal{L}$ and $\widetilde{\mathcal{L}}$, it is sufficient to consider LMDPs with state-dependent rewards.

Theorem 6.2 *Consider an LMDP of the form* $\bar{\mathcal{L}} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \bar{\mathcal{R}}, \mu)$*, where the reward is given by* $\bar{\mathcal{R}} : \mathcal{S}^+ \times \mathcal{A} \to \mathbb{R}$*. Let* q *be the optimal action-value function satisfying Bellman equations of the form* $q = \bar{\mathcal{R}} + \lambda \log \mathbb{E}_{s'}\left[\sum_{a'} \mu(a'|s') e^{q(s',a')/\lambda}\right]$*. Now we define the state-action exponentiated values as* $\bar{z}(s,a) = e^{\frac{q(s,a)}{\lambda}}$ *for* $s \in \mathcal{S}$ *and* $\bar{z}(\tau, a) = e^{\frac{\bar{\mathcal{R}}(\tau,a)}{\lambda}}$ *for* $\tau \in \mathcal{T}$*. Let* $\bar{z} \in \mathbb{R}^{S \times A}$ *be the vector of exponentiated non-terminal action-values and* $\bar{y} \in \mathbb{R}^{T \times A}$ *its terminal equivalent. Then, we can compute the state-action TR* $\bar{M}$ *under this formulation and use it to recover action-values through the relation* $\bar{z} = \bar{M}\bar{y}$*.*

Proof: Found in Section A.6 of the appendix. ■

Composition: An attractive attribute of LMDPs is the idea of *compositionality* [24]. Given value functions for a set of base LMDPs, we can automatically recover a value function for the overarching LMDP that represents their combination. We formalise this idea as follows:

Consider a set of LMDPs $\{\mathcal{L}_1, \ldots, \mathcal{L}_T\}$, where each $\mathcal{L}_i$ is given by $(\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}_i, \mu)$. These LMDPs differ only in the reward at terminal states, i.e. $\mathcal{R}_i(s) = \mathcal{R}_j(s)$ for all $s \in \mathcal{S}$.

Now let there be a single LMDP $\mathcal{L} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mu)$, with an associated set of weights $\{w_1, \ldots, w_T\}$. Assume $z(\tau) = \sum_{i=1}^{T} w_i z_i(\tau)$ for all $\tau \in \mathcal{T}$. The Bellman equation is linear over z for non-terminal states, and so $z(s) = \sum_{i=1}^{T} w_i z_i(s)$ for all $s \in \mathcal{S}$.

We will show that the TR can also be used to perform compositionality in this fashion, bypassing the need to learn the base LMDP value functions.

Theorem 6.3 *Let* z_i *be the value of the base LMDP* $\mathcal{L}_i$ *with terminal values* $z_i(\tau) = \mathbb{1}_{\tau=\tau_i}$ *(i.e.* $\mathcal{R}_i(\tau_i) = 0$*,* $\mathcal{R}_i(\tau) = -\infty$ *otherwise). Then* $M^+(\cdot, \tau_i) = z_i$ *for* $i = 1, \ldots, T$*, and hence the TR can be used to perform compositionality via the relation* $z(s) = \sum_{i=1}^{T} z(\tau_i) M^+(s, \tau_i)$*.*

Proof: Found in Section A.7 of the appendix. ■

This result allows us to perform compositionality using only the TR, which encodes knowledge of all constituent LMDPs and eliminates the need to obtain them independently.

7 Experiments

We mirror the experimental setup used by Tse et al. [25]. Their work is focused on using the DR as an alternative to the SR, specifically analysing the impact of reward awareness. We compare the TR to both these approaches. Our narrative is not fixated on the TR as fundamentally superior across these settings, and in fact there is semantic similarity between the TR and the DR. Rather, we emphasise the competitive performance of the TR, while keeping in mind the structural advantages that have been discussed throughout.

Option discovery: Options are temporally extended actions made up of multiple 'primitive actions' [22], defined by the tuple $o = (I_o, \pi_o, \beta_o)$. $I_o \subset \mathcal{S}$ is the set of states where the option may be initiated, π_o is the option policy followed by the agent, and $\beta_o : \mathcal{S} \to [0, 1]$ controls option termination.

The SR [15] and DR [25] have been used to discover options for exploration. This is done through an iterative online algorithm: as the representation is learned, its top eigenvector generates options that are added to the action set. The SR variant is titled *covering eigenoptions* (CEO), while the DR equivalent is *reward-aware covering eigenoptions* (RACE). The TR can also be used in this cycle, but instead of computing the top eigenvector, TR columns are directly extracted. Four variants are considered: 1) use the column $M_{\cdot,\tau^*}$ of highest-reward terminal state, 2) use all columns, 3) randomly select a column, 4) cycle through the columns. We evaluate these variants against CEO, RACE, and a random walk baseline in two settings: When option discovery aids exploration during learning, and when option discovery is used to collect samples for offline Q-learning. We use the state-action TR formulation to perform Q-learning experiments (Theorem 6.2).

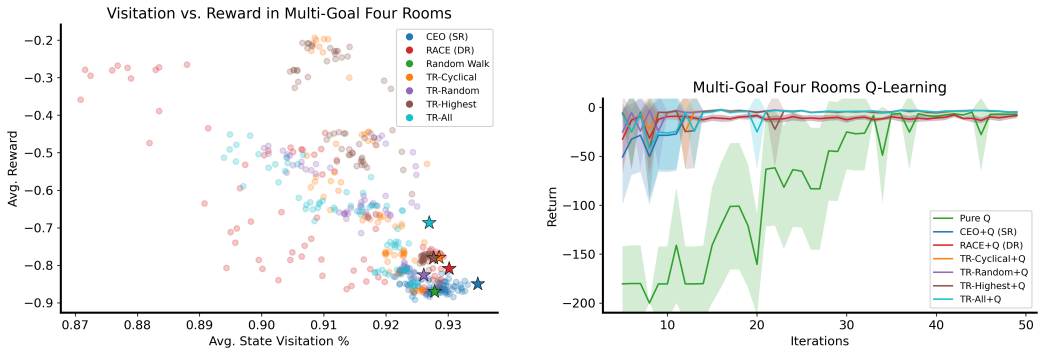

Figure 1: **Left:** TR option discovery variants vs. baselines using the DR (RACE), the SR (CEO), and a random walk. Experiments conducted in Four Rooms environment with 4 differently weighted goals. Each dot is one seed and stars mark each method's best seed across the 10 tested. **Right:** Q-learning boosted by different representations: all TR variants, the DR (RACE+Q), the SR (CEO+Q).

All TR variants produce competitive performance. Using all column vectors achieves higher average reward than the other representations. Reward-awareness of the DR leads to better returns and worse visitation than the SR [25]. The TR stretches this further, by pushing the agent directly to more desirable goals.

Reward shaping: RL environments are often defined with *sparse rewards*, where the agent only receives instructive feedback at the end of an episode. This can cause inefficient learning. *Reward shaping* combats this, letting the agent augment its reward signal with internal knowledge [17]. If $\mathbf{e}$ denotes the top eigenvector of either the SR or DR, $\hat{r}_t = -\big(\mathbf{e}(s_{\text{goal}}) - \mathbf{e}(s_{t+1})\big)^2$ and $\hat{r}_t = \mathbf{e}(s_{t+1}) - \mathbf{e}(s_t)$ are two choices of shaping reward used in previous work. For any terminal state $\tau \in \mathcal{T}$, we form the potential-based shaping reward $\hat{r}_t = M^+(\tau,\tau) - M^+(s_{t+1},\tau) = 1 - M(s_{t+1},\tau)$.

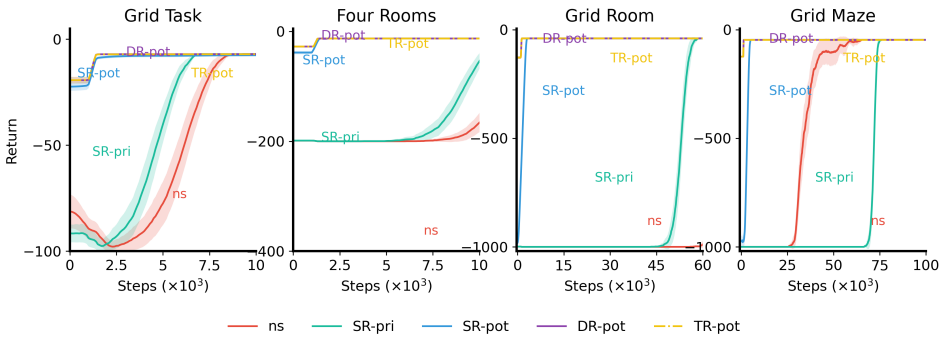

Figure 2: Reward shaping in the indicated environments, averaged over 50 seeds (TR in yellow).

Potential-based shaping with the TR achieves identical performance to the DR, successfully navigating through low reward regions—again with the caveat that this is done through direct column extraction. Both reward-aware methods outperform the SR in converging to optimal return.

Count based exploration: The SR captures state visitation counts, and thus its norm can be used as a bonus to promote exploration [14]. The same principle is applied with the DR norm [25], which

has been shown to outperform the SR as an exploration bonus. We evaluate the TR norm under the same conditions—in the RiverSwim and SixArms exploration environments. Comparison is done between unmodified Sarsa, and Sarsa with each of the three norms used as an exploration bonus.

Table 1: Returns in exploration tasks when using SARSA boosted by each representation as an exploration bonus. Values are $\times 10^3$. 95% confidence intervals shown in parentheses.

Environment	Sarsa	+SR	+DR	+TR
RiverSwim	23 (0.1)	98 (25.4)	2,933 (22.2)	2,906 (19.3)
SixArms	276 (9.4)	678 (72.1)	3,127 (214.2)	2,443 (93.6)

The TR achieves competitive performance with the DR, particularly in RiverSwim where results are near-identical. Both reward-aware representations significantly outperform the SR.

Transfer: The SR, which decouples environment dynamics and reward, has been used for transfer learning in the case where the reward function is altered. This is done through the extension to *successor features* (SF) [2]. SF is deployed alongside *generalised policy improvement* to compute a policy for a new task that combines behaviour from previously learned policies.

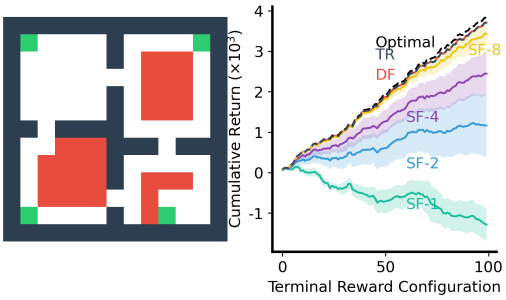

Figure 3: Transfer learning in the multi-goal Four Rooms environment, averaged over 10 seeds. The TR performs similarly to default features, while outperforming all iterations of successor features.

The policy-independent DR equivalent, *default features* (DF), views non-terminal rewards as part of the environment dynamics [25]. The decoupling is done with respect to terminal rewards only, and features are learned to model the flow of knowledge from terminal states to non-terminal states. This is equivalent to the TR where the choice of features is a one-hot encoding of terminal states. The TR is compared as a tool for transfer with the DF and the SF (under a varying number of base policies). TR performance matches the DF, without the need for an explicit feature learning step, while outperforming all variations of the SF.

8 Conclusion

We propose the terminal representation as a novel method for RL state representation. The TR captures knowledge of the underlying information flow in terminating environments, weighted by the desirability of relevant terminal states. We prove convergence of both dynamic programming and sample-based learning algorithms, and show the TR can be used to: 1) zero-shot recover optimal value functions for arbitrary terminal reward specification, 2) perform potential-based reward shaping, 3) achieve intelligent reward-aware exploration during learning, and 4) discover extended macro-actions. Through all this, we note that the TR is more compact than the default representation and successor representation, while not relying on reversible transition dynamics or requiring eigenvector computations—and in fact, the TR inherently encodes information found in the top DR eigenvector. We also provide a result on the equivalence between alternative reward structures in the LMDP setting, allowing our TR theory to translate cleanly between these formulations. Two natural directions for future work are extending to higher-dimensional environments through function approximation, and handling settings without explicitly defined terminal states.

AI use statement

In this work, we used generative AI tools for helping formulate code to run experiments and visualise their results. We have not used generative AI tools for writing the paper, finding sources and ideas, or

proving theoretical results, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for proof-reading to find typos and latex errors. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

Complete proofs of all theorems are given in Appendix A. Experimental settings, hyperparameter search spaces, and selected hyperparameters are given in Appendix B; our implementation builds on the publicly available code of Tse et al. [25] and Machado et al. [14]. The computation of confidence intervals is described in Appendix C.

References

[1] Lucas N. Alegre, Ana L. C. Bazzan, and Bruno C. da Silva. Optimistic linear support and successor features as a basis for optimal policy transfer. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 394–413, 2022.

[2] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P Van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[3] André Barreto, Diana Borsa, Shaobo Hou, Gheorghe Comanici, Eser Aygün, Philippe Hamel, Daniel Toyama, Jonathan Hunt, Shibl Mourad, David Silver, and Doina Precup. The option keyboard: Combining skills in reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pp. 13052–13062, 2019.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828, 2013.

[5] Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

[6] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4th edition, 2013.

[7] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[8] Guillermo Infante, Anders Jonsson, and Vicenç Gómez. Globally optimal hierarchical reinforcement learning for linearly-solvable markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6970–6977, 2022.

[9] Guillermo Infante, Anders Jonsson, and Vicenç Gómez. Hierarchical average-reward linearly-solvable Markov decision processes. In *Proceedings of the 27th European Conference on Artificial Intelligence (ECAI)*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, pp. 1325–1332. IOS Press, 2024. doi: 10.3233/FAIA240631.

[10] Anders Jonsson and Vicenç Gómez. Hierarchical linearly-solvable Markov decision problems. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, volume 26, pp. 193–201, 2016. doi: 10.1609/icaps.v26i1.13750.

[11] David Kuric, Guillermo Infante, Vicenc Gomez, Anders Jonsson, and Herke van Hoof. Planning with a learned policy basis to optimally solve complex tasks. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, volume 34, pp. 333–341, 2024. doi: 10.1609/icaps.v34i1.31492.

[12] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. A Laplacian framework for option discovery in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2295–2304, 2017.

[13] Marlos C. Machado, Clemens Rosenbaum, Xiaoxiao Guo, Miao Liu, Gerald Tesauro, and Murray Campbell. Eigenoption discovery through the deep successor representation. In *International Conference on Learning Representations (ICLR)*, 2018.

[14] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. Count-based exploration with the successor representation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[15] Marlos C. Machado, Andre Barreto, Doina Precup, and Michael Bowling. Temporal abstraction in reinforcement learning with the successor representation. *Journal of Machine Learning Research*, 24(80):1–69, 2023.

[16] Sridhar Mahadevan and Mauro Maggioni. Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 8(10), 2007.

[17] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning (ICML)*, 1999.

[18] Payam Piray and Nathaniel D. Daw. Linear reinforcement learning in planning, grid fields, and cognitive control. *Nature Communications*, 12(1):4942, 2021.

[19] Yousef Saad. *Numerical Methods for Large Eigenvalue Problems*. SIAM, revised edition, 2011.

[20] Andrew M. Saxe, Adam C. Earle, and Benjamin Rosman. Hierarchy through composition with multitask LMDPs. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3017–3026, 2017.

[21] Eddie Seabrook and Laurenz Wiskott. What is the relation between slow feature analysis and the successor representation? *arXiv preprint arXiv:2409.16991*, 2024.

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

[23] Emanuel Todorov. Linearly-solvable Markov decision problems. *Advances in Neural Information Processing Systems (NIPS)*, pp. 1369–1376, 2006.

[24] Emanuel Todorov. Compositionality of optimal control laws. *Advances in neural information processing systems*, 22, 2009.

[25] Hon Tik Tse, Siddarth Chandrasekar, and Marlos C Machado. Reward-aware proto-representations in reinforcement learning. *arXiv preprint arXiv:2505.16217*, 2025.

[26] John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

A THEORY

In this appendix we prove several of the theorems stated in the main text. For clarity we restate the theorems here.

A.1 PROOF OF THEOREM 4.1

Theorem 4.1 *Let $M_0 = D_T$. The update rule*

$$M_{k+1} = D_T + D_S M_k \tag{6}$$

converges to the TR: $\lim_{k\to\infty} M_k = M$.

Proof. We have $M_{k+1} = \left[\sum_{i=0}^{k+1} D_S^i\right] D_T$. Thus, $\lim_{k\to\infty} M_k = \left[\sum_{i=0}^{\infty} D_S^i\right] D_T$. The sum can be interpreted as a Neumann series, and we want to show this converges. $||D_S||_\infty = \max_{s\in\mathcal{S}} e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}} \mathcal{P}^\mu(s' \mid s)$. Since $\mathcal{R}(s) < 0$ for all $s \in \mathcal{S}$, we have $e^{\frac{\mathcal{R}(s)}{\lambda}} < 1$. The probability sum is ≤ 1, since we are considering transitions from some non-terminal state to all other non-terminal states. Therefore, the series converges and we can write $\lim_{k\to\infty} M_k = \left[\sum_{i=0}^{\infty} D_S^i\right] D_T = (I_S - D_S)^{-1} D_T = M$. ■

A.2 PROOF OF THEOREM 4.2

Theorem 4.2 *Assume the learning rate α_t is such that $\sum_t^\infty \alpha_t = \infty$ and $\sum_t^\infty \alpha_t^2 < \infty$. The sample-based update rule*

$$\widehat{M}(s_t) \leftarrow (1-\alpha_t)\widehat{M}(s_t) + \alpha_t e^{\frac{r_t}{\lambda}} \widehat{M}^+(s_{t+1})$$

will converge to M.

Proof: We can define the operator F acting on rows of M as

$$(FM)(s) = e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s) M^+(s').$$

We also introduce the stochastic noise term $w_t(s_t) = e^{\frac{r_t}{\lambda}} \widehat{M}^+(s_{t+1}) - (F\widehat{M})(s_t)$, which lets us reformat our update in stochastic approximation form:

$$\widehat{M}(s_t) \leftarrow (1-\alpha_t)\widehat{M}(s_t) + \alpha_t((F\widehat{M})(s_t) + w_t(s_t)).$$

We want to show that the operator F is a contraction. We have

$[FM - FM'](s,\tau) = e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s)[M^+ - M'^+](s',\tau)$, and so applying the triangle inequality followed by bounding each term with the infinity norm we get

$$\begin{aligned}
|FM - FM'|(s,\tau) &= |e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s)[M^+ - M'^+](s',\tau)| \\
&\leq e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s)|M^+ - M'^+|(s',\tau) \\
&\leq e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s)\|M^+ - M'^+\|_\infty \\
&= e^{\frac{\mathcal{R}(s)}{\lambda}} \|M^+ - M'^+\|_\infty \sum_{s'\in\mathcal{S}^+} \mathcal{P}^\mu(s' \mid s).
\end{aligned}$$

The probability sum evaluates to 1, so this expression becomes $e^{\frac{\mathcal{R}(s)}{\lambda}}\|M^+ - M'^+\|_\infty$. For $s \in \mathcal{S}$, it is trivial that $M^+(s,\tau) = M(s,\tau)$. For $s \in \mathcal{T}$, we have $M^+(s,\tau) - M'^+(s,\tau) = \mathbb{1}_{s=\tau} - \mathbb{1}_{s=\tau} = 0$. Thus we can write

$$\begin{aligned}\|FM - FM'\|_\infty &\leq \max_{s\in\mathcal{S}} e^{\frac{\mathcal{R}(s)}{\lambda}}\|M^+ - M'^+\|_\infty \\ &= \max_{s\in\mathcal{S}} e^{\frac{\mathcal{R}(s)}{\lambda}}\|M - M'\|_\infty \\ &= \gamma\|M - M'\|_\infty\end{aligned}$$

where $\gamma = \max_{s\in\mathcal{S}} e^{\frac{\mathcal{R}(s)}{\lambda}} < 1$ follows from the fact that $\mathcal{R}(s) < 0$ for all $s \in \mathcal{S}$. Thus F is a contraction over $\|\cdot\|_\infty$. Since $r_t = \mathcal{R}(s_t)$, the noise $w_t(s_t)$ has zero mean given the history, and $\|w_t\|_\infty \leq 2\max(1, \|\widehat{M}\|_\infty)$. With the contraction and the conditions of the theorem, the asynchronous stochastic approximation result of Tsitsiklis [26] gives $\widehat{M} \to M$ with probability 1. ■

A.3 Proof of Theorem 5.1

Theorem 5.1 *Let $\mathbf{Z}$ be the DR over $\mathcal{S}^+$ and $\mathbf{Z}_S$ its restriction to non-terminal states. Let $\tau^* = \text{argmax}_{\tau\in\mathcal{T}}\mathcal{R}(\tau)$, $\mathbf{e}_{\tau^*}$ be a one-hot vector of length T that encodes τ^*, and $M_{\cdot,\tau^*}$ be the corresponding column vector of the TR M. Suppose $e^{\mathcal{R}(\tau^*)/\lambda} > \rho(\mathbf{Z}_S)$, where $\rho(\cdot)$ denotes spectral radius, which holds in particular whenever $e^{\mathcal{R}(\tau^*)/\lambda} > \frac{\gamma}{1-\gamma}$ for $\gamma := \max_{s\in\mathcal{S}} e^{\mathcal{R}(s)/\lambda} < 1$. Then the top eigenvector of $\mathbf{Z}$ is*

$$\mathbf{v}^* = \begin{bmatrix}\sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}}\mathbf{Z}_S^k M_{\cdot,\tau^*} \\ \mathbf{e}_{\tau^*}\end{bmatrix}$$

with corresponding eigenvalue $\beta^* = e^{\mathcal{R}(\tau^*)/\lambda}$.

Proof: The full DR is given by the $S^+ \times S^+$ matrix

$$\mathbf{Z} = \Big[R^{-1} - P\Big]^{-1}. \tag{7}$$

To analyse its eigenvectors, we start with the inverted component $\mathbf{Z}^{-1} = \Big[R^{-1} - P\Big]$. This can be partitioned into submatrices of the form $\mathcal{S}\times\mathcal{S} \to \mathbb{R}, \mathcal{T}\times\mathcal{T} \to \mathbb{R}, \mathcal{S}\times\mathcal{T} \to \mathbb{R}$ and $\mathcal{T}\times\mathcal{S} \to \mathbb{R}$.

The $\mathcal{S}\times\mathcal{S}$ part is just $\Big[R_S^{-1} - P_S\Big]$.

The $\mathcal{T}\times\mathcal{S}$ part is $\mathbf{0}_{TS}$, as we are only taking off-diagonal entries from the reward matrix and because there is 0 probability of transitioning from $\tau \in \mathcal{T}$ to $s \in \mathcal{S}$.

The $\mathcal{S}\times\mathcal{T}$ part will also have off-diagonal 0 entries in the reward matrix, and the transition matrix becomes P_T. The full expression is thus $-P_T$.

Since $\mathcal{P}(s \mid \tau) = 0$ for all $s \in \mathcal{S}^+$, the $\mathcal{T}\times\mathcal{T}$ part is just a matrix over terminal rewards, $\text{diag}(y)^{-1}$.

Putting this together, we can say that

$$\mathbf{Z}^{-1} = \begin{bmatrix}(R_S^{-1} - P_S) & -P_T \\ \mathbf{0}_{TS} & \text{diag}(y)^{-1}\end{bmatrix}.$$

We can invert a matrix of this form [7] to get

$$\mathbf{Z} = \begin{bmatrix}(R_S^{-1} - P_S)^{-1} & (R_S^{-1} - P_S)^{-1}P_T\text{diag}(y) \\ \mathbf{0}_{TS} & \text{diag}(y)\end{bmatrix} \tag{8}$$

$$= \begin{bmatrix}\mathbf{Z}_S & M\text{diag}(y) \\ \mathbf{0}_{TS} & \text{diag}(y)\end{bmatrix}. \tag{9}$$

Let $\mathbf{v}^{(i)}$ and β_i denote respectively the ith eigenvector of $\mathbf{Z}$ and its corresponding eigenvalue, where the ordering is done by eigenvalue magnitude. Each $\mathbf{v}^{(i)}$ is of length S^+, and so we can decompose them into their non-terminal and terminal components: $\mathbf{v}^{(i)} = \begin{bmatrix}\mathbf{v}_S^{(i)} \mathbf{v}_T^{(i)}\end{bmatrix}$. Then we can write

$$\begin{bmatrix}\mathbf{Z}_S & M\text{diag}(y) \\ \mathbf{0}_{TS} & \text{diag}(y)\end{bmatrix}\begin{bmatrix}\mathbf{v}_S^{(i)} \\ \mathbf{v}_T^{(i)}\end{bmatrix} = \beta_i \begin{bmatrix}\mathbf{v}_S^{(i)} \\ \mathbf{v}_T^{(i)}\end{bmatrix}$$

which leads to $\mathbf{Z}_S\mathbf{v}_S^{(i)} + M\text{diag}(y)\mathbf{v}_T^{(i)} = \beta_i\mathbf{v}_S^{(i)}$ and $\text{diag}(y)\mathbf{v}_T^{(i)} = \beta_i\mathbf{v}_T^{(i)}$, from which we can conclude that $\mathbf{v}_S^{(i)} = \beta_i(\beta_i I - \mathbf{Z}_S)^{-1}M\mathbf{v}_T^{(i)}$.

So,

$$\mathbf{v}^{(i)} = \begin{bmatrix}(I - \beta_i^{-1}\mathbf{Z}_S)^{-1}M\mathbf{v}_T^{(i)} \\ \mathbf{v}_T^{(i)}\end{bmatrix}.$$

$\text{diag}(y)$ is a diagonal matrix of exponentiated terminal rewards, thus $\text{diag}(y)\mathbf{v}_T^{(i)} = \beta_i\mathbf{v}_T^{(i)}$ tells us $\mathbf{v}_T^{(i)} = e_\tau$ (one-hot vector over terminal states) and $\beta_i = e^{\frac{\mathcal{R}(\tau)}{\lambda}}$ for some $\tau \in \mathcal{T}$. If terminal rewards are constant, we have the same β_i for all eigenvectors of $\text{diag}(y)$. Eigenvalues arising instead from $\mathbf{Z}_S$ must have $\mathbf{v}_T^{(i)} = \mathbf{0}$, since under our assumption no eigenvalue of $\mathbf{Z}_S$ is also an eigenvalue of $\text{diag}(y)$, and so the only solution to $\text{diag}(y)\mathbf{v}_T^{(i)} = \beta_i\mathbf{v}_T^{(i)}$ in this case is the zero vector.

Under the assumption $e^{\mathcal{R}(\tau^*)/\lambda} > \rho(\mathbf{Z}_S)$, the largest eigenvalue of the full DR is $\beta^* = e^{\mathcal{R}(\tau^*)/\lambda}$, with $\mathbf{v}_T^* = e_{\tau^*}$. This same assumption guarantees convergence of the Neumann series

$$(I - \beta^{*-1}\mathbf{Z}_S)^{-1} = \sum_{k=0}^{\infty}\beta^{*-k}\mathbf{Z}_S^k$$

allowing us to evaluate the top eigenvector of the full DR as

$$\begin{aligned}\mathbf{v}^* &= \begin{bmatrix}\mathbf{v}_S^* \\ \mathbf{v}_T^*\end{bmatrix} \\ &= \begin{bmatrix}e^{\frac{\mathcal{R}(\tau^*)}{\lambda}}(e^{\frac{\mathcal{R}(\tau^*)}{\lambda}}I - \mathbf{Z}_S)^{-1}Me_{\tau^*} \\ e_{\tau^*}\end{bmatrix} \\ &= \begin{bmatrix}(I - e^{-\frac{\mathcal{R}(\tau^*)}{\lambda}}\mathbf{Z}_S)^{-1}M_{\cdot,\tau^*} \\ e_{\tau^*}\end{bmatrix} \\ &= \begin{bmatrix}\sum_{k=0}^{\infty}e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}}\mathbf{Z}_S^k M_{\cdot,\tau^*} \\ e_{\tau^*}\end{bmatrix}.\end{aligned}$$

■

Further analysis:

Expanding the series gives

$$\mathbf{v}^* = \begin{bmatrix}M_{\cdot,\tau^*} + e^{-\frac{\mathcal{R}(\tau^*)}{\lambda}}\mathbf{Z}_S M_{\cdot,\tau^*} + e^{-2\frac{\mathcal{R}(\tau^*)}{\lambda}}\mathbf{Z}_S^2 M_{\cdot,\tau^*} + \dots \\ e_{\tau^*}\end{bmatrix}.$$

In this expression, $\mathbf{v}_S^*$ evaluates to a vector of length S, where each entry gives us a measure relating the current non-terminal state to the highest reward terminal state. $\mathbf{v}_T^*$ identifies the highest reward terminal state among the index of all terminal states. If we have $\mathcal{R}(\tau^*) > 0$, then the exponentiated reward components in $\mathbf{v}_S^*$ are decaying with index k, and so we can interpret this as a discount factor. For applications like reward shaping, the entries of $\mathbf{v}^*$ at particular states are used. Clearly for

$\tau \in \mathcal{T}, \mathbf{v}^*(\tau) = \mathbb{1}_{\tau=\tau^*}$. For $s \in \mathcal{S}$, we have

$$\mathbf{v}^*(s) = \sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}} \sum_{s' \in \mathcal{S}} \mathbf{Z}_S^k(s, s') M(s', \tau^*).$$

Recall the definitions $\mathbf{Z}_S = (R_S^{-1} - P_S)^{-1}, M = (R_S^{-1} - P_S)^{-1} P_T$. Then,

$$\begin{aligned}\sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}} \mathbf{Z}_S^k M_{\cdot,\tau^*} &= \sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}} (R_S^{-1} - P_S)^{-(k+1)} P_{\tau^*} \\ &= \left(\sum_{k=0}^{\infty} e^{-k\frac{\mathcal{R}(\tau^*)}{\lambda}} (R_S^{-1} - P_S)^{-(k+1)} \right) P_{\tau^*}.\end{aligned}$$

This can be interpreted as a discounted reward-aware transition term over trajectories of different lengths, weighted by transitions to the most desirable terminal state.

Sufficient condition. Let $\gamma := \max_{s \in \mathcal{S}} e^{\mathcal{R}(s)/\lambda} < 1$. Since $\|D_S\|_\infty \leq \gamma$, the Neumann series gives $\|(I_S - D_S)^{-1}\|_\infty \leq \frac{1}{1-\gamma}$. With $\mathbf{Z}_S = (I_S - D_S)^{-1} R_S$ and $\|R_S\|_\infty = \gamma$,

$$\rho(\mathbf{Z}_S) \leq \|\mathbf{Z}_S\|_\infty \leq \frac{\gamma}{1-\gamma},$$

so $e^{\mathcal{R}(\tau^*)/\lambda} > \frac{\gamma}{1-\gamma}$ implies $e^{\mathcal{R}(\tau^*)/\lambda} > \rho(\mathbf{Z}_S)$.

A.4 Continuous state spaces

Let $\mathcal{S}$ be a measurable space, $\mathcal{T}$ finite, and $B(\mathcal{S})$ the Banach space of bounded measurable functions under the norm $\|\cdot\|_\infty$. Assume $\mathcal{R}$ is measurable with $\sup_{s \in \mathcal{S}} \mathcal{R}(s) < 0$, so that $\gamma := \sup_{s \in \mathcal{S}} e^{\mathcal{R}(s)/\lambda} < 1$.

Existence: For $z \in B(\mathcal{S})$, $|(D_S z)(s)| \leq e^{\mathcal{R}(s)/\lambda} \mathcal{P}^\mu(\mathcal{S} \mid s) \|z\|_\infty \leq \gamma \|z\|_\infty$. Hence $\|D_S\| \leq \gamma < 1$, $(I_S - D_S)^{-1} = \sum_{i \geq 0} D_S^i$ is bounded, and $M = \sum_{i \geq 0} D_S^i D_T$ satisfies $M = D_T + D_S M$.

Pointwise definition: All quantities are non-negative, so

$$[D_S^i D_T](s, \tau) = \int e^{\frac{\mathcal{R}(s)}{\lambda}} \mathcal{P}^\mu(ds_1 \mid s) \cdots e^{\frac{\mathcal{R}(s_{i-1})}{\lambda}} \mathcal{P}^\mu(ds_i \mid s_{i-1})\, e^{\frac{\mathcal{R}(s_i)}{\lambda}} \mathcal{P}^\mu(\tau \mid s_i),$$

and $M(s, \tau) = \sum_{i \geq 0} [D_S^i D_T](s, \tau)$. Since $\sum_\tau D_T(s, \tau) \leq \gamma$ and $\int e^{\mathcal{R}(s)/\lambda} \mathcal{P}^\mu(ds' \mid s) \leq \gamma$, induction gives $\sum_\tau [D_S^i D_T](s, \tau) \leq \gamma^{i+1}$, hence $\sum_\tau M(s, \tau) \leq \frac{\gamma}{1-\gamma}$ for every s.

Value Recovery: Let $L(s) = \sum_\tau M(s, \tau) y(\tau)$, so $\|L\|_\infty \leq \frac{\gamma}{1-\gamma} \|y\|_\infty$. Separating the $i = 0$ term gives $L = D_T y + D_S L$. Since $I_S - D_S$ is invertible, $L = (I_S - D_S)^{-1} D_T y = z$. ■

A.5 Proof of Theorem 6.1

Theorem 6.1 *Consider an LMDP of the form $\widetilde{\mathcal{L}} = \left(\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \widetilde{\mathcal{R}}, \widetilde{\mu}\right)$, with transition dependent reward $\widetilde{\mathcal{R}} : \mathcal{S} \times \mathcal{S}^+ \to \mathbb{R}$. We can always transform $\widetilde{\mathcal{L}}$ to an equivalent LMDP*

$$\mathcal{L} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mu)$$

with state-dependent rewards $\mathcal{R} : \mathcal{S}^+ \to \mathbb{R}$, such that $\mathcal{L}$ and $\widetilde{\mathcal{L}}$ have the same optimal value function.

Proof: For each $(s, s') \in \mathcal{S} \times \mathcal{S}^+$, define

$$x(s, s') = \log \mathcal{P}^{\widetilde{\mu}}(s' \mid s) + \frac{\widetilde{\mathcal{R}}(s, s')}{\lambda}.$$

Note that the Bellman optimality equation can be written as

$$z(s) = \sum_{s' \in \mathcal{S}^+} e^{x(s,s')} z(s'), \quad \forall s \in \mathcal{S}.$$

The idea is to choose μ and $\mathcal{R}$ such that

$$\log \mathcal{P}^{\mu}(s' \mid s) + \frac{\mathcal{R}(s)}{\lambda} = x(s, s')$$

for each $(s, s') \in \mathcal{S} \times \mathcal{S}^{+}$. To compute $\mathcal{R}(s)$ we can write

$$\sum_{s'} e^{x(s,s')} = e^{\mathcal{R}(s)/\lambda} \sum_{s'} \mathcal{P}^{\mu}(s' \mid s) = e^{\mathcal{R}(s)/\lambda} \iff \mathcal{R}(s) = \lambda \log \sum_{s'} e^{x(s,s')}.$$

In turn, this allows us to compute $\mathcal{P}^{\mu}(s' \mid s)$ as

$$\log \mathcal{P}^{\mu}(s' \mid s) = x(s, s') - \log \sum_{s''} e^{x(s,s'')}$$

$$\iff \quad \mathcal{P}^{\mu}(s' \mid s) = \frac{e^{x(s,s')}}{\sum_{s''} e^{x(s,s'')}} = \frac{e^{\widetilde{\mathcal{R}}(s,s')/\lambda} \mathcal{P}^{\widetilde{\mu}}(s' \mid s)}{\sum_{s''} e^{\widetilde{\mathcal{R}}(s,s'')/\lambda} \mathcal{P}^{\widetilde{\mu}}(s'' \mid s)}.$$

Here we have to assume that there exists a policy μ that realizes the transition kernel $\mathcal{P}^{\mu}$.

Under the assumption that $\tilde{\mathcal{R}}(s, s') < 0$ for each $(s, s') \in \mathcal{S} \times \mathcal{S}^{+}$, we obtain

$$\mathcal{R}(s) = \lambda \log \sum_{s'} e^{x(s,s')} = \lambda \log \sum_{s'} \mathcal{P}^{\widetilde{\mu}}(s' \mid s) e^{\widetilde{\mathcal{R}}(s,s')/\lambda} < \lambda \log \sum_{s'} \mathcal{P}^{\widetilde{\mu}}(s' \mid s) = 0.$$

Hence $\mathcal{L}$ satisfies $\mathcal{R}(s) < 0$ for each $s \in \mathcal{S}$. ■

A.6 Proof of Theorem 6.2

Theorem 6.2 *Consider an LMDP of the form $\bar{\mathcal{L}} = (\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \bar{\mathcal{R}}, \mu)$, where the reward is given by $\bar{\mathcal{R}} : \mathcal{S}^{+} \times \mathcal{A} \to \mathbb{R}$. Let q be the optimal action-value function satisfying Bellman equations of the form $q = \bar{\mathcal{R}} + \lambda \log \mathbb{E}_{s'}\big[\sum_{a'} \mu(a'|s') e^{q(s',a')/\lambda}\big]$. Now we define the state-action exponentiated values as $\bar{z}(s, a) = e^{\frac{q(s,a)}{\lambda}}$ for $s \in \mathcal{S}$ and $\bar{z}(\tau, a) = e^{\frac{\bar{\mathcal{R}}(\tau,a)}{\lambda}}$ for $\tau \in \mathcal{T}$. Let $\bar{z} \in \mathbb{R}^{S \times A}$ be the vector of exponentiated non-terminal action-values and $\bar{y} \in \mathbb{R}^{T \times A}$ its terminal equivalent. Then, we can compute the state-action TR $\bar{M}$ under this formulation and use it to recover action-values through the relation $\bar{z} = \bar{M}\bar{y}$.*

Proof: The induced distribution $\bar{\mathcal{P}}^{\mu} : \mathcal{S}^{+} \times \mathcal{A} \to \Delta(\mathcal{S}^{+} \times \mathcal{A})$ is computed as $\bar{\mathcal{P}}^{\mu}(s', a' \mid s, a) = \mathcal{P}(s' \mid s, a) \cdot \mu(a' \mid s')$, with $\bar{\mathcal{P}}^{\mu}(s', a' \mid \tau, a) = 0$ for all $\tau \in \mathcal{T}$.

$\bar{R}_{SA} \in \mathbb{R}^{SA \times SA}$ is a diagonal exponentiated reward matrix for non-terminal, while $\bar{P}_{SA} \in \mathbb{R}^{SA \times SA}$ and $\bar{P}_{TA} \in \mathbb{R}^{SA \times TA}$ encode transitions.

We can partition the Bellman equations as in the state-dependent case: For all $s \in \mathcal{S}$ and $a \in \mathcal{A}$,

$$\begin{aligned}
\bar{z}(s, a) &= e^{\frac{\bar{\mathcal{R}}(s,a)}{\lambda}} \sum_{s', a' \in \mathcal{S}^{+} \times \mathcal{A}} \bar{\mathcal{P}}^{\mu}(s', a' \mid s, a) \bar{z}(s', a') \\
&= e^{\frac{\bar{\mathcal{R}}(s,a)}{\lambda}} \sum_{s', a' \in \mathcal{S} \times \mathcal{A}} \bar{\mathcal{P}}^{\mu}(s', a' \mid s, a) \bar{z}(s', a') + e^{\frac{\bar{\mathcal{R}}(s,a)}{\lambda}} \sum_{\tau, a' \in \mathcal{T} \times \mathcal{A}} \bar{\mathcal{P}}^{\mu}(\tau, a' \mid s, a) e^{\frac{\bar{\mathcal{R}}(\tau,a')}{\lambda}}.
\end{aligned}$$

This leads to

$$\bar{z} = \bar{R}_{SA}\bar{P}_{SA}\bar{z} + \bar{R}_{SA}\bar{P}_{TA}\bar{y} \iff \bar{z} = (\bar{R}_{SA}^{-1} - \bar{P}_{SA})^{-1}\bar{P}_{TA}\bar{y} = (I_{SA} - \bar{R}_{SA}\bar{P}_{SA})^{-1}\bar{R}_{SA}\bar{P}_{TA}\bar{y}.$$

We introduce equivalent compact matrices to the state-dependent case: $\bar{D}_{SA} = \bar{R}_{SA}\bar{P}_{SA}$ and $\bar{D}_{TA} = \bar{R}_{SA}\bar{P}_{TA}$.

Now,

$$\begin{aligned}
\bar{M} &= (\bar{R}_{SA}^{-1} - \bar{P}_{SA})^{-1}\bar{P}_{TA} \\
&= (I_{SA} - \bar{R}_{SA}\bar{P}_{SA})^{-1}\bar{R}_{SA}\bar{P}_{TA} \\
&= (I_{SA} - \bar{D}_{SA})^{-1}\bar{D}_{TA}
\end{aligned}$$

is the *state-action* TR, and we can compute $\bar{z} = \bar{M}\bar{y}$. ■

A.7 Proof of Theorem 6.3

Theorem 6.3 *Let z_i be the value of the base LMDP $\mathcal{L}_i$ with terminal values $z_i(\tau) = \mathbb{1}_{\tau=\tau_i}$ (i.e. $\mathcal{R}_i(\tau_i) = 0$, $\mathcal{R}_i(\tau) = -\infty$ otherwise). Then $M^+(\cdot, \tau_i) = z_i$ for $i = 1, \ldots, T$, and hence the TR can be used to perform compositionality via the relation $z(s) = \sum_{i=1}^{T} z(\tau_i) M^+(s, \tau_i)$.*

Proof:

The TR matrix M can be interpreted as a function $M : \mathcal{S} \times \mathcal{T} \to \mathbb{R}$. Then, for $s \in \mathcal{S}$ and $\tau_i \in \mathcal{T}$, our recursion $M = D_T + D_S M$ can be written element-wise as

$$M(s, \tau_i) = e^{\frac{\mathcal{R}(s)}{\lambda}} (\mathcal{P}^{\mu}(\tau_i \mid s) + \sum_{s' \in \mathcal{S}} \mathcal{P}^{\mu}(s' \mid s) M(s', \tau_i)).$$

This is not quite the Bellman equation in M, because of the separation of $\mathcal{P}^{\mu}$ into P_S and P_T.

Returning to the previous recursion and using the extended domain M^+ defined above, we can write:

$$\begin{aligned} M(s, \tau_i) &= e^{\frac{\mathcal{R}(s)}{\lambda}} (\mathcal{P}^{\mu}(\tau_i \mid s) + \sum_{s' \in \mathcal{S}} \mathcal{P}^{\mu}(s' \mid s) M(s', \tau_i)) \\ &= e^{\frac{\mathcal{R}(s)}{\lambda}} (\sum_{\tau' \in \mathcal{T}} \mathcal{P}^{\mu}(\tau' \mid s) M^+(\tau', \tau_i) + \sum_{s' \in \mathcal{S}} \mathcal{P}^{\mu}(s' \mid s) M^+(s', \tau_i)) \\ &= e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s' \in \mathcal{S}^+} \mathcal{P}^{\mu}(s' \mid s) M^+(s', \tau_i). \end{aligned}$$

This is not a valid recursion, since the LHS uses M while the RHS uses M^+. For $s \in \mathcal{S}$ we have $M(s, \tau_i) = M^+(s, \tau_i)$, and so it is trivial to replace the LHS. This gives us the following expression for non-terminal states:

$$M^+(s, \tau_i) = e^{\frac{\mathcal{R}(s)}{\lambda}} \sum_{s' \in \mathcal{S}^+} \mathcal{P}^{\mu}(s' \mid s) M^+(s', \tau_i).$$

We now examine the boundary conditions, i.e. states $\tau' \in \mathcal{T}$. By definition, we have $M^+(\tau', \tau_i) = \mathbb{1}_{\tau'=\tau_i}$.

$z_i(s)$, the exponentiated value function for base LMDP i, must satisfy the exponentiated Bellman equation for non-terminal states:

$$z_i(s) = e^{\frac{\mathcal{R}_i(s)}{\lambda}} \sum_{s' \in \mathcal{S}^+} \mathcal{P}^{\mu}(s' \mid s) z_i(s').$$

Since the base LMDPs share the same non-terminal rewards as the overarching LMDP, $\mathcal{R}_i(s) = \mathcal{R}(s)$ for all $s \in \mathcal{S}$. Therefore, $z_i(s)$ obeys the same recursion as $M^+(s, \tau_i)$.

Under the assumed reward structure, we have that the terminal values $z_i(\tau) = e^{\mathcal{R}_i(\tau)/\lambda} = \mathbb{1}_{\tau=\tau_i}$

Because $M^+(s, \tau_i)$ and $z_i(s)$ satisfy identical recursive equations for non-terminal states and share identical boundary conditions at terminal states, and since the Bellman equations have a unique solution, we can conclude that $M^+(s, \tau_i) = z_i(s)$ for all $s \in \mathcal{S}^+$ and $i = 1 \ldots T$.

We have that $z(s) = \sum_{i=1}^{T} z(\tau_i) z_i(s)$. By direct substitution, we arrive at the final result:

$$z(s) = \sum_{i=1}^{T} z(\tau_i) M^+(s, \tau_i).$$

■

B Additional Experimental Details

Experiments are adapted from the work of Tse et al. [25]. Details regarding the relevant settings, as well as hyperparameter configurations relating to the DR and SR, can be found in Appendix C of their paper. The code they used is publicly available at `https://github.com/httse9/Reward-Aware-Proto-Representations`, and we use this as a basis to implement our TR experiments. An independent repository for this paper will be made available at a later stage. Tse et al. [25] themselves adapt their Count-Based Exploration experiments and implementation from Machado et al. [14], with code for this work found at `https://github.com/mcmachado/count_based_exploration_sr`.

Experimental settings and hyperparameters are thus used as reported for the SR, the DR, and any 'neutral' baselines—e.g. the random walk in option discovery, SARSA without exploration bonus in count based exploration, learning with no shaped reward in reward shaping. We thus focus on details directly relevant to the TR experiments within this framework. Number of seeds used to generate confidence intervals are mentioned in the plot captions of the main text. Where this is fewer than what is reported by Tse et al. [25] (or Machado et al. [14]), all experiments are run independently from scratch over the same seeds and numbers of seeds. Note that in general, though we use reported hyperparameters, experiments are run from scratch and we do not use stored results. Across the experiments, we do not perform comparisons under unequal conditions. It was necessary to modify the setting of one experiment in order to accommodate the TR—option discovery. We discuss this in the following subsection.

B.1 Option discovery

As acknowledged in the main text, the core limitation of the TR is its dependence on explicitly defined terminal states. Tse et al. [25] conduct their option discovery experiments in environments without terminal states, which naturally prevents usage of the TR. Our option discovery experiment is conducted in a Four Rooms environment with four goal states, each of which inducing termination when reached. The environment layout and goal placement is identical to the multi-goal Four Rooms used by Tse et al. [25], but the original version treats these goals as non-terminal. We make another variation to this environment in that the goal rewards are differently weighted—+4 in the top left, +3 in the bottom left, +2 in the bottom right, and +1 in the top right. The original version uses a reward of 0 for all goals. This change is to induce meaningful difference across the four TR variants.

The hyperparameter search space for option discovery follows Table 2 of Tse et al. [25], and is reproduced in Table 2. Best hyperparameters are selected by maximising a composite score of normalised state visitation percentage and normalised average reward, evaluated over the first 1500 steps across a minimum of 5 independent seeds. The Q-learning hyperparameter search sweeps over initialisation values $Q_{\text{init}} \in \{-10, 0\}$, exploration rates $\varepsilon \in \{0.15, 0.2\}$, and step sizes $\in \{0.01, 0.3, 1.0\}$. The same combination proved optimal across all methods. Best hyperparameters found for each method are reported in Tables 3 and 4.

Table 2: Hyperparameter search space for option discovery.

Name	Description	Values
p_{option}	Probability of selecting an option	$[0.01, 0.05, 0.1]$
N_{learn}	Number of sweeps through dataset to learn the representation	$[1, 10, 100]$
α	Step size for learning the representation	$[0.01, 0.03, 0.1]$
N_{option}	Number of latest options to keep	$[1, 8, 1000]$

Table 3: Best hyperparameters for all methods on `fourrooms_multigoal`.

Method	p_{option}	N_{learn}	α	N_{option}
CEO (SR)	0.1	1	0.01	8
RACE (DR)	0.05	1	0.03	1
TR-Cyclical	0.05	1	0.01	8
TR-Random	0.1	10	0.03	1
TR-Highest	0.1	1	0.01	1
TR-All	0.1	1	0.01	8

Table 4: Best Q-learning hyperparameters on `fourrooms_multigoal`.

Method	Q_{init}	ε	Step size
Pure Q	-10	0.2	0.3
CEO+Q (SR)	-10	0.2	0.3
RACE+Q (DR)	-10	0.2	0.3
TR-Cyclical+Q	-10	0.2	0.3
TR-Random+Q	-10	0.2	0.3
TR-Highest+Q	-10	0.2	0.3
TR-All+Q	-10	0.2	0.3

B.2 Count based exploration

Hyperparameters for Sarsa, Sarsa+SR, and Sarsa+DR are used as reported by Machado et al. [14] and Tse et al. [25] respectively, and are reproduced in Tables 5–7. For Sarsa+TR, we perform a random search over 80 configurations sampled from the space given in Table 8. Best hyperparameters are reported in Table 9.

Table 5: Hyperparameters for Sarsa, reproduced from Machado et al. [14].

Environment	α	ε	γ
RiverSwim	0.37	0.12	0.95
SixArms	0.43	0.01	0.95

Table 6: Hyperparameters for Sarsa+SR, reproduced from Machado et al. [14].

Environment	α	η	β	γ_ϕ
RiverSwim	0.1	0.5	10000	0.5
SixArms	0.5	0.25	10000	0.5

Table 7: Hyperparameters for Sarsa+DR, reproduced from Tse et al. [25].

Environment	α	η	β	λ	transform
RiverSwim	0.5	0.25	100	1.0	$\log(\|\mathbf{x}\|_2)$
SixArms	0.5	0.01	0.1	1.5	$\log(\|\mathbf{x}\|_2)$

Table 8: Hyperparameter search space for Sarsa+TR.

Name	**Description**	**Values**
α	Step size for Q-value updates	$[0.01, 0.1, 0.25, 0.5]$
η	Step size for TR updates	$[0.01, 0.1, 0.25, 0.5]$
β	Intrinsic reward scaling factor	$[0.1, 1, 10, 100, 1000, 10000]$
λ	Exponential weighting of cumulant	$[1.0, 1.5, 2.0]$
γ	TR discount factor	$[0.95, 1.0]$
transform	Transformation applied to TR norm	$[\log \|\cdot\|_1,\ \log \|\cdot\|_2]$

Table 9: Best hyperparameters for Sarsa+TR.

Environment	α	η	β	λ	γ	transform
RIVERSWIM	0.25	0.01	10	2.0	0.95	$\log(\|\mathbf{x}\|_2)$
SIXARMS	0.01	0.5	10	1.5	0.95	$\log(\|\mathbf{x}\|_2)$

C STATISTICAL SIGNIFICANCE

We report 95% confidence intervals throughout. These capture randomness across independent runs, where the primary source of randomness is random exploration via ε-greedy policies. In option discovery and transfer experiments, randomness also arises from the stochastic environment dynamics across seeds. Following Tse et al. [25], we compute 95% confidence intervals as $\pm 1.96 \frac{\hat{\sigma}}{\sqrt{N}}$, where $\hat{\sigma}$ is the sample standard deviation. For experiments conducted over $N \geq 50$ seeds this approximation is well-justified; for smaller N (as in option discovery, where $N = 10$) it should be interpreted as approximate.

D COMPUTE RESOURCES

All experiments were conducted on a standard laptop CPU. Run times are modest, but can become more prohibitive with higher number of seeds and parameter configurations. This is especially the case for option discovery experiments.