Symmetry and Generalization in Local Learning of Predictive Representations

Janis Keck*[1,2,3,], Caswell Barry[4], Christian F. Doeller[2,3,5], and Jürgen Jost[1,3,6,7]

[1]Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany
[2]Max Planck Institute for Human Cognitive and Brain Sciences, Leipzig, Germany
[3]Max Planck School of Cognition
[4]Department of Cell and Developmental Biology, University College London, London, WC1E 6BT, UK
[5]Kavli Institute for Systems Neuroscience and Jebsen Centre for Alzheimer's Disease, Norwegian University of Science and Technology, Trondheim, Norway
[6]ScaDS.AI - Center for Scalable Data Analytics and Artificial Intelligence, Leipzig, Germany
[7]Santa Fe Institute for the Sciences of Complexity, Santa Fe, New Mexico, USA

May 27, 2024

Abstract

It is an increasingly accepted view that the representations which the brain generates are not merely descriptive of the current state of the world; rather, representations serve a predictive purpose. In spatial cognition, the Successor Representation (SR) from reinforcement learning provides a compelling candidate of how such predictive representations are used to encode space, in particular, hippocampal place cells are assumed to encode the SR. Here, we investigate how varying the temporal symmetry in learning rules influences those representations. To this end, we use a simple local learning rule which can be made insensitive to the temporal order. We analytically find that a symmetric learning rule results in a successor representation under a symmetrized version of the experienced transition

*Corresponding author: `janis.keck@maxplanckschools.de`

structure. We then apply this rule to a two-layer neural network model loosely resembling hippocampal subfields CA3 - with a symmetric learning rule and recurrent weights - and CA1 - with an asymmetric learning rule and no recurrent weights. Here, when exposed repeatedly to a linear track, CA3 neurons in our model show less shift of the centre of mass than those in CA1, in line with existing empirical findings - an effect which is not observed using an asymmetric learning rule. We furthermore investigate the functional benefit of such representations in simple RL navigation tasks. Here, we find that using a symmetric learning rule yields representations which afford better generalization, when a model is probed to navigate to a new target without relearning the SR. This effect is reversed when the state space is not symmetric anymore. Thus, our results hint at a potential benefit of the inductive bias afforded by symmetric learning rules in areas employed in spatial navigation, where there naturally is a symmetry in the state space. In conclusion, we expand the SR theory of hippocampus by including symmetry in SR learning, which might yield an advantageous inductive bias for learning in space.

1 Introduction

The hippocampus and its adjacent sub- and neocortical regions are widely believed to form both a crucial part in the acquisition and storage of memory, as well as the encoding of spatial and navigational variables in the form of spatially stable neural responses. (Scoville and Milner, 1957; O'Keefe and Nadel, 1978; Eichenbaum et al., 1999; Squire et al., 2004; Hafting et al., 2005).

It is an increasingly popular assumption that the representations that the brain generates in general, and in particular for space and memory, are not merely descriptive of the current state of the world, post-dictions of events or places just passed. Rather, it is believed that a predictive representation is learned, such that the objective is to infer future states of the world from one's experience (Rao and Ballard, 1999; Friston, 2002; Stachenfeld et al., 2017; Russek et al., 2017; Behrens et al., 2018).

One framework that has extensively been used to describe this objective on the algorithmic level comes from reinforcement learning. The so called 'successor representation' (SR), or more broadly 'successor features' (SF) are a generalization of the well known value function, and are essentially a conditional expectation: Given the current state, they encode a (weighted) expectation of future values of a given function of the states of the world (Dayan, 1993; Barreto et al., 2017). If that function is simply an indicator of the states, then one obtains the SR, which thus roughly encodes how often states will be visited in the future.

Originally, the SR was proposed as an intermediate between 'model-based' and 'model-free' reinforcement learning (Momennejad et al., 2017; Gershman, 2018), allowing the storage of certain information about the transition structure under a given policy - hence, affording some generalization to different reward structures - while still being possible to learn with a efficient TD-learning algo-

rithm (Dayan, 1993; Russek et al., 2017). Later work has also used the SR for different objectives such as option discovery (Machado et al., 2017b) and reward free exploration (Yu et al., 2023).

In the hippocampal navigation literature, the successor representation view has been influential because apart from fitting well with the more general predictive brain hypothesis, it could explain non-trivial effects of place cells that had been previously observed, for example the skewing of place fields in direction of travel or the non-extension of place-fields through obstacles in the environment (Stachenfeld et al., 2017). Furthermore, SR theory yielded an algorithmic explanation for grid cells as an eigendecomposition of place-cell structure, which could also be connected to efficient neurally plausible navigation (Dordek et al., 2016; Corneil and Gerstner, 2015; De Cothi and Barry, 2020; De Cothi et al., 2022).

Despite the immanent success of the SR theory to explain neural data on an algorithmic level, there has been considerably less work dedicated to providing a mechanism through which the SR should be learned using biologically plausible learning rules (Vértes and Sahani, 2019). Recently, this question has been tackled by the community: Two recent papers (George et al., 2023b; Bono et al., 2023) used feedforward networks to learn synaptic weights that compute successor features from their inputs. (George et al., 2023b) focuses on the predictiveness afforded by the theta cycle together with a compartment-neuron learning rule, while (Bono et al., 2023) uses spiking neural networks and a STDP-like rule. On the other hand, (Fang et al., 2023) used a recurrent neural network, to learn successor features directly in the activities of the recurrently connected neurons.

Anatomically, the latter approach can be linked to plasticity occurring at the recurrent synapses of CA3, while the former approach maps to the feedforward synapses from CA1 (Knierim, 2015). Both areas are known to show a considerable proportion of place cells (Leutgeb et al., 2004), hence both are indeed candidate regions to encode successor representations. However, it has been suggested that different learning rules might be in place at the respective synapses: the feedforward synapse from CA3 to CA1 is the classical location for the study of STDP, that is plasticity which requires presynaptic increased activity to precede postsynaptic increased activity (Bi and Poo, 1998; Markram et al., 2011). On the other hand, recent work has identified a regime in which recurrent CA3 synapses get strengthened if pre- and postsynaptic increased activity are close in time, regardless of the temporal order - and computationally linked a symmetric learning rule to benefit in memory storage of a recurrent network (Mishra et al., 2016).

Here, we want to investigate the effect of such symmetric learning rules on the construction of predictive representations. That is, we aim to understand whether using a learning rule insensitive to the temporal order of the inputs learns different successor representations - and which (dis-)advantages it yields.

To this end, we first construct a model which has both a recurrent and a feedforward component, reminiscent of the architecture of CA, and study the successor features that are learned using a local learning rule. Thereby we

extend the earlier work which focused on learning in a single layer to learning at multiple levels. This extension is rather straightforward and results in both layers learning successor features based on their respective inputs.

We then find that by changing the learning rule, the representations also undergo a similar modification: In the symmetric setting, instead of encoding future expectations under the current true policy of the agent, successor features under a symmetrized version of the transition probabilities are learned, while an asymmetric rule learns the 'true' successor features. We then contrast the utility of the respective representations in a reinforcement learning setting. There, we find that a symmetric learning rule yields benefits for generalization in navigational tasks, where the symmetry of the state space can be exploited, while an asymmetric learning rule is more advantageous for generalization in asymmetric state spaces. We conclude that implementing both an asymmetric and a symmetric learning rule might yield complementary representations.

2 Results

2.1 Successor Representations

The successor representation and the more general successor features describe future expectations of a quantity, conditional on the current state of the world. They are most easily defined in the following setting: Assume the environment of an agent/animal consists of a set of states $\mathcal{S}$. The states of the world are changing according to a time homogeneous Markov chain, denoted $s_t \in \mathcal{S}$. Then for any feature/observation function

$$\phi : \mathcal{S} \to \mathbb{R}^m \tag{1}$$

one can define an expectation of weighted, cumulative future values of that function, given the current state:

$$SF_\phi(s) = \mathbb{E}\left[\sum_{k=0} \gamma^k \phi(S_{t+k}) \middle| S_t = s\right]. \tag{2}$$

The weighting factor $\gamma \in [0, 1)$ puts relatively more importance on proximal times. In the case that ϕ is an injective function, that is for every state of the world there is an unique observation value, it makes sense to define the 'successor representation' (SR)

$$SR_\phi(\nu) = \mathbb{E}\left[\sum_{k=0} \gamma^k \phi(S_{t+k}) \middle| \phi(S_t) = \nu\right]. \tag{3}$$

We use this terminology here in a little more generality than is usual, but it can be seen that one important special case leads to what is usually called SR: Using indicator vectors $\phi(s) = e_{s'}(s)$, for some state s', one obtains

$$SR_{e_{s'}} = \sum_k \gamma^k P^k e_{s'} = (Id - \gamma P)^{-1}_{\cdot s'}, \tag{4}$$

with P the transition matrix of the Markov Chain. The matrix $(Id - \gamma P)^{-1}$ is widely referred to as the successor representation, so our definition encompasses this special case. This SR of indicator vectors thus gives a weighted sum of expected future visitation probabilities of states, and it is this expression that has originally been used to model the predictive representations that hippocampus ought to encode (Stachenfeld et al., 2017; Russek et al., 2017).

2.2 Learning Successor representations with local learning rules

We construct a simple model of two neuron population activities p_t which have dynamics of the form

$$\frac{d}{dt}p^1 = -p^1 + \sigma(\lambda_1 W^r p^1 + (1-\lambda_1)\phi^1) \tag{5}$$

$$\frac{d}{dt}p^2 = -p^2 + \sigma(\lambda_2 W^f p^1 + (1-\lambda_2)\phi^2). \tag{6}$$

Here, W^r is a recurrent connectivity matrix that feeds the activity of the first population back into itself, while W^f is feedforward matrix, which encodes how activity of the first population is fed into the second. This architecture of one population of highly recurrently connected neurons feeding into second one with little recurrent connectivity is reminiscent of hippocampal subfields CA3, CA1 respectively (Knierim, 2015). The two populations obtain additional external inputs ϕ^1, ϕ^2, which might represent the input to $CA3$ via mossy fibers, or directly through the perforant path, and the input to $CA1$ from EC through the latter, respectively. In the simplest case, these inputs are just indicator-functions for particular states, that is $\phi_i(s) = \delta(s = s_i)$. A more realistic shape, which we employ in our experiments in section 2.5, might be given by Gaussian inputs of the form $\phi_i(s) = \exp\left(-\frac{\|s-\mu_i\|}{2\sigma}\right)$.

For analysis, we assume that the activation function σ is the identity (i.e. the system is linear), and that the population vectors take the equilibrium values of the above dynamics, that is

$$p^1 = (1-\lambda_1)(\mathrm{Id} - \lambda_1 W^r)^{-1}\phi^1 \tag{7}$$

$$p^2 = \lambda_2 W^f p^1 + (1-\lambda_2)\phi^2. \tag{8}$$

We then define a learning rule for the synaptic weights, using these equilibrium values. Hence, we implicitly assume that neural dynamics happen on a timescale τ_p which is way quicker than that of learning, τ_W. Taking the equilibrium values is the limit case which simplifies analysis, but in practice one can also take $\tau_p << \tau_W$ and simply update activities and weights concurrently. The learning rule we use is a slight modification of the learning rule used in Fang et al. (2023). In particular, we use the same general learning rule for both recurrent and feedforward weights, only varying certain parameters. Let $p^{post,i}, p^{pre,i}$ be the activity of the i-th post/pre-synaptic neuron respectively. We then update

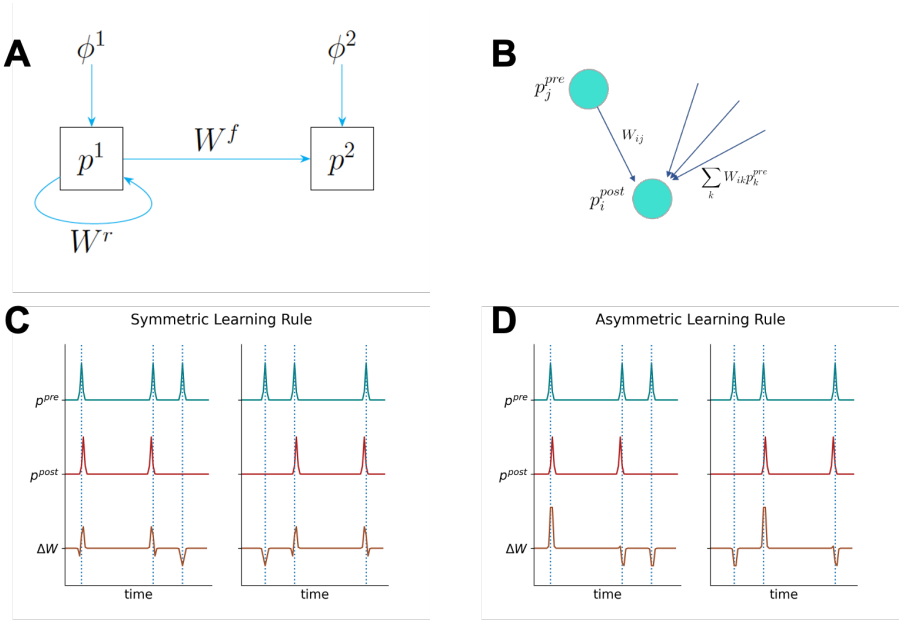

Figure 1: Model and learning rule. **(A)** Cartoon depiction of the model we are using in the main text. A recurrently connected population of neurons p_1, putatively $CA3$, receives external input ϕ_1, putatively from dentate gyrus or entorhinal cortex(EC). It projects to another population p_2, which receives input ϕ_2. The latter could be $CA1$ receiving input again from EC. Note that there are no recurrent connections in the second layer and no backwards connections. **(B)** Quantities relevant for the update of synapse W_{ij}: pre- and postsynaptic activities, as well as the sum of the total input to the postsynaptic neuron through the synapses W. **(C), (D)** Invariance of learning rules with respect to temporal order. We plot synaptic weight change of a single synapse in a setup with a single pre- and postsynaptic neuron, respectively. The right column has the same pre- and postsynaptic activities as the left column, only in reverse order. In **(C)**, the learning rule with parameters $\alpha = 1, \beta = 0$ is used, while in **(D)** $\alpha = \beta = \frac{1}{2}$. Only in the latter the synaptic weight changes are preserved (in reverse order), while in **(C)**, postsynaptic activity before presynaptic activity leads to a net weight decrease. Note that in this illustrative example W is fixed, in reality, network dynamics and weights would influence each other and lead to more complex changes.

the weight from the $j-$th presynaptic neuron to the i-th postsynaptic neuron via

$$\Delta W_{ij} = \alpha \left(p_{t+1}^{post,i} - \sum_k W_{ik} p_t^{pre,k} \right) p_t^{pre,j} + \beta \left(p_t^{post,i} - \sum_k W_{ik} p_{t+1}^{pre,k} \right) p_{t+1}^{pre,j}. \tag{9}$$

The update rule contains terms of the form $p^{post,i} p^{pre,j}$, which are simply Hebbian terms. The other summands perform a subtractive normalization: they subtract the total overall input to a post-synaptic neuron, such that only activity exceeding this input will actually be considered positive. This term has been interpreted as a decorrelative term by (Fang et al., 2023) - in total, the learning rule can be understand as a predictive coding rule approximating a conditional expectation operation, as we explain in appendix A. In matrix-vector notation the update rule reads

$$\Delta W = \alpha \left(p_{t+1}^{post} - W p_t^{pre} \right) p_t^{pre\,T} + \beta \left(p_t^{post} - W p_{t+1}^{pre} \right) p_{t+1}^{pre\,T}. \tag{10}$$

The parameters $\alpha, \beta \in \mathbb{R}$ control the sensitivity of this update to the order of the activity, put differently, the temporal symmetry of the rule: If we write our synaptic weight change as $\Delta_W(W, p_{t+1}^{post}, p_t^{post}, p_{t+1}^{pre}, p_t^{pre})$, then for parameters $\alpha = \beta$ we obtain a learning rule that is invariant under a reversal of time, that is

$$\Delta_W \left(W, p', p, q', q \right) = \Delta_W \left(W, p, p', q, q' \right), \tag{11}$$

while for the original learning rule with $\alpha = 1, \beta = 0$, no such relation holds - see also Figure 1. For $\alpha = -\beta$ we would obtain a rule that is antisymmetric in this sense - it turns out however that this would yield unstable dynamics.

2.3 Network learns successor representation and successor features

Before exhibiting the representations that are learned under the modified learning rule, it might be helpful understanding which representations a two layer-network as the above learns with the simplest choice of parameters ($\alpha = 1, \beta = 0$). Let us assume the has been exposed extensively to features ϕ^1, ϕ^2 under the same random walk with transition probabilities P, such that the synaptic weights could converge. Then, as we show in appendix A and appendix D we find that the activities in the network, when presented with inputs $\tilde{\phi_1}, \tilde{\phi_2}$, compute successor representations/features: They take the equilibrium values

$$p^1 = (1 - \lambda_1) SF_{\phi^1}(\tilde{\phi^1}) = (1 - \lambda_1) \sum_{k=0}^{\infty} \mathbb{E}[\phi^1(S_{t+k}) | \phi^1(S_t) = \tilde{\phi_1}] \tag{12}$$

$$p^2 = (1 - \lambda_2) \left(\tilde{\phi^2} + \sum_{k=0}^{\infty} \mathbb{E}[\phi^2(S_{t+k}) | SF_{\phi^1}(S_t) = SF_{\phi^1}(\tilde{\phi^1})] \right). \tag{13}$$

In words, this means that the first, recurrent layer computes the weighted cumulative sum of the predicted values of the feature ϕ^1 it was trained on, but

given the possibly new feature $\tilde{\phi}^1$ - one could see this a form of pattern completion, but with a predictive component. Similarly, the second layer computes predictions of ϕ^2, only that these predictions themselves depend on those predictive representations passed to it from the first layer. In particular, when the inputs are the same as the model was trained on, and the maps ϕ_i are injective (i.e., sufficiently rich features exist for the environment), then these equations simplify to

$$p_1 = (1 - \lambda_1) SR_{\phi_1} \tag{14}$$

$$p_2 = (1 - \lambda_2) SR_{\phi_2}. \tag{15}$$

That is, in this case the two layers simply learn to encode the successor representations of their respective inputs.

2.4 Influence on the representations by choice of parameters

We now proceed to ask the question "which representations would be learned depending on the choice of the parameters α, β?". It turns out that the resulting representations are still Successor representations, albeit corresponding to transition probabilities that are not necessarily faithful to those of the environmental dynamics anymore. To be precise, we define a weighted sum of transition matrices

$$P_{\alpha,\beta} := \frac{\alpha}{\alpha + \beta} P_{forward} + \frac{\beta}{\alpha + \beta} P_{backward}. \tag{16}$$

Here, $P_{forward}$ contains the transition probabilities of the actually observed process (that is, 'forward' in time), while $P_{backward}$ contains the transition probabilities of the reverse process, i.e. $p(s_t = s | s_{t+1} = s')$. We then show in appendix D that under suitable conditions, with the learning rule defined above, the model is able to learn the successor representations under $P_{\alpha,\beta}$. This entails that the neuronal population activities at convergence yield

$$p_t^i = SR_{\phi^i}^{P_{\alpha^i,\beta^i}}(\phi_t^i), \tag{17}$$

with α^i, β^i the respective parameters used in the learning rule. That is, in particular, under the regime $\alpha = 1, \beta = 0$, this corresponds to the 'true' successor representation, while under a symmetric regime, the transition probabilities forward and backward in time are averaged over, that is

$$\frac{1}{2}(P_{forward} + P_{backward}). \tag{18}$$

In fact, in this case the transition probabilities are reversible in time - this is not surprising, as the learning rule was defined to be invariant under a time reversal and hence should only extract aspects of the dynamics which are reversible. We note here that such reversible dynamics have been typically assumed in the theory of SR when construing grid cells as efficient representations of the geometry of an environment through the eigenvectors of the SR - see Appendix C.

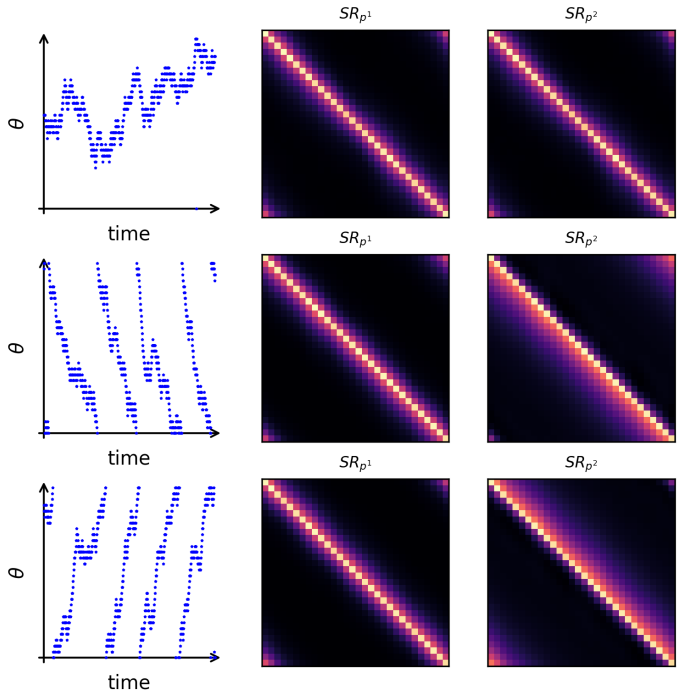

Figure 2: **Successor Representations learned in circular random walks.** We construct a circular state space with possible actions stay, move left and move right. We simulate three random walks, one where the actions are selected uniformly (first row), one where right actions are preferably selected (second row) and one where left actions are preferably selected (third row). The first column shows an example trajectory of the respective walk. The second and third column show the successor representations learned by the first and second layer of our model, using a symmetric ($\alpha = \beta = \frac{1}{2}$ and an asymmetric ($\alpha = 1, \beta = 0$) learning rule, respectively. Note how the successor representation learned with a symmetric rule does not distinguish between the policies. Here, the inputs to the cells are one-hot vectors encoding the respective states and the plotted successor representations are obtained by taking the average population activity in the respective states.

2.4.1 Activities in the model converge to theoretically obtained limits

Having theoreticallly obtained the limits of the weights and the corresponding activities, we next verified these limits empirically in simulations. In these sim-

ulations, we consider an environment with a discrete number of states $s \in \mathcal{S}$ and inputs ϕ^i which are functions of these states. In the simplest case, $\phi^i(s) = e_s$, we obtain the classical successor representation. In particular, for a symmetric learning rule we obtain a symmetrized successor representation - which shows less dependence on the policy. For example, on a circular track, the representation becomes indifferent to whether the agent is performing a clock-wise or a counter-clockwise walk (Figure 2). One might argue that a reversible representation encodes more of the geometry of the underlying state space and less of the actual dynamics (although there is still an indirect influence of the dynamics through the stationary distribution). Indeed, we show in appendix G that the symmetrized transition probabilities are always closer to a uniform policy than the unsymmetrized ones.

We also verified our theoretical results in more complex scenarios: the convergence prevails also when the features are random inputs instead of one-hot vectors, and also when the random walk is arbitrary instead of circular - see Figure 3. Additionally, we investigated the stability under the choice of parameters α, β. Here we found that it seems that the model is only stable when the positive weight is bigger (in absolute value) than the negative weight, with no convergence at the boundary case of $\alpha = -\beta$. The latter is in line with the theoretical results - note that Equation 16 is undefined in this case.

2.5 Place fields under symmetrized rule show less shift

Although both areas CA3 and CA1 show place cells, these cells exhibit different properties and dynamics (Lee et al., 2004a,b). It is well known, that on a linear or circular track place fields shift backwards opposite the direction of travel in both regions (Mehta et al., 1997; Roth et al., 2012). However, when directly comparing cell recordings from both, it has been observed for example in Dong et al. (2021) that the shift in CA3 is in general less pronounced, that is, the center of mass of these cells is more stable than in their counterparts in CA1. We hypothesized that a difference in learning rules could explain this effect. Indeed, it is not hard to see theoretically why this should be the case: Through learning, the features $\phi^i(s)$ get replaced by their successor features $SF_{\phi^i}(s)$. If there is a preferred direction of travel, then states preceding those where the feature puts a lot of mass will also have more mass, since they are predictive of the former states. If there is an asymmetry in the policy, the same will not hold true for succeeding states, hence one observes a shift towards the predecessors. If now however one has symmetric transition probabilities, then there is no directionality, hence this shift would not occur. Indeed, we provide a simple proof of this in Appendix B

We confirmed this intuition, running our two-layer model in a simple linear track where the agent repeatedly moves from the left side to the right. Indeed, we find a tendency to shift in the CA1 cells, which isn't as pronounced in the CA3 population. Qualitatively, our results match those obtained in Dong et al. (2021). Importantly, these results only hold when using the symmetrized version of the learning rule for CA3, while the asymmetric variant yields almost

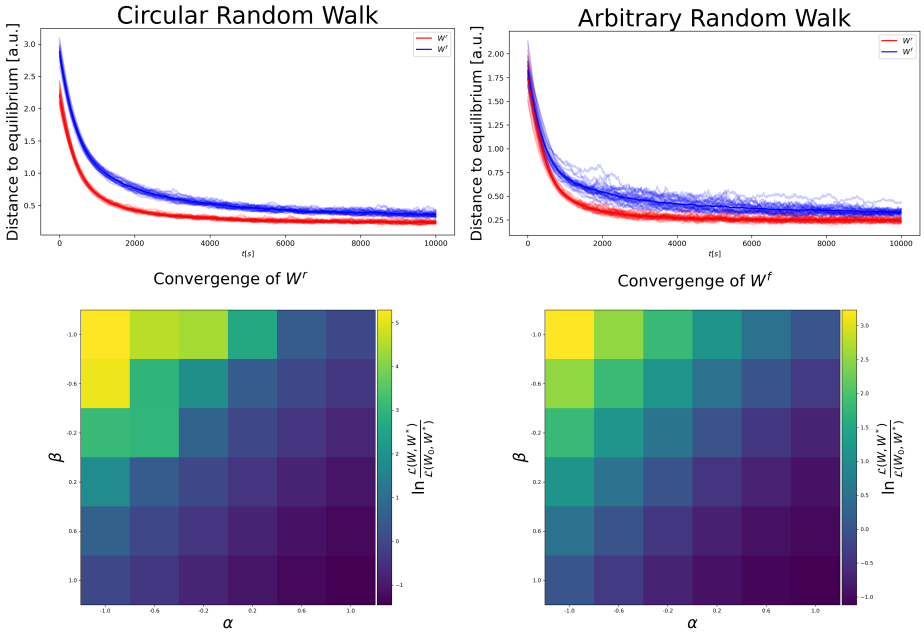

Figure 3: **Successor representations are learned for a variety of inputs, dynamics, and parameters. Top:** Convergence of recurrent (red) and feedforward (blue) matrices to their theoretical limit with random features in circular (**left**) and arbitrary (**right**) random walks. **Bottom:** Convergence of recurrent weight (**left**) and feedforward weight (**right**) for different parameters α, β. The other set of parameters is fixed to $(1, 0)$ and $(\frac{1}{2}, \frac{1}{2})$ in these experiments, respectively. In graphs, we measure convergence by the loss term $\mathcal{L}$ as explained in Methods section. In the bottom row, we compute the fraction of the loss at the final step over the initial loss and display the result in a logscale. Thus, negative values indicate converging towards the target. Note that the values on the antidiagonal are approximately 0.

no distinction.

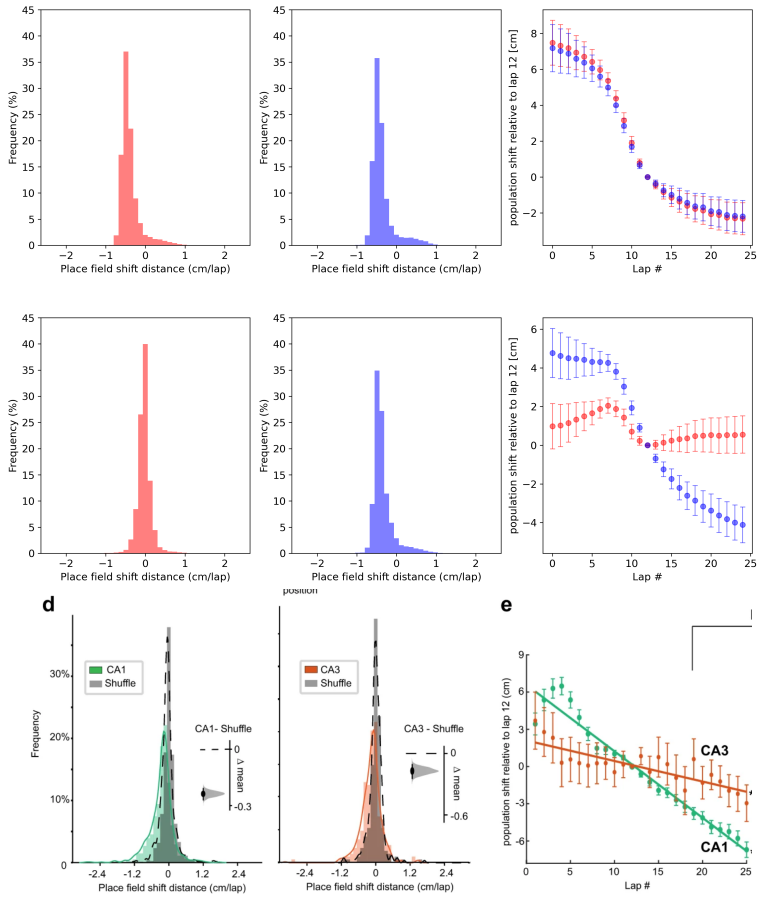

Figure 4: **Symmetric learning rule leads to more stable place fields in linear track.** We simulated an experiment with a rat repeatedly running on a linear track, similar to Dong et al. (2021). A two-layer SR network was used where the recurrent weights had a symmetric (**middle row**) or asymmetric (**top row**) learning rule. In the the symmautorefetric case, there is less shift of the centre of mass of place fields in the modelled CA3 population (*red*) than in the CA1 population (*blue*), which is not the case in the asymmetric version. Histograms show distribution of shifts comparing last five laps versus first five laps, while the rightmost plot shows shift relative to the 12-th lap. The results in the symmetric case are qualitatively similar to data (**bottom row**) from Ca^{2+} recordings of hippocampal neurons in a similar experiment - figure adapted from Dong, C., Madar, A. D., & Sheffield, M. E. (2021).

2.6 Generalization and learning with (a)-symmetric rules

Having derived the different kinds of successor representations that symmetric and asymmetric learning rules encode, we next sought to understand what the functional benefits of these representations might be.

In particular, we hypothesized that a symmetric learning rule for successor representations might be a relatively simple inductive bias that would favour learning such representations that are invariant under time reversal. This could be useful in such environments where there is a symmetry in transition structure, that is, whenever there is a mapping $\sigma : S \times A \to A$ such that

$$p(s'|s,a) = p(s|s', \sigma(s,a)). \tag{19}$$

A particularly simple example of this setting - but still likely for biological agents to encounter - is when the transition structure of the environment is deterministic and transitions in both directions between states are possible. In this case, the state space becomes a metric space and the metric a particular invariant under the symmetry - that is $d(s,s') = d(s',s)$. This then hints at a possible benefit of using a learning rule biased towards such symmetry: In a metric space, an optimal policy for navigating towards any target depends only on the metric - in fact one may give a closed form expression as we show in appendix G.1. Hence, the more the representations that are learned in one particular tasks encode something akin to the distance on the underlying space, the more useful these representations should be for generalizing to new such tasks. In other words, one would want to bias the representations to encode more of the geometry of the space and less of the dynamics of the particular tasks.

In the light of this hypothesis, we trained an reinforcement agent, equipped with a TD-learning rule that for a fixed policy would converge to the same weighted representation under $P_{\alpha,\beta}$, and investigated performance in simple navigation tasks. Note that since we now want to understand the benefits on the computational level irrespective of the biological implementation, we use a classical RL model and not the neural network model from the preceding sections - however, all experiments could also be conducted using such a model.

Under a policy π, the RL agent updates a successor representation $SR^{\pi}_{\alpha,\beta}$. Furthermore, the agent learns a reward vector R, and computes the value function of states via $V^{\pi}_{R}(s) = (SR^{\pi}_{\alpha,\beta}R)(s)$. Together with a local transition model $p(s'|s,a)$ (which we assume as given), the agent can then define Q values of state action pairs $Q(s,a)$ and take the next action based on these $Q-$values Sutton and Barto (2018); Dayan (1993).

The task for the agent is split in two parts: In the first part, in each epoch, the agent is initialized in a random location and has to navigate to a fixed goal s_{target}, where a unit reward is received -i.e., $\phi = e_{s_{target}}$. This goal does not change over epochs. After a fixed number of epochs, the goal is changed to a new location s'_{target}, randomly drawn from all other locations. Importantly, the agent is then only allowed to relearn the reward vector, not the successor representation matrix.

We find that both the classical SR, corresponding to parameters $\alpha = 1, \beta = 0$, as well as the symmetrized version $\alpha = \beta = \frac{1}{2}$ are able to learn the navigation tasks with similar mean learning curves - the symmetric agent shows a higher variation though, as can be observed in Figure 5. Although it might be counterintuitive at first that symmetrization yields a policy which still navigates to the correct target, we actually prove in appendix G.1 that indeed an optimal policy is stable under such symmetrization - which shows at least that once such a policy is learned, it can be maintained.

Importantly, we then find empirically that on the new targets, the symmetric learning rule outperforms the classical one on average, while both show higher variation in these tests (Figure 5). Thus, one may argue that the successor representation in a symmetric learning regime affords better generalization - at least in a navigational setting. This is not merely an effect of the trajectories that are sampled with the different learning rules - that is, in particular it cannot be attributed to the higher variation during training on the first target: We repeated the above experiment while learning the successor representations based on the transitions obtained from the classical agent alone - the results remain unchanged, suggesting that the symmetric rule yields representations more apt to generalization without the need of a different sampling regime (Figure H.5). Similarly, the results also hold when controlling the norm of the updates, such that both the asymmetric and symmetric update make an equally big update step at each point.

In other words, the agent can concentrate on solving the current task and still gets afforded a map of the environment which is less influenced by the current policy.

Indeed, one can show that the transition probabilities encoded in $P_{\frac{1}{2},\frac{1}{2}}$ will be closer (than the observed transition probabilities) to those that correspond to an uniform policy, choosing every transition with equal probability - see appendix G. The successor representation of the uniform policy in turn is closely related to the shortest-path distance (Zhang et al., 2021). It can thus be used to generalize to *any* navigational target, while the successor representation under other policies will not neccesarily have this property. Together with our previous considerations on the symmetry of the state space, this led us to hypothesize that the generalization effect of the symmetric learning rule should vanish as soon as there is no such symmetry in the state space any more. We thus repeated the above experiment on a state space that corresponds to a directed graph, where the number of transitions needed to go from s to s' is not necessarily equal to those needed to travel from s' to s. Indeed, we find that in such a setting the effect is reversed: there, the classical learning rule leads to better generalization (Figure 6).

3 Discussion

Here, we have expanded the existing work on successor representation models of the hippocampus. We extended previous local learning rule models by in-

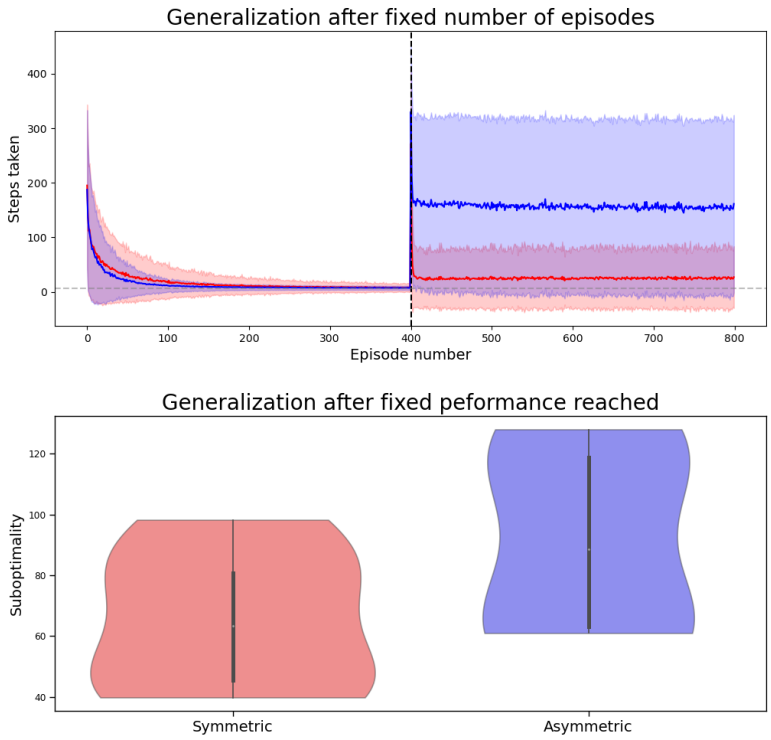

Figure 5: **Generalization experiments in navigation tasks**. **Top:** Agents started in random locations in the environments and had to learn to navigate to fixed targets. After 400 episodes, reward location was switched to a new random location, where agents could only relearn the reward prediction vector but not the SR. (Generalization) performance is visualized by total number of steps taken per episode, for an agent using the classical rule (blue) and an agent using the symmetric rule (red). Dashed line indicates change of target location. We show the average performance over different environments as performance is qualitatively similar, see Figure H.3 for plots in individual environments. **Bottom:** Similar to left plot, but instead of switching target after a fixed number of episodes, the target was switched when the previous target was found with a fixed accuracy. Violin plots show distribution of suboptimality (steps - optimal number of steps) over all environments, for individual environments see Figure H.4. For an outline of the environments see Figure H.2
.

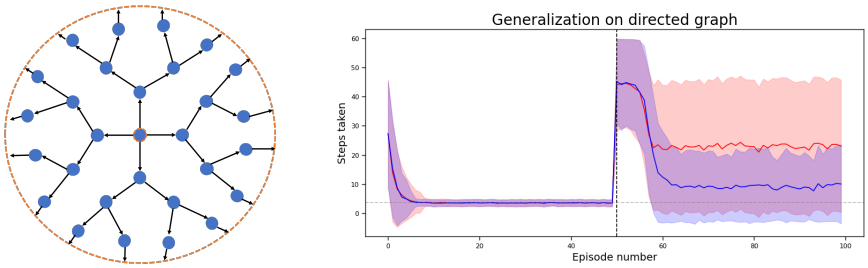

Figure 6: **Generalization experiment on a directed graph**. We conducted the same kind of experiment as in Figure 5 on a directed graph. **Left**: The state space is tree-like, with the addition that from the leaf nodes at the last level one travels back to the central node (orange dashed line). **Right**: In this scenario an SR agent with the classical learning rule (blue) performs better in generalization than one with the symmetric learning rule (red).

cluding learning at two synapses, corresponding to CA3-CA3 recurrent synapses and CA3-CA1 feedforward synapses. In this model, we then studied the effect of making a local learning rule invariant to a reversal of time on the learned representations. We have found that the successor representations that are learned under such a learning rule correspond to encoding a transition structure which is also invariant to a time reversal - irrespective of the actual dynamics which are experienced. In particular, we could show that under such a symmetrized learning rule, place fields shift less when on a linear track, which is in line with empirical findings, showing a distinction between place field shifting in CA3/CA1. The local learning rule we have modified from Fang et al. (2023) is not the only one which could be used to obtain successor representations. Indeed, also George et al. (2022) and Bono et al. (2023) use local learning rules to learn these representations. It seems plausible that to obtain the successor representations we studied here, the exact shape of the learning rule is not important, as long as one can symmetrize it in an appropriate way. This might not even necessarily mean symmetrizing the plasticity kernel: For example, George et al. (2022) use STDP to learn feedforward connections between CA3 and CA1 neurons. Specifically, phase precession was used to provide a location code in the timescale of STDP, and importantly, in the absence of phase precession, a symmetric SR is learned in a biased random walk. Thus, it is plausible that precession in their model breaks the temporal invariance.

To understand the functional significance of such learning rules, we then went on to reinforcement learning experiments, where we trained an agent in navigation tasks. Here, we could show that a symmetrized learning rule affords a better generalization performance when the agent should navigate to a new target. This is interesting, because successor features have been explicitly employed in RL to obtain better generalization to new tasks (Dayan, 1993; Barreto et al., 2017, 2018). In particular, also when the SR was introduced as a model for hippocampus in the neuroscience literature, the generalization capability of

such representations was measured (Stachenfeld et al., 2017; Gershman, 2018). In fact, it was argued in (Stachenfeld et al., 2017) that especially a successor representation that corresponds to a uniform policy should be beneficial for generalization. This was later also identified as a flaw of classical successor representation theory when linear reinforcement learning was suggested as a model for the hippocampal formation instead (Piray and Daw, 2021). In the latter, instead of storing the successor representation under the current policy, the representation under a default policy is stored, and the current policy can be efficiently represented by only encoding the deviations from default. This default policy in navigation is of course intuitively the uniform policy, which could be learned by an agent as soon as it encounters a new environment, in an explorative manner. Our symmetrized learning rule provides a middle ground between these two perspectives: the successor representation that one learns with this learning rule is depending on the current policy, but one can show that the reversible version is always closer to the SR of uniform policy than the SR under the original policy (see appendix). Thus, an agent does not necessarily have to start with exploration to construct a map of its environment, but can do so while performing a particular task. This is of course only possible due to the inductive bias that is inherent in the learning rule, which assumes that the state space is symmetric, since we observed that when this assumption is not true, the SR affords worse generalization.

Adapting learning mechanisms to symmetries of the data is a topic under ongoing investigation in biological and machine learning communities (Higgins et al., 2022; Mercatali et al., 2022). In reinforcement learning, symmetries are frequently considered under the framework of MDP homomorphisms (Van der Pol et al., 2020a; Mavor-Parker et al., 2022; Van der Pol et al., 2020b). This line of research for example aims at learning efficient abstractions of large state spaces into more amenable ones, or exploits known symmetries in the task structure to learn more efficiently. The symmetric state spaces we consider here are simple cases of an equivalence of states induced by the optimal value function: in a navigational problem in a metric space, any two states are equivalent with respect to the optimal value function if they have the same distance to the target (Givan et al., 2003). It would be interesting to investigate whether modifications of TD learning are also beneficial for the more general case of symmetries that are considered in this literature. Interestingly, it has been shown before that TD learning can be considered as performing a form of gradient descent if and only if the dynamics under the policy are reversible (Ollivier, 2018). This is intriguing, since TD-learning is known to be unstable in the continuous setting (Sutton and Barto, 2018) - if our symmetrized learning rule extends to this setting, then it might be possible that it could be useful for a more stable learning in symmetric settings.

Using a fixed learning rule for all synapses of a region might be a simplistic assumption, and in real networks, possibly a variety of learning rules are at play (Debanne et al., 1999). Our model of course does not capture this diversity. However, it would be easy to adapt the learning rules of individual neurons in such a sense that they have varying temporal profile. This could then possibly

lead to a whole spectrum of successor features, each with it's own sensitivity to future and past. In fact, it has been proposed before that the hippocampus encodes representation of both predecessors and successors (Namboodiri and Stuber, 2021). Representing preceding states has furthermore been suggested as useful for exploration in unsupervised reinforcement learning (Yu et al., 2023). Additionally, it is noteworthy that replay phenomena are taking place in CA3 both in a forward as well as a reverse direction. Indeed, for offline planning agents might want to be able to simulate replay in the backward as well as the forward direction (Penny et al., 2013; Yu et al., 2021). These purely predictive respectively postdictive representations would be the two ends of a continuum of representations, with the symmetric learning rule in the middle. Thus, a model representing such a continuum could provide a more nuanced understanding of neural encoding and learning processes.

Our model naturally is a broad oversimplification of matters and lacks biological realism. This is not a problem per se, because we aimed here at analytical amenability and to expand on the theory of successor representations, which operates on the computational level. Still, we could only obtain an explicit relation to successor representations by considering CA3/CA1 in isolation, with external input synapses not subject themselves to learning. In reality, it is now accepted that there is not a simple forward pass through the hippocampus, but rather there are projections from the deep layers of entorhinal cortex back into the superficial layers (which provide input to hippocampus), essentially creating a loop (Kumaran and McClelland, 2012; Canto et al., 2008). With plasticity also happening at these synapses, one would then obtain a model that is not as simple anymore as the one we presented here. Studying the representations in such an extended model, which would include learning for example in synapses from HC to EC, and whether these can still be framed in Successor representation theory, would be an interesting future research direction and could possibly build a bridge to computational models which include hippocampal-entorhinal interactions (George et al., 2023a; Whittington et al., 2020).

Finally, we want to mention that the hippocampal formation is also of high interest in studying generalization in human cognitive neuroscience (Theves et al., 2021; Garvert et al., 2023). Since evidence is growing that the systems which are partaking in spatial representations are also recruited to encode more abstract variables, possibly forming 'conceptual spaces' (Nitsch et al., 2023; Bottini and Doeller, 2020; Constantinescu et al., 2016), it might be interesting to understand whether an inherent bias for symmetry also shapes these representations - and thus, aspects of cognition where metric space structure might be inadequate.

In conclusion, our model contributes to the theoretical framework of hippocampal predictive representations, both on the mechanistic level through suggesting the use of symmetric local learning rules, as well as on the functional level, where such learning might be useful for generalization in spatial learning.

4 Methods and Materials

5 Methods

5.1 Neural Network Model

We consider populations of rate-based neurons $p^1 \in \mathbb{R}^m$, $p^2 \in \mathbb{R}^n$ respectively. The population p^1 is recurrently connected via a matrix of synaptic weights $W^r \in \mathbb{R}^{m\times m}$, and feeds its activity forward to population p^2 via the matrix of synaptic weights $W^f \in \mathbb{R}^{n\times m}$. Both regions receive additional external inputs, ϕ^1, ϕ^2, and decay to equilibrium in the absence input. Then we can posit the following dynamics

$$\frac{d}{dt}p^1 = -p^1 + \sigma\left(\lambda_p W^r p^1 + (1-\lambda_p)\phi^1\right) \tag{20}$$

$$\frac{d}{dt}p^2 = -p^2 + \sigma\left(\lambda_q W^f p^1 + (1-\lambda_q)\phi^2\right). \tag{21}$$

Here, σ is an activation function, and the λ parameters control the relative weight of the different types of inputs. Of course, on the computational level, these parameters will correspond to the discounting factor of the successor representation, effectively scaling how far in the future predictions are made. On the level of biological implementation, these parameters might speculatively be associated with acetylcholine, controlling the relative strength of external and internal input in hippocampus (Hasselmo, 2006; Fang et al., 2023). Assuming the weights (and the external inputs) change on a slower timescale than the population dynamics, we can let these equations go to equilibrium for analysis - in practice, we can also integrate the ODE above with a timescale τ orders of magnitude smaller than the timescale of learning. If σ is approximately linear, this results in

$$p^1 = (1-\lambda_p)(\mathrm{Id}\text{-}\lambda_\mathrm{p}\mathrm{W}^\mathrm{r})^{-1}\phi^1 \tag{22}$$

$$p^2 = \lambda_q W^f p^1 + (1-\lambda_q)\phi^2. \tag{23}$$

We now take these equations as a basis to define population activities in discrete time,

$$p^1_t := (1-\lambda_p)(\mathrm{Id}\text{-}\lambda_\mathrm{p}\mathrm{W}^\mathrm{r}_\mathrm{t})^{-1}\phi^1_t \tag{24}$$

$$p^2_t := \lambda_q W^f_t p^1_t + (1-\lambda_q)\phi^2_t, \tag{25}$$

and define a learning rule which changes the matrices of synaptic weights at each discrete timestep, see next section.

We assume that the animal receives inputs depending on the current state of the environment, which we will denote by the process S_t. In the reinforcement learning setting, the agent influences the way in which the state-process is sampled by selecting actions A_t, but at this point this is not relevant since

we are only building up a predictive representation of states, not actions. The inputs to the two neural populations thus take the form

$$\phi_t^k = \phi^k(S_t) \tag{26}$$

where ϕ^k is a function mapping the state space to an activation pattern in neural space. If the state space is discrete, we can also write this as

$$\phi_t^k = \Phi^k e_{S_t} \tag{27}$$

with Φ^k now being a matrix of appropriate dimension, and e_{S_t} is the unit-vector corresponding to state S_t, which hence selects the corresponding activation pattern from Φ^ks columns. In the special case $\Phi^k = \mathrm{Id}$, we have a one-hot encoding, where hence every cell corresponds to a distinct state.

5.2 Reinforcement Learning and Successor Features

In Reinforcement learning, one generally considers a Markov decision process (MDP), that is a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$, where $\mathcal{S}, \mathcal{A}$ are the sets of possible states and actions, $T(s'|s,a)$ gives the transition probability from state s to state s' when choosing action a, $R(s,a)$ is obtained reward when chosing action a in state S, and γ is a discount factor. The goal in RL usually is to find an optimal policy, that is a probability distribution $\pi(a|s)$ over actions, given states, which maximizes the expected discounted cumulative reward

$$\mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R(S_t, A_t)\right], \tag{28}$$

where the states and actions S_t, A_t are sampled according to the policy π and the transition probabilities T. In our experiments, we are only interested in the particularly simple case where the transitions are deterministic - that is, taking an action a in state s surely leads to a specified state $s'(a)$. Furthermore, we only consider the situation where the reward is a function of the state only. However, all definitions in the paragraph below readily generalize to functions of states and actions. Assume a fixed policy π and a process S_t following said policy. Now consider any mapping from the state space

$$\phi : \mathcal{S} \to \mathbb{R}^m, \tag{29}$$

which we can interpret as an observable generated by the state space. Then define a function on the state space

$$SF_\phi^t(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \phi(S_{t+k}) | S_t = s\right]. \tag{30}$$

SF_ϕ thus gives the expected (exponentially weighted) cumulative future sum of the observation or feature ϕ, given the current state s. This is hence called

a 'successor feature' in the literature. Define $G_t = \sum_{k=0}^{\infty} \gamma^k \phi(S_{t+k})$. We then have

$$SF_\phi^t(s) = \mathbb{E}[G_t|S_t = s] = \mathbb{E}[\phi(S_t) + \sum_{k=1}^{\infty} \gamma^k \phi(S_{t+k})|S_t = s] \tag{31}$$

$$= \mathbb{E}[\phi(S_t) + \gamma G_{t+1}|S_t = s] \tag{32}$$

$$= \mathbb{E}[\phi(S_t)|S_t = s] + \gamma \mathbb{E}[\mathbb{E}[G_{t+1}|S_{t+1}]|S_t = s] \tag{33}$$

$$= \phi(s) + \sum_{a,s'} \mathbb{P}[S_{t+1} = s'|S_t = s]\gamma SF_\phi^{t+1}(s'). \tag{34}$$

In particular, if the transition probabilities of S_t are time-homogeneous, we see that SF itself does not depend on t, and we can write

$$SF_\phi(s) = \mathbb{E}\left[\phi(S_t) + SF_\phi(S_{t+1})|S_t = s\right]. \tag{35}$$

Temporal Difference learning (TD-learning) uses this relationship to construct a target to update an estimate of SF_ϕ online. Indeed, let

$$\tilde{SF}_{t+1}(S_t) = \tilde{SF}_t(S_t) + \varepsilon \Delta_t \tag{36}$$

$$\Delta_t = \phi(S_t) + \gamma \tilde{SF}_t(S_{t+1}) - \tilde{SF}_t(S_t). \tag{37}$$

Then we see that if $\tilde{SF}_t = SF_\phi$, we get $\mathbb{E}[\Delta_t|S_t = s] = 0$. Thus, here $\phi(S_t) + \gamma \tilde{SF}_t(S_{t+1})$ is used as a bootstrapping estimate of the target $\mathbb{E}[\phi(S_t) + \gamma \tilde{SF}_t(S_{t+1})|S_t = s]$, and $\tilde{SF}_t$ is updated according to the error to that target - in total, Δ_t is thus also called TD-error.

Successor features encompass important special case examples: For the choice of $\phi = R$ the reward function, SF_R becomes the value function: The value function under a policy π, which is typically denoted as V^π, hence encodes the weighted cumulative sum of expected rewards, given the current state. In particular, the current estimate of the reward function may be used to define a new policy as

$$\begin{aligned} \pi^*(\cdot|s) &= \delta_{a^*} \\ a^* &= \arg\max_a V^\pi(s'(a)). \end{aligned} \tag{38}$$

That is, the policy deterministically selects the action a^* which leads to the next state with the highest value according to the current value function estimate. Iterating this process then successively learns a better value function - this process of updating an estimate of the value function and then choosing an optimal policy with respect to it is the basis of the classical TD-learning algorithm (Sutton and Barto, 2018). Indeed, for our navigation experiments we use this approach, only replacing the maximum in Equation 38 by a softmax which smoothens the transition probabilities. Note that the above assumes a model of which actions lead to which next states - which posing as given should be a sensible assumption in navigation problems - but a completely model-free approach simply computes the value of a state-action pair instead.

Successor Representations In the case that we have that ϕ is in fact an injective mapping, we can define a modified version of successor features as

$$SR_\phi(\rho) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \phi(S_{t+k}) | \phi(S_t) = \rho\right], \tag{39}$$

where $\rho \in \phi(\mathcal{S})$. Here, the injectivity of the mapping is necessary to ensure SR defined as above still enjoys desirable properties like homogenity in time and the Bellman equation, but besides of that one could also define a similar quantity without using injectivity. Now, in the injective situation, we would like to call the above 'successor representation'. This is because when we take mapping $\phi(s) = e_s$, where e_S is a unit vector in $\mathbb{R}^{|S|}$, then we obtain

$$SR_\phi(e_s) = (Id - \gamma P)_s^{-1} \tag{40}$$

which corresponds to the classical definition of successor representation. In general, if we assume $\Phi \in \mathbb{R}^{m \times |S|}$ is the matrix of features (could also be an operator if we allow for continuous state space), then we have that

$$SR_\phi(\phi(s)) = (\Phi(Id - \gamma P)^{-1})_s = (\phi(Id - \gamma P)^{-1}\Phi^-)\phi(s), \tag{41}$$

where Φ^- is such that $\Phi^-\Phi = \mathrm{Id}_{|S|}$.

Generalization and Successor Features The idea behind using successor features for some set of function ϕ^i, instead of simply directly computing the the value function is that of generalization/transfer: Assume the reward function R can be written as a linear combination of the features ϕ, that is

$$R = \sum_i w_i \phi^i. \tag{42}$$

Then also for the respective predictive representations one has

$$SF_R(s) = \sum_i w_i SF_\phi(s). \tag{43}$$

Now assuming that the reward changes to a new function $\tilde{R}$, which also can be expressed with the features, then the only thing that has to be relearned is the weights w_i, while the successor features SF_ϕ can be reused. Thus, one may then generalize more easily to a new task, since the transition structure under a policy is essentially already learnt. In particular, in a discrete state space, a fixed reward function can R can of course be encoded by a reward vector $\mathbf{R}$, that is

$$\mathbf{R} = \sum_s R(s) e_s. \tag{44}$$

The value function is then simply given through the classical SR as

$$V(s) = (Id - \gamma P)^{-1} \mathbf{R}(s). \tag{45}$$

Thus, for a fixed policy, this separates the computation of value into learning a successor representation and learning a reward vector. Our navigation experiments probe the generalization capability of this approach, by introducing a new reward vector $\tilde{\mathbf{R}}$, after an optimal policy and a SR for a reward vector $\mathbf{R}$ have been learned. Importantly, only the reward vector is allowed to be relearned, while the SR has to be reused from the previous task.

Symmetric TD-learning In our reinforcement learning experiments, we use a modified version of TD-learning to mimic the behaviour of the local learning rule in the SR-network. In practice, Successor features are typically parametrized by some parameter θ (e.g., the weights of a neural network), which is then updated to reduce the TD-error. That is, for each value of θ we obtain a map $\tilde{SF}_\theta(s)$ of states. We can then update the parameter θ via

$$\theta_{t+1} = \theta_t + \varepsilon \Delta(\theta)_t \tag{46}$$

$$\Delta(\theta)_t = \alpha \left(\phi(S_t) + \gamma \tilde{SF}_{\theta_t}(S_{t+1}) - \tilde{SF}_{\theta_t}(S_t) \right)^T \nabla_\theta \tilde{SF}_{\theta_t}(S_t) \tag{47}$$

$$+ \beta \left(\phi(S_{t+1}) + \gamma \tilde{SF}_{\theta_t}(S_t) - \tilde{SF}_{\theta_t}(S_{t+1}) \right)^T \nabla_\theta \tilde{SF}_{\theta_t}(S_{t+1}), \tag{48}$$

where α, β are parameters corresponding to the ones controlling the local learning rule. In particular, for $\alpha = 1, \beta = 0$ one obtains the classical TD-learning rule for function approximation (Sutton and Barto, 2018) and for $\alpha = 0, \beta = 1$ one obtains the 'predecessor representation' (Yu et al., 2023). The case $\alpha = \beta$ yields a symmetrized version of TD-learning. In our experiments, we use a particularly simple version of the above: for a discrete state space, one can simply parametrize $\tilde{SF}$ by means of a matrix $M \in \mathbb{R}^{d \times k}$, where d is the number of features and k is the number of states. Then one has $\tilde{SF}_M(s) = Me_s$, and hence the update rule

$$\Delta(M)_t = \alpha \left(\phi(S_t) e_{S_t}^T + \gamma M e_{S_{t+1}} e_{S_t}^T - M e_{S_t} e_{S_t}^T \right)^T + \beta \left(\phi(S_{t+1}) e_{S_{t+1}}^T + \gamma M e_{S_t} e_{S_{t+1}}^T - M e_{S_{t+1}} e_{S_{t+1}}^T \right). \tag{49}$$

5.3 A predictive representation leads to backward shift only in the asymmetric case

Here we want to give an analytic explanation for the stronger backward shifts of the centre of mass when using an asymmetric learning rule, while a symmetric learning rule does not result in a shift. This is easily explained if we ignore boundaries and go to a continuous situation. Let's assume an agent is repeatedly running on the real line $\mathbb{R}$. In the beginning, before learning, the cells in our model are just driven by the external input, and thus can be taken equal to the features. That is, cell j will take the value $\phi^j(x)$ when at position x. After

learning, it will instead encode a successor feature, which in the continuous case becomes

$$SF_{\phi^j}(x) = \int_0^\infty e^{-\gamma t} \int \phi^j(y) p_t(y|x) dy dt, \tag{50}$$

where $p_t(y|x)$ is the probability of transitioning from state x to state y in time t. The centre of mass (COM) is just the normalized activity, that is, for example for the raw feature it is

$$\frac{\int x\phi^j(x)dx}{\int \phi^j(x)dx}. \tag{51}$$

It is instructive to study the successor feature with deterministic dynamics: Assume we are moving with constant velocity v, that is, $p_t(y|x) = \delta(y-(x+tv))$ then the successor feature takes the form

$$SF_{\phi^j}(x) = \int_0^\infty e^{-\gamma t} \phi^j(x+tv) dt. \tag{52}$$

Now, taking the mean over all positions yields:

$$\int x \int_0^\infty e^{-\gamma t} \phi^j(x+tv) dt dx = \int_0^\infty x \int e^{-\gamma t} \phi^j(x+tv) dx dt \tag{53}$$

$$= \int_0^\infty (u - tv) \int e^{-\gamma t} \phi^j(u) du dt \tag{54}$$

$$= \int_0^\infty u \int e^{-\gamma t} \phi^j(u) du dt - \int_0^\infty tv \int e^{-\gamma t} \phi^j(u) du dt \tag{55}$$

$$= \frac{1}{\gamma} \int u\phi^j(u) du - \frac{v}{\gamma^2} \int \phi^j(u) du. \tag{56}$$

After normalization (that is, dividing by $\int SF_{\phi^j}(x)dx = \frac{1}{\gamma}\int \phi^j(x)dx$), this simply becomes

$$\frac{\int u\phi^j(u)du}{\int \phi^j(u)du} - \frac{v}{\gamma}. \tag{57}$$

That is, the center of mass is shifted backward by a factor which is controlled by the prediction timescale γ and the velocity of the movement. We thereby also obtain the seemingly new prediction that faster running speed and smaller terminal place field size (as a proxy for γ) should result in bigger shifts. The same relation holds also when the dynamics come from a Brownian motion with constant drift, that is $dx_t = vx_t = \sqrt{\rho}B_t$, as we show in appendix B . Now when we symmetrize the Brownian motion with drift, the drift term will disappear, that is, we have a standard Brownian motion. It is thus clear that under this motion, there should be no shift of the centre of mass.

5.4 Experiments

5.5 Convergence Experiments

For our initial convergence experiments (Figure 3, top row) we simulate our model in a circular random walk setting with 30 states and use $n = 40$ cells in

each layer. The input at each state is drawn from an i.i.d. Gaussian distribution ($\sigma = 0.1$) (i.e., for each cell and each state we draw a value from a Gaussian distribution, this is then assumed to be the input for that cell whenever the specific state is visited). Throughout this and the following experiments, we use a moderately high value of $\gamma = 0.7$ - meaning a relatively large timescale/small discounting. This is an arbitrary choice, but relatively high values have been used in the past in SR-theories of hippocampus Geerts et al. (2020).

We observe $1e5$ transitions, and repeat the experiment 30 times. We then repeat the same experiment (top row, right of Figure 3), also drawing the transition matrix P randomly. To track the convergences, we consider the loss terms

$$\mathcal{L}W_r = ||\mathbb{E}[\Delta W_r]|| \tag{58}$$

$$\mathcal{L}W_f = ||\mathbb{E}[\Delta W_f]||, \tag{59}$$

where the terms on the right hand vanish at convergence and are defined in Equation 93. The resulting convergence curves are shown in Figure 3. These experiments are conducted with a linear activation function, we also repeat this experiment with the activation functions **tanh**, **relu** as shown in Figure H.1

To check the effect of different values α, β, we conduct a parameter sweep in a circular random walk, and random Gaussian features as above. For each combination α, β we run 30 random initializations for $1e3$ iterations, and check this combination once for the recurrent layer and once for the feedforward layer. Here, we track convergence similar as above, but now we take the fraction

$$\frac{\mathcal{L}W}{\mathcal{L}W_0} \tag{60}$$

with W_0 the initial weight, to judge if we are moving towards or away from equilibrium. This results in the matrices shown in *Figure* 3. To speed up the experiments, we use a smaller network for these experiments, with 20 neurons in each layer and 10 states. Again we use $dt = 0.1$.

5.6 Linear Track Experiments

To simulate the animal running on a linear track, we partitioned a track of length $300cm$ in 50 discrete states and let the agent perform a rightward biased walk that either took a step to the right ($p = 0.9$) or stayed in place ($p = 0.1$). We chose a time step of $dt = 0.4$, corresponding to a velocity of $\approx 15cm/s$. The agent would run 25 laps on the linear track, with a short resting phase in between two laps. We set up a two-layer model, with $n = 100$ cells in each layer, and again $\gamma = 0.7$. For the feedforward weights, we always used the parameters $\alpha = 1, \beta = 0$, while for the recurrent weights we tested both the symmetric case $\alpha = \beta = \frac{1}{2}$ as well as the asymmetric case $\alpha = 1, \beta = 0$. We then performed an analysis approach as in Dong et al. (2021). For each lap, we collected the mean activities of each layer per state and computed the center of mass (COM) by the formula

$$\text{COM}_j = \frac{\sum_x x\overline{p}_j(x)}{\sum_x \overline{p}_j(x)} \tag{61}$$

where x is position on the track and $\overline{p}_j(x)$ denotes the mean activity of cell j when at that location. To obtain a distribution of the overall shifts of the COMs over all laps, we calculated the average COMs for the first and the last five laps and subtracted them from each other, obtaining the histograms in Figure 4. To track the evolution of the shift, we subtracted the COMs from the 12-th lap, which yields a more gradual tracking of shifts, see Figure 4.

5.7 Reinforcement Learning

We investigated the generalization capabilities of a symmetric over an asymmetric learning rule in TD-learning in different scenarios.

5.7.1 Navigation Experiments

Since the hippocampal formation is prominently involved in navigation tasks, we first focused on tasks of such nature. We thus studied different variations of the same general problem setup: Given a grid environment with a deterministic transition structure, the agent would receive a unit reward only when arriving at the designated target state s_T, upon which the episode is terminated, and the next episode starts in an initial state s_0, drawn uniformly at random. The agent is thus encouraged to navigate from all possible starting states to the target state to receive a reward. We used the Neuronav toolbox (Juliani et al., 2022) to implement the grid environment and implemented our modified SR-agent in the framework of that toolbox.

To check for generalization capability, we would train agents which utilize the respective learning rules (symmetric,asymmetric) to navigate to a fixed s_T first. We trained the agent either for a fixed number of episodes (we used 400 episodes with a maximum number of steps of 400), or until fixed performance criterion was met (the mean deviation from optimal performance for the preceding 8 e-episodes was lesser than 2). Then, we would randomly select a new target state $\tilde{s_T}$ from among all possible states. Importantly, after the modification of the target states, the agents were only allowed to modify the reward-prediction vector w, but not the successor representations SF themselves. That is, they could only learn the reward structure of the new task, but had to rely on the transition structure that was encoded during learning the previous task, which also depends on the policy. We repeated this experiment for 200 times, with randomly drawn combinations $(s_T, \tilde{s_T})$ in each repetition. For all experiments, we used a learning rate of $1e{-}1$ and set $\gamma = 0.7$. The results of these experiments are depicted in Figure 5. We repeated the second kind of analysis for different values of γ and the learning rate, as shown in *Figure H.7*

Since all the environments we tested in the above setting have a symmetric nature, that is, their underlying state space is an undirected graph, we repeated the same generalization experiment in a setting where the state space is a directed graph, and hence travel time between two nodes is not symmetric. We constructed a simple graph with 17 nodes, which is essentially a tree graph with the addition of a directed edge from the lowest level to the highest. On this

graph, we performed the same kind of navigation experiment. We used the same values for the learning rate and γ, but reduced the number of episodes to 50 since the state space is smaller and hence can be learned faster.

5.8 Acknowledgements

We thank William de Cothi, Tom George and Nikola Milosevic for helpful discussions. JK is supported by the Max Planck School of Cognition. CFD's research is supported by the Max Planck Society, the European Research Council, the Kavli Foundation, the Jebsen Foundation, Helse Midt Norge and The Research Council of Norway. CB is funded by a Wellcome SRF.

5.9 Code Availability

All code and data used for this project is available at https://github.com/jakeck1/sympredlearning.

References

Barreto, A., Borsa, D., Quan, J., Schaul, T., Silver, D., Hessel, M., Mankowitz, D., Zidek, A., and Munos, R. (2018). Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pages 501–510. PMLR.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., and Silver, D. (2017). Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30.

Behrens, T. E., Muller, T. H., Whittington, J. C., Mark, S., Baram, A. B., Stachenfeld, K. L., and Kurth-Nelson, Z. (2018). What is a cognitive map? organizing knowledge for flexible behavior. *Neuron*, 100(2):490–509.

Bi, G.-q. and Poo, M.-m. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience*, 18(24):10464–10472.

Bono, J., Zannone, S., Pedrosa, V., and Clopath, C. (2023). Learning predictive cognitive maps with spiking neurons during behavior and replays. *Elife*, 12:e80671.

Bottini, R. and Doeller, C. F. (2020). Knowledge across reference frames: Cognitive maps and image spaces. *Trends in Cognitive Sciences*, 24(8):606–619.

Canto, C. B., Wouterlood, F. G., Witter, M. P., et al. (2008). What does the anatomical organization of the entorhinal cortex tell us? *Neural plasticity*, 2008.

Chung, F. (2005). Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9:1–19.

Chung, F. R. (1996). Laplacians of graphs and cheeger's inequalities. *Combinatorics, Paul Erdos is Eighty*, 2(157-172):13–2.

Constantinescu, A. O., O'Reilly, J. X., and Behrens, T. E. (2016). Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468.

Corneil, D. S. and Gerstner, W. (2015). Attractor network dynamics enable preplay and rapid path planning in maze–like environments. *Advances in neural information processing systems*, 28.

Dayan, P. (1993). Improving generalization for temporal difference learning: The successor representation. *Neural computation*, 5(4):613–624.

Dayan, P. and Sejnowski, T. J. (1994). Td (λ) converges with probability 1. *Machine Learning*, 14:295–301.

De Cothi, W. and Barry, C. (2020). Neurobiological successor features for spatial navigation. *Hippocampus*, 30(12):1347–1355.

De Cothi, W., Nyberg, N., Griesbauer, E.-M., Ghanamé, C., Zisch, F., Lefort, J. M., Fletcher, L., Newton, C., Renaudineau, S., Bendor, D., et al. (2022). Predictive maps in rats and humans for spatial navigation. *Current Biology*, 32(17):3676–3689.

Debanne, D., Gähwiler, B. H., and Thompson, S. M. (1999). Heterogeneity of synaptic plasticity at unitary CA3–CA1 and CA3–CA3 connections in rat hippocampal slice cultures. *Journal of Neuroscience*, 19(24):10664–10671.

Dong, C., Madar, A. D., and Sheffield, M. E. (2021). Distinct place cell dynamics in CA1 and CA3 encode experience in new environments. *Nature communications*, 12(1):2977.

Dordek, Y., Soudry, D., Meir, R., and Derdikman, D. (2016). Extracting grid cell characteristics from place cell inputs using non-negative principal component analysis. *Elife*, 5:e10094.

Eichenbaum, H., Dudchenko, P., Wood, E., Shapiro, M., and Tanila, H. (1999). The hippocampus, memory, and place cells: is it spatial memory or a memory space? *Neuron*, 23(2):209–226.

Fang, C., Aronov, D., Abbott, L., and Mackevicius, E. L. (2023). Neural learning rules for generating flexible predictions and computing the successor representation. *Elife*, 12:e80680.

Friston, K. (2002). Functional integration and inference in the brain. *Progress in neurobiology*, 68(2):113–143.

Garvert, M. M., Saanum, T., Schulz, E., Schuck, N. W., and Doeller, C. F. (2023). Hippocampal spatio-predictive cognitive maps adaptively guide reward generalization. *Nature Neuroscience*, 26(4):615–626.

Geerts, J. P., Chersi, F., Stachenfeld, K. L., and Burgess, N. (2020). A general model of hippocampal and dorsal striatal learning and decision making. *Proceedings of the National Academy of Sciences*, 117(49):31427–31437.

George, T., Stachenfeld, K., Barry, C., Clopath, C., and Fukai, T. (2023a). A generative model of the hippocampal formation trained with theta driven local learning rules. In *Thirty-seventh Conference on Neural Information Processing Systems*.

George, T. M., de Cothi, W., Clopath, C., Stachenfeld, K., and Barry, C. (2022). Ratinabox: A toolkit for modelling locomotion and neuronal activity in continuous environments. *bioRxiv*, pages 2022–08.

George, T. M., de Cothi, W., Stachenfeld, K. L., and Barry, C. (2023b). Rapid learning of predictive maps with stdp and theta phase precession. *Elife*, 12:e80663.

Gershman, S. J. (2018). The successor representation: its computational logic and neural substrates. *Journal of Neuroscience*, 38(33):7193–7200.

Givan, R., Dean, T., and Greig, M. (2003). Equivalence notions and model minimization in markov decision processes. *Artificial Intelligence*, 147(1-2):163–223.

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., and Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806.

Hasselmo, M. E. (2006). The role of acetylcholine in learning and memory. *Current opinion in neurobiology*, 16(6):710–715.

Higgins, I., Racanière, S., and Rezende, D. (2022). Symmetry-based representations for artificial and biological general intelligence. *Frontiers in Computational Neuroscience*, 16:836498.

Huang, Y. and Rao, R. P. (2011). Predictive coding. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(5):580–593.

Johns, J. and Mahadevan, S. (2007). Constructing basis functions from directed graphs for value function approximation. In *Proceedings of the 24th international conference on Machine learning*, pages 385–392.

Jost, J. and Mulas, R. (2019). Cheeger-like inequalities for the largest eigenvalue of the graph laplace operator. *arXiv preprint arXiv:1910.12233*.

Juliani, A., Barnett, S., Davis, B., Sereno, M., and Momennejad, I. (2022). Neuro-nav: a library for neurally-plausible reinforcement learning. *arXiv preprint arXiv:2206.03312*.

Knierim, J. J. (2015). The hippocampus. *Current Biology*, 25(23):R1116–R1121.

Kumaran, D. and McClelland, J. L. (2012). Generalization through the recurrent interaction of episodic memories: a model of the hippocampal system. *Psychological review*, 119(3):573.

Kushner, H. J. and Clark, D. S. (2012). *Stochastic approximation methods for constrained and unconstrained systems*, volume 26. Springer Science & Business Media.

Lee, I., Rao, G., and Knierim, J. J. (2004a). A double dissociation between hippocampal subfields: differential time course of CA3 and CA1 place cells for processing changed environments. *Neuron*, 42(5):803–815.

Lee, I., Yoganarasimha, D., Rao, G., and Knierim, J. J. (2004b). Comparison of population coherence of place cells in hippocampal subfields CA1 and CA3. *Nature*, 430(6998):456–459.

Leutgeb, S., Leutgeb, J. K., Treves, A., Moser, M.-B., and Moser, E. I. (2004). Distinct ensemble codes in hippocampal areas CA3 and CA1. *Science*, 305(5688):1295–1298.

Machado, M. C., Bellemare, M. G., and Bowling, M. (2017a). A laplacian framework for option discovery in reinforcement learning. In *International Conference on Machine Learning*, pages 2295–2304. PMLR.

Machado, M. C., Rosenbaum, C., Guo, X., Liu, M., Tesauro, G., and Campbell, M. (2017b). Eigenoption discovery through the deep successor representation. *arXiv preprint arXiv:1710.11089*.

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

Markram, H., Gerstner, W., and Sjöström, P. J. (2011). A history of spike-timing-dependent plasticity. *Frontiers in synaptic neuroscience*, 3:4.

Mavor-Parker, A. N., Sargent, M. J., Banino, A., Griffin, L. D., and Barry, C. (2022). A simple approach for state-action abstraction using a learned mdp homomorphism. *arXiv preprint arXiv:2209.06356*.

Mehta, M. R., Barnes, C. A., and McNaughton, B. L. (1997). Experience-dependent, asymmetric expansion of hippocampal place fields. *Proceedings of the National Academy of Sciences*, 94(16):8918–8921.

Mercatali, G., Freitas, A., and Garg, V. (2022). Symmetry-induced disentanglement on graphs. *Advances in Neural Information Processing Systems*, 35:31497–31511.

Mishra, R. K., Kim, S., Guzman, S. J., and Jonas, P. (2016). Symmetric spike timing-dependent plasticity at CA3–CA3 synapses optimizes storage and recall in autoassociative networks. *Nature communications*, 7(1):11552.

Momennejad, I., Russek, E. M., Cheong, J. H., Botvinick, M. M., Daw, N. D., and Gershman, S. J. (2017). The successor representation in human reinforcement learning. *Nature human behaviour*, 1(9):680–692.

Namboodiri, V. M. K. and Stuber, G. D. (2021). The learning of prospective and retrospective cognitive maps within neural circuits. *Neuron*, 109(22):3552–3575.

Nitsch, A., Garvert, M. M., Bellmund, J. L., Schuck, N. W., and Doeller, C. F. (2023). Grid-like entorhinal representation of an abstract value space during prospective decision making. *bioRxiv*, pages 2023–08.

O'Keefe, J. and Nadel, L. (1978). The hippocampus as a cognitive map. *Hippocampus*, 3:570.

Ollivier, Y. (2018). Approximate temporal difference learning is a gradient descent for reversible policies. *arXiv preprint arXiv:1805.00869*.

Penny, W. D., Zeidman, P., and Burgess, N. (2013). Forward and backward inference in spatial cognition. *PLoS computational biology*, 9(12):e1003383.

Piray, P. and Daw, N. D. (2021). Linear reinforcement learning in planning, grid fields, and cognitive control. *Nature communications*, 12(1):4942.

Rao, R. P. and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87.

Roth, E. D., Yu, X., Rao, G., and Knierim, J. J. (2012). Functional differences in the backward shifts of CA1 and CA3 place fields in novel and familiar environments. *PloS one*, 7(4):e36035.

Russek, E. M., Momennejad, I., Botvinick, M. M., Gershman, S. J., and Daw, N. D. (2017). Predictive representations can link model-based reinforcement learning to model-free mechanisms. *PLoS computational biology*, 13(9):e1005768.

Scoville, W. B. and Milner, B. (1957). Loss of recent memory after bilateral hippocampal lesions. *Journal of neurology, neurosurgery, and psychiatry*, 20(1):11.

Sprekeler, H. (2011). On the relation of slow feature analysis and laplacian eigenmaps. *Neural computation*, 23(12):3287–3302.

Squire, L. R., Stark, C. E., and Clark, R. E. (2004). The medial temporal lobe. *Annu. Rev. Neurosci.*, 27:279–306.

Stachenfeld, K. L., Botvinick, M., and Gershman, S. J. (2014). Design principles of the hippocampal cognitive map. *Advances in neural information processing systems*, 27.

Stachenfeld, K. L., Botvinick, M. M., and Gershman, S. J. (2017). The hippocampus as a predictive map. *Nature neuroscience*, 20(11):1643–1653.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Theves, S., Neville, D. A., Fernández, G., and Doeller, C. F. (2021). Learning and representation of hierarchical concepts in hippocampus and prefrontal cortex. *Journal of Neuroscience*, 41(36):7675–7686.

Van der Pol, E., Kipf, T., Oliehoek, F. A., and Welling, M. (2020a). Plannable approximations to mdp homomorphisms: Equivariance under actions. *arXiv preprint arXiv:2002.11963*.

Van der Pol, E., Worrall, D., van Hoof, H., Oliehoek, F., and Welling, M. (2020b). Mdp homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210.

Vértes, E. and Sahani, M. (2019). A neurally plausible model learns successor representations in partially observable environments. *Advances in Neural Information Processing Systems*, 32.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17:395–416.

Whittington, J. C., Muller, T. H., Mark, S., Chen, G., Barry, C., Burgess, N., and Behrens, T. E. (2020). The tolman-eichenbaum machine: unifying space and relational memory through generalization in the hippocampal formation. *Cell*, 183(5):1249–1263.

Wu, Y., Tucker, G., and Nachum, O. (2018). The laplacian in rl: Learning representations with efficient approximations. *arXiv preprint arXiv:1810.04586*.

Yu, C., Burgess, N., Sahani, M., and Gershman, S. (2023). Successor-predecessor intrinsic exploration. *arXiv preprint arXiv:2305.15277*.

Yu, C., Li, D., Hao, J., Wang, J., and Burgess, N. (2021). Learning state representations via retracing in reinforcement learning. *arXiv preprint arXiv:2111.12600*.

Zhang, T., Rosenberg, M., Jing, Z., Perona, P., and Meister, M. (2021). Endotaxis: A neuromorphic algorithm for mapping, goal-learning, navigation, and patrolling. *bioRxiv*, pages 2021–09.

A Learning Rule

Recall the learning rule, which with parameters $\alpha = 1, \beta = 0$ reads

$$\Delta W = (p_{post} - W p_{pre}) p_{pre}^T. \tag{62}$$

This learning rule can be seen as performing a sort of conditional expectation objective: We have that in general

$$\mathbb{E}\left[\Delta W | p_{pre}\right] = 0 \tag{63}$$

only if

$$W p_{pre} = \mathbb{E}\left[p_{post}(W, pre) | p_{pre}\right] \tag{64}$$

where we have written the dependency of the postsynaptic activity more explicitly. Thus in some sense, this learning rule is very similar in spirit to classical TD-learning, since also there one tries to fulfill a conditional expectation equation (the Bellman equation) and works towards a solution by bootstrapping (Sutton and Barto, 2018). Indeed, also in Fang et al. (2023) this learning rule was derived starting from the assumption of td-learning with function approximation.

Of course, the equation above will mostly not have an exact solution, since condition on an input is in general a nonlinear operation. However, in special cases there will be a solution: If for example we have a conditional Gaussian relationship, that is $p_{post}|p_{pre} \sim \mathcal{N}(p_{post}; A p_{pre}, \sigma)$, then an exact solution is possible and simply retrieves A.More generally, the solution will simply correspond to a regression, as we will see below. In general, one can thus understand this as a simple form of a predictive coding objective, where the weights are adapted in such a way that postsynaptic activity becomes predictable Huang and Rao (2011). Let us now understand how this predictive coding rule learns successor representations.

A.1 Learned representations through the predictive coding rule

Consider first a feedforward network with update rules

$$p_t = \gamma V \phi_t + (1 - \gamma) \psi_t. \tag{65}$$

Here, $\phi_t(s_t)$ is the input through the feedforward weights V and $\psi_t(s_t)$ is external input, both depending on the state s_t of a Markov process. Now Equation 64 becomes

$$(1 - \gamma)\mathbb{E}[\psi_{t+1}|s_t] = V(\phi_t - \gamma \mathbb{E}[\phi_{t+1}|s_t]) \tag{66}$$

Assume the case of a finite state space, where we can write $\phi(s_t) = \Phi e_{s_t}, \psi(s_t) = \Psi e_{s_t}$, for matrices Φ, Ψ which we assume invertible. It is then not hard to see that a solution is

$$V = (1 - \gamma) \Psi P^T \left(\mathrm{Id} - \gamma P^T\right)^{-1} \Phi^{-1}. \tag{67}$$

But this is now just successor features in another basis: We have that

$$V\phi = (1-\gamma)\sum_{k=1}^{\infty}\gamma^k \mathbb{E}\left[\psi_{t+k}|\phi_t = \phi\right]. \tag{68}$$

In particular, plugging in V into equation Equation 65 then just includes the current timestep:

$$p_t = (1-\gamma)\sum_{k=1}^{\infty}\gamma^k \mathbb{E}\left[\psi_{t+k}|\phi_t\right]. \tag{69}$$

The situation for the recurrent net is maybe even simpler (cf. Fang et al. (2023)): Assume a network with update rules

$$p_t = (1-\gamma)(\mathrm{Id} - \gamma W)^{-1}\phi_t. \tag{70}$$

Then Equation 64 becomes

$$(1-\gamma)(\mathrm{Id} - \gamma W)^{-1}\mathbb{E}[\phi_{t+1}|\phi_t] = W(1-\gamma)(\mathrm{Id} - \gamma W)^{-1}\phi_t. \tag{71}$$

Since W and $(\mathrm{Id} - \gamma W)^{-1}$ commute, it is again not hard to see that a solution is

$$W = \Phi P^T \Phi^{-1}, \tag{72}$$

and hence

$$p_t = (1-\gamma)(\mathrm{Id} - \gamma \Phi P^T \Phi^{-1})^{-1} = (1-\gamma)\sum_{k=0}^{\infty}\gamma^k \mathbb{E}\left[\phi_{t+k}|\phi_t\right]. \tag{73}$$

Now finally, what happens when the input to the feedforward-layer itself is also a successor representation, under the same prediction horizon γ, i.e. $\Phi = \tilde{\Phi}(\mathrm{Id} - \gamma P^T)^{-1}$? In this case the matrix V takes the particularly easy form $V = \Psi P^T \Phi^{-1}$, which is just the one step prediction (analogous to the recurrent case above). Still in this case p_t will encode successor features, just conditional on the basis $\tilde{\Phi}$.

These limits give the right intuition, we give a more formal analysis of the convergence below.

B Backward shifts of successor features

Here we want to prove backward shifting of Successor features which are Gaussian, but let us first make some general remarks. Recall that we are considering a feature $\phi : \mathbb{R} \to \mathbb{R}$ and it's center of mass, defined as the normalized mean. Since the feature in our case is obtained as inputs from other neurons, it makes sense to assume that ϕ is non-negative. Without loss of generality, we also assume that the feature is normalized, i.e. ϕ is a probability density, and the center of mass simply becomes $\int x\phi(x)dx = \mathbb{E}_\phi[x]$. Furthermore, when we normalize

the successor operator (which is simply the Laplace transform) by multiplying by γ, that is letting

$$SF(\phi)(x) = \int_0^\infty \int \gamma e^{-\gamma t} \phi(y) p_t(y|x) dy dt, \tag{74}$$

then we have that $SF(\phi)(x) = \mathbb{E}_{q(y|x)}[\phi(y)]$, with $q(y|x) = \int_0^\infty \gamma e^{-\gamma t} p_t(y|x) dt$ the density obtained from transforming $p_t(y|x)$ (one can think of this as the transition density from x to y when time t is sampled from an exponential distribution). In general, it might be difficult to study the centre of mass, because for example SF is not necessarily normalized anymore. However, the case we consider is more benign. Assume that we have Gaussian features, that is

$$\phi(x) = \mathcal{N}(x; \mu, \sigma). \tag{75}$$

Hence, the centre of mass is simply the mean μ. Let the dynamics be a Brownian motion with a constant drift, that is

$$p_t(y|x) = \mathcal{N}\left(y; x + vt, \rho\sqrt{t}\right). \tag{76}$$

In particular, as a function of x we can write this as

$$\mathcal{N}\left(x; y - vt, \rho\sqrt{t}\right). \tag{77}$$

Then integrating the transition density and ϕ just yields the marginal distribution, which is

$$\int p_t(y|x)\phi(y) = \mathcal{N}\left(x; \mu - vt, \rho\sqrt{t} + \sigma\right). \tag{78}$$

Thus, the successor feature is

$$SF(\phi)(x) \int_0^\infty = \gamma e^{-\gamma t} \mathcal{N}\left(x; \mu - vt, \rho\sqrt{t} + \sigma\right) dt. \tag{79}$$

In particular, the centre of mass is

$$\int_0^\infty \gamma e^{-\gamma t}(\mu - vt) = \mu - \frac{v}{\gamma}. \tag{80}$$

Thus, we observe again a shift, which depends on the predictive time scale factor γ and the velocity v. In particular, if the velocity is zero, then no shift should be observed.

C Eigenvectors of the Successor Representation

The eigenvectors of the successor representation matrix $(\mathrm{Id} - \gamma P)^{-1}$ are of special importance in the theory both in computational neuroscience as well as in reinforcement learning. In the latter, these eigenvectors are used to construct a

natural basis for functions on the state space, which yield a representation of the large-scale geometry of the state space and can be used for example for option discovery (Machado et al., 2017b,a; Mahadevan and Maggioni, 2007). In neuroscience in turn, these eigenvectors have been used as a model of grid cells, since in spatial settings there are remarkable similarities of some of the eigenvectors to grid cells observed in the entorhinal cortex Stachenfeld et al. (2014, 2017). The key insight to why these eigenvectors are particularly useful to represent the state space is that they are essentially the eigenvectors of a Laplace operator Stachenfeld et al. (2014). On an undirected graph with adjacency matrix A, the random-walk Laplacian is defined as

$$\mathrm{Id} - D^{-1}A \tag{81}$$

where $D_{ii} = \frac{1}{\sum_j A_{ij}}$. There are other ways to define a Laplacian (Chung, 1996), but on an undirected graph they are all related (Jost and Mulas, 2019). In Equation 81, the matrix $P := D^{-1}A$ corresponds to the transition probability matrix of a random walk that uniformly selects among all edges of a Graph. It is then not hard to see that the random walk Laplacian and the successor representation under P share the same eigenvectors (although for different eigenvalues). These eigenvectors and associated eigenvalues have many desirable properties: they are invariant under certain symmetries (automorphisms) of the graph Mahadevan and Maggioni (2007), they can be used to cluster the graph (Von Luxburg, 2007) and they represent slowly varying/smooth features Sprekeler (2011). Many of the technical results on graph Laplacians hinge on self-adjointness (i.e., symmetry), which is why they do not directly translate to the setting of directed graphs. In this situation, it is thus common to include some form of symmetrization - in fact, from the beginning it was proposed to use Laplacian eigenfunctions under some sort of symmetrization (Mahadevan and Maggioni, 2007). There are at least two straightforward ways how to do this: either, one can treat each directed edge as an undirected edge - this corresponds to constructing a new adjacency matrix $\frac{1}{2}(A + A^T)$, and was for example used in Mahadevan and Maggioni (2007) to obtain basis functions for RL. The other possibility is to use the information from the stationary distribution, and symmetrize as $\frac{1}{2}(P + \Pi^{-1}P^T\Pi)$ - this leads to the so called 'Chung-Laplacian' (Chung, 2005); also this Laplacian construction has been successfully applied in RL (Johns and Mahadevan, 2007; Wu et al., 2018). We have seen that our symmetrized learning rule in the stationary setting will learn the successor representation under $P + \Pi^{-1}P^T\Pi$, and therefore eigenvalues constructed from it would correspond to those of the Chung-Laplacian. However, in practice stationarity might not be a valid assumption - this is for example the case in our simulations, where episodes are terminated on reward. In this case, the symmetric representation that is learned is actually more closely related to the former case.

D Convergence proofs

In the following, we will give convergence proofs for both the neural network model and the TD-learning setting. In both situations, we will use the same strategy as in Dayan and Sejnowski (1994). Therein, he uses a classical result from stochastic approximation theory (Kushner and Clark, 2012). We state now a formulation of this result close to the one used in Dayan and Sejnowski (1994), but note that this omits some detail.

Theorem 1. *Let $Z_n \in \mathbb{R}^n$ be a sequence of random variables, which fulfill*

$$Z_{n+1} = Z_n + \varepsilon_n k(Z_n), \tag{82}$$

where k is a stochastic function of Z mapping to $\mathbb{R}^n$, and $\varepsilon_n \in (0,1)$ is a sequence such that $\sum_n \alpha_n = \infty, \sum_n \alpha_n^2 < \infty$. Define $\overline{k} : z \mapsto \mathbb{E}[k(z)]$, assume that Z_n is bounded almost surely and that $\mathrm{Var}[k(Z_n) - \overline{k}(Z_n)]$ is bounded. Let A^0 be the set of asymptotically stable equillibria of the differential equation

$$\frac{d}{dt} z(t) = \overline{k}(z). \tag{83}$$

Then if A^0 is nonempty, $Z_n \to A^0$.

As in Dayan and Sejnowski (1994), we will not verify the boundedness conditions, because these could be enforced by projecting the sequence back into a bounded region if necessary - i.e., one could modify the update equations to include a projection step that enforces staying in a certain bounded region. Then, all quantities above are bounded and hence this new sequence converges to A^0 if A^0 is in that region.

D.1 Convergence of Model

In the following, to avoid using two many superscripts, we change notation compared to the main text, and set $W = W^1$ the recurrent weight,$V = W^2$ the feedforward weight, $p = p^1, q = p^2$ the population activities, $\phi = \phi^1, \psi = \phi^2$ the inputs. Furthermore, we use the notation

$$R(W) = (Id - \gamma W)^{-1}$$

and recall our definition

$$P_{\alpha,\beta} = \frac{\alpha}{\alpha+\beta} P^T + \frac{\beta}{\alpha+\beta} \Pi P \Pi^{-1}.$$

Consider the update rule for the system

$$W_{t+1} = W_t + \varepsilon_t \Delta_{W,t} \tag{84}$$

$$V_{t+1} = V_t + \varepsilon_t \Delta_{V,t} \tag{85}$$

$$\Delta_{W,t} = (\alpha_W (p_{t+1} - W_t p_t) p_t^T + \beta_W (p_t - W_t p_{t+1}) p_{t+1}^T). \tag{86}$$

$$\Delta_{V,t} = (\alpha_V (q_{t+1} - V_t p_t) p_t^T + \beta_V (q_t - V_t p_{t+1}) p_{t+1}^T). \tag{87}$$

As stated above, we can show convergence to W^*, V^* if we can show that the differential equation

$$\frac{d}{dt}\begin{pmatrix} W \\ V \end{pmatrix} = \mathbb{E}\left[\begin{pmatrix} \Delta(W) \\ \Delta(V) \end{pmatrix}\right] \tag{88}$$

has an asymptotically stable fixed point at W^*, V^* - to do so, we will study eigenvalues of the jacobian at putative fixed points.

We have approximately (assuming for the expectation that $W_{t+1} \approx W_t$, $V_{t+1} \approx V_t$):

$$\Delta(W) = \alpha_W \left(R(W)\phi_{t+1} - WR(W)\phi_t\right)\phi_t^T R(W)^T + \tag{89}$$

$$\beta_W \left(R(W)\phi_t - WR(W)\phi_{t+1}\right)\phi_{t+1}^T R(W)^T \tag{90}$$

and (recall $q = \gamma_V V p + u$)

$$\Delta(V) = \alpha_V \left(V\left(\gamma_V R(W)\phi_{t+1} - R(W)\phi_t\right) + \psi_{t+1}\right)\phi_t^T R(W) + \tag{91}$$

$$\beta_V \left(\gamma_V V\left(R(W)\phi_t - R(W)\phi_{t+1}\right) + \psi_t\right)\phi_{t+1}^T R(W)^T \tag{92}$$

We now assume that the inputs ψ_t, ϕ_t are in stationary distribution (we state below a condition under which this holds) and then take the expectation to obtain

$$\mathbb{E}[\Delta(W)] = \left(R(W)\left(\alpha\mathbb{E}[\phi_1\phi_0^T]] + \beta\mathbb{E}[\phi_0\phi_1^T]] - (\alpha+\beta)W\mathbb{E}[\phi_0\phi_0^T]\right)R(W)^T \tag{93}$$

$$\mathbb{E}[\Delta(V)] = VR(W)\left(\gamma_V(\alpha_V\mathbb{E}[\phi_1\phi_0^T] + \beta_V\mathbb{E}[\phi_0\phi_1^T]) - (\alpha_V+\beta_V)\mathbb{E}[\phi_0\phi_0^T]\right)R(W)^T \tag{94}$$

$$+ (\alpha_V\mathbb{E}[\psi_1\phi_0^T] + \beta_V\mathbb{E}[\psi_0\phi_1^T])R(W)^T \tag{95}$$

where we have used that W and $R(W)$ commute. For the first equation, setting the innermost term zero, we see that this equation has an equilibrium if

$$W^*\mathbb{E}[\phi_0\phi_0^T] = \frac{\alpha_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_1\phi_0^T] + \frac{\beta_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_0\phi_1^T] \tag{96}$$

We see that this equation essentially prescribes W to implement a weighted average of the linear regression of ϕ_0 against ϕ_1 and vice versa. Indeed, if $\mathbb{E}[\phi_0\phi_0^T]$ is invertible one sees that it becomes

$$W^* = \frac{\alpha_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_1\phi_0^T]\mathbb{E}[\phi_0\phi_0^T]^{-1} + \frac{\beta_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_0\phi_1^T][\phi_1\phi_1^T]^{-1}. \tag{97}$$

Setting the second equation to zero yields the relation

$$V^*R(W)\left(\mathbb{E}[\phi_0\phi_0^T] - \gamma_V\left(\frac{\alpha_V}{\alpha_V+\beta_V}\mathbb{E}[\phi_1\phi_0^T] + \frac{\beta_V}{\alpha_V+\beta_V}\mathbb{E}[\phi_0\phi_1^T]\right)\right) = \frac{\alpha_V}{\alpha_V+\beta_V}\mathbb{E}[\psi_1\phi_0^T] + \frac{\beta_V}{\alpha_V+\beta_V}\mathbb{E}[\psi_0\phi_1^T]. \tag{98}$$

We now want perform a stability analysis of the equations at an equillibrium (W^*, V^*). Since the dynamics of W do not depend on V, we can look at the respective equations separately and if both are stable, the whole system is stable. The easier part is the equation for V: It is affine linear in V, that is we only have to study the eigenvalues of the matrix

$$R(W^*)\left(\gamma_V(\alpha_V\mathbb{E}[\phi_1\phi_0^T] + \beta_V\mathbb{E}[\phi_0\phi_1^T]) - (\alpha_V + \beta_V)\mathbb{E}[\phi_0\phi_0^T]\right)R(W^*)^T \tag{99}$$

In fact, using 2, we only have to study the eigenvalues of $\left(\gamma_V(\alpha_V\mathbb{E}[\phi_1\phi_0^T] + \beta_V\mathbb{E}[\phi_0\phi_1^T]) - (\alpha_V + \beta_V)\mathbb{E}[\phi_0\phi_0^T]\right)$. According to 5, these have all negative real parts, hence we have stability for this equation at any equilibrium.

The stability of the differential equation for W is slightly more delicate. Let F denote the right hand side of the differential equation for W. The derivative of F (in the Frechet sense) at W^*, applied to a matrix U becomes

$$DF_{W^*}(U) = -R(W^*)\gamma U R(W^*)\left(\alpha\mathbb{E}[\phi_1\phi_0^T] + \beta\mathbb{E}[\phi_0\phi_1^T] - (\alpha+\beta)W^*\mathbb{E}[\phi_0\phi_0^T]\right)R(W^*) \tag{100}$$

$$- R(W^*)\left(\alpha\mathbb{E}[\phi_1\phi_0^T] + \beta\mathbb{E}[\phi_0\phi_1^T] - (\alpha+\beta)W^*\mathbb{E}[\phi_0\phi_0^T]\right)R(W^*)^T\gamma U^T R(W^*)^T \tag{101}$$

$$- R(W^*)(\alpha+\beta)U\mathbb{E}[\phi_0\phi_0^T]R(W^*)^T \tag{102}$$

$$= -R(W^*)(\alpha+\beta)U\mathbb{E}[\phi_0\phi_0^T]R(W^*)^T, \tag{103}$$

where the first two terms vanished at W^* by the defining relation of the latter. Since we have the identity

$$\mathrm{vec}(ABC) = (C^T \otimes A)\,\mathrm{vec}(B), \tag{104}$$

where $\otimes$ denotes the Kronecker product and vec denotes vectorization, the stability is determined by the eigenvalues of the matrix

$$M = -(\alpha+\beta)R(W^*)\mathbb{E}[\phi_0\phi_0^T] \otimes R(W^*). \tag{105}$$

Since for the Kronecker-product of two matrices, the eigenvalues are given as the products of the eigenvalues of the individual matrices, there seems to be in general *no* guarantee that the matrix M will be stable for all choices of α, β, ϕ.

However, we can obtain a results in the symmetric case, under some assumptions:

Proposition 1. *For $\alpha = \beta > 0$, $\mathbb{E}[\phi_0\phi_0^T]$ invertible, and the real part of any eigenvalue μ of W^* fulfilling $\mathrm{Re}(\mu) \leq 1$ the matrix M is stable, that is all eigenvalues have negative real parts.*

Proof. First, we note that by the defining relation of W^* we obtain that

$$R(W^*)\mathbb{E}[\phi_0\phi_0^T] = R(W^*)\left(\mathbb{E}[\phi_0\phi_0^T] - \gamma\left(\frac{\alpha_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_1\phi_0^T] + \frac{\beta_W}{\alpha_W+\beta_W}\mathbb{E}[\phi_0\phi_1^T]\right)\right)^T R(W^*)^T. \tag{106}$$

According to 5 and 2, the eigenvalues of the right hand side have positive real parts, and for $\alpha = \beta$, the matrix is symmetric hence the eigenvalues are positive reals. On the other hand, the eigenvalues of the matrix $R(W)^* = (\mathrm{Id} - \gamma W^*)^{-1}$ have positive real parts by assumption and 3. Thus, also the products of the eigenvalues of the two matrices have positive real parts, which proves the claim. □

We have thus confirmed that in the symmetric case, it is possible to have stability (and thus convergence) at some (W^*, V^*). Below, we will now assume a specific structure of the inputs which will link to successor features and which will make it possible to state the existence of a solution. We will now assume

$$\phi_t = \tilde{\phi}(S_{t+\tau_1}) = \Phi e_{S_{t+\tau_1}} \tag{107}$$

$$\psi_t = \tilde{\psi}(S_{t+\tau_2}) = \Psi e_{S_{t+\tau_2}} \tag{108}$$

where S_t is the state process, which we assume in stationary distribution, $\tilde{\phi}, \tilde{\psi}$ are injective functions, the matrices Ψ, Φ have the respective values of these functions as their columns, The τ_i indicate that the two inputs could have a temporal offset, we will however set these to zero in the sequel. We note at these points that these choices are made to relate to the definition successor features, and other choices of input could equally yield a similar result.

With the above definitions, we have

$$\mathbb{E}[\phi_0 \phi_0^T] = \Phi \Pi \Phi^T \tag{109}$$

$$\mathbb{E}[\phi_1 \phi_0^T] = \Phi P^T \Pi \Phi^T \tag{110}$$

$$\mathbb{E}[\psi_1 \phi_0^T] = \Psi P^T \Pi \Phi^T \tag{111}$$

where P is the transition matrix of S_t and Π is again the diagonal matrix of the stationary distribution. With this, it is easy to see that all possible solutions for W^* are of the form

$$W^* = \Phi P_{\alpha_W, \beta_W} \Phi^- + C \tag{112}$$

where Ψ^- denotes the pseudoinverse, and C is a matrix with $C\Psi = 0$. On the other hand, all possible solutions for V^* are of the form

$$V^* = \left(\Psi P^T_{\alpha_V, \beta_V} (\mathrm{Id} - \gamma_V P^T_{\alpha_W, \beta_W})^{-1} \Phi^- + C \right) R(W^*)^{-1} \tag{113}$$

where C is again a matrix having Φ in its kernel.

We note that with this particular form of W^*, we can indeed achieve stable solutions: $\Phi P_{\alpha_W, \beta_W} \Phi^-$ has eigenvalues either being zero or corresponding to eigenvalues of P_{α_W, β_W}, which is a stochastic matrix and hence has eigenvalues in the unit circle, such that as long as the matrix C has eigenvalues with real part lesser or equal than 1, the requirements of our proposition above are met and we have a stable equilibrium. In the particular special case where Φ is invertible, we get unique stable solutions.

D.1.1 Limits of activities

With these particular solutions, we can understand the form the activities of the populations will take. We will plug in the limits of the weight matrices into the equations

$$p_t = (\mathrm{Id} - \gamma_W W^*)^{-1} \phi_t \tag{114}$$

$$q_t = \gamma_V V^* p_t + \psi_t \tag{115}$$

First, we recall that we assume inputs of the form $\phi_t = \phi(S_t)$. Second, we note that the action of the matrices on the space spanned by Φ is the same for all solutions W^*, regardless of the additional matrix C. In particular, we have that

$$\Phi P_{\alpha,\beta^T} \Phi^- \phi_t = \mathbb{E}_{P_{\alpha,\beta^T}}[\phi_{t+1}|\phi_t] = \mathbb{E}_{[P_{\alpha,\beta^T}]}[\phi_{t+1}|(S_t)]. \tag{116}$$

This then yields

$$p_t = \left(\mathrm{Id} - \gamma_W \Phi P^T_{\alpha_W,\beta_W} \Phi^-\right)^{-1} \phi_t = \sum_k \gamma_W^k \mathbb{E}_{P_{\alpha,\beta}}[\phi(S_{t+k})|S_t] \tag{117}$$

which is the successor representation of the function ϕ. On the other hand, note that

$$\mathrm{Id} + \gamma_V P^T_{\alpha_V,\beta_V} \left(\mathrm{Id} - \gamma_V P^T_{\alpha_V,\beta_V}\right)^{-1} = \left(\mathrm{Id} - \gamma_V \ P^T_{\alpha_V,\beta_V}\right)^{-1} \tag{118}$$

and that $\Phi^- \phi_t = \Psi^- \psi_t$ by definition. Using this, we have

$$q_t = \gamma_V \Psi P^T_{\alpha_V,\beta_V} \left(\mathrm{Id} - \gamma_V P^T_{\alpha_V,\beta_V}\right)^{-1} \Phi^- \phi_t + \psi_t = \Psi \left(\mathrm{Id} + \gamma_V P^T_{\alpha_V,\beta_V} \left(\mathrm{Id} - \gamma_V P^T_{\alpha_V,\beta_V}\right)^{-1}\right) \Psi^- \psi_t \tag{119}$$

$$= \Psi \left(\mathrm{Id} - \gamma_V \ P^T_{\alpha_V,\beta_V}\right)^{-1} \Psi^- \psi_t = \sum_k \gamma_V^k \mathbb{E}_{P_{\alpha,\beta}}[\psi(S_{t+k})|S_t] \tag{120}$$

E Convergence of TD learning

Here we show in a very similar manner that classical TD-learning with the inclusion of parameters α, β learns the successor representation under the modified probabilities. In fact, one can see classical TD-learning as a one layer feedforward network implementing the learning rule/

Consider the following modified TD-learning rule for the parameter θ_t parametrizing $\tilde{V}$

$$\theta_{t+1} = \theta_t + \varepsilon_t \Delta_t \tag{121}$$

$$\Delta_t = \alpha \left(\phi(S_t) + \gamma \tilde{V}(\theta_t, S_{t+1}) - \tilde{V}(\theta_t, S_t)\right)^T \nabla_\theta \tilde{V}(\theta_t, S_t) \tag{122}$$

$$+ \beta \left(\phi(S_{t+1}) + \gamma \tilde{V}(\theta_t, S_t) - \tilde{V}(\theta_t, S_{t+1})\right)^T \nabla_\theta \tilde{V}(\theta_t, S_{t+1}). \tag{123}$$

We can see this as a semi-gradient descent with respect to the loss

$$L(s,s',\theta,\theta') = \alpha||\phi(s)+\gamma\tilde{V}(\theta',s')-\tilde{V}(\theta,s)||^2+\beta||\phi(s')+\gamma\tilde{V}(\theta',s)-\tilde{V}(\theta,s')||^2. \tag{124}$$

Now assume the linear and tabular case, that is

$$\tilde{V}(W,s) = We_s, \phi(S) = \Phi e_s \tag{125}$$
$$W_{t+1} = W_t + \varepsilon_t \Delta_t \tag{126}$$
$$\Delta_t = \alpha\left(\Phi e_{S_t} + \gamma W_t e_{S_{t+1}} - W_t e_{S_t}\right) e_{S_t}^T \tag{127}$$
$$+ \beta\left(\Phi e_{S_{t+1}} + \gamma W_t e_{S_t} - W_t e_{S_{t+1}}\right) e_{S_{t+1}}^T \tag{128}$$

Note that we can also write the updates as

$$W_{t+1}(ij) = (1-\epsilon_t(ij))W_t(ij) \tag{129}$$
$$+ \varepsilon_t(ij)\left(\alpha\left(\Phi e_{S_t} e_{S_t}^T + \gamma W_t e_{S_{t+1}} e_{S_t}^T\right) + \beta\left(\Phi e_{S_{t+1}} e_{S_{t+1}}^T + \gamma W_t e_{S_t} e_{S_{t+1}}^T\right)\right)(ij) \tag{130}$$

where $\epsilon_t(ij) = \varepsilon_t(\alpha\mathbb{I}[S_t = j] + \beta\mathbb{I}[S_{t+1} = j])$.

We have

$$\mathbb{E}[\Delta(W)] = \mathbb{E}[\alpha\left(\Phi e_{S_t} + \gamma W e_{S_{t+1}} - W e_{S_t}\right) e_{S_t}^T \tag{131}$$
$$+ \beta\left(\Phi e_{S_{t+1}} + \gamma W e_{S_t} - W e_{S_{t+1}}\right) e_{S_{t+1}}^T.] \tag{132}$$
$$= \alpha(\Phi\Pi + \gamma W P^T \Pi - W\Pi) \tag{133}$$
$$+ \beta(\Phi\Pi + \gamma W \Pi P - W\Pi) \tag{134}$$
$$= (\alpha+\beta)\left(\Phi - W\left(\mathrm{Id} - \gamma\left((\frac{\alpha}{\alpha+\beta}P^T + \frac{\beta}{\alpha+\beta}\Pi P \Pi^{-1}\right)\right)\right)\Pi. \tag{135}$$

Setting to zero, we obtain

$$W^* = \Phi\left(\mathrm{Id} - \gamma\left((\frac{\alpha}{\alpha+\beta}P^T + \frac{\beta}{\alpha+\beta}\Pi P \Pi^{-1}\right)\right)^{-1}, \tag{136}$$

which is precisely the successor representation under $P_{\alpha,\beta}$. The stability of the differential equation, which is linear, is determined by the eigenvalues of the matrix

$$M = \left(\mathrm{Id} - \gamma\left(\frac{\alpha}{\alpha+\beta}P^T + \frac{\beta}{\alpha+\beta}\Pi P \Pi^{-1}\right)\right)\Pi \tag{137}$$

Using again lemma 5 below this has all positive eigenvalues, which proves the stability and hence the convergence.

F Auxiliary results

Here we state some lemmas we used for the convergence proofs above, these should all be standard results, but we prove them here for convenience.

Lemma 1. *Let $M \in \mathbb{R}^{n\times n}$ (not necessarily symmetric). Then if M is positive definite in the sense that for any $0 \neq x \in \mathbb{R}^n$, $x^T M x > 0$, all eigenvalues of M have positive real part.*

Proof. Let $\lambda \in \mathbb{C}$ be an eigenvalue of M with eigenvector $z \in \mathbb{C}^n$. Then on the one hand we have

$$\langle z, Mz\rangle_{\mathbb{C}^n} = \langle z, \lambda z\rangle_{\mathbb{C}^n} = \lambda\langle z, z\rangle_{\mathbb{C}^n}, \tag{138}$$

and on the other hand

$$\mathrm{Re}\left(\langle z, Mz\rangle_{\mathbb{C}^n}\right) = \sum_{i,j} \mathrm{Re}(z_i) M_{ij}\, \mathrm{Re}(z_j) + \sum_{i,j} \mathrm{Im}(z_i) M_{ij}\, \mathrm{Im}(z_j) \tag{139}$$

Now since $\langle z, z\rangle_{\mathbb{C}^n} > 0$, if M is positive definite, the right hand side of the last equation is positive and this implies that $Re(\lambda) > 0$. □

Lemma 2. *Let A, B be in $\mathbb{R}^{n\times n}$ and B be invertible. Then A is positive definite in the sense of the previous lemma if and only if so is BAB^T.*

Proof. Since B is invertible,

$$x^T A x > 0 \forall x \iff (By)^T A (By) = y^T B^T A B y > 0 \forall y. \tag{140}$$

□

Lemma 3. *Let Q be a matrix $Re(\lambda) < 1$ for all its eigenvalues $\lambda \in \mathbb{C}$. Then $Id - Q$ has eigenvalues with all positive real parts, is invertible, and the inverse also has eigenvalues with all positive real parts.*

Proof. The eigenvalues of $Id - Q$ are of the form $1-\lambda$. Hence we have $\mathrm{Re}(1-\lambda) > 0$. This then implies that $Id - Q$ is invertible. The last part follows since inversion maps the positive half-plane to the positive half-plane. □

Lemma 4. *Let A be a square matrix and A^* be it's adjoint with respect to some inner product $\langle\cdot,\cdot\rangle$. Then we have, denoting by σ the spectrum,*

$$\max_{\lambda\in\sigma(A)} Re(\lambda) \leq \max_{\mu\in\sigma(\frac{1}{2}(A+A^*))} \mu. \tag{141}$$

Proof. We note that for an eigenpair (v, λ) of A

$$\langle v, (A + A^*)v\rangle = \langle v, Av\rangle + \langle Av, v\rangle = (\lambda + \overline{\lambda})\langle v, v\rangle = 2Re(\lambda)\langle v, v\rangle \tag{142}$$

Hence, by the Courant-principle

$$Re(\lambda) = \frac{\langle v, \frac{1}{2}(A + A^*)v\rangle}{\langle v, v\rangle} \leq \max_x \frac{\langle x, \frac{1}{2}(A + A^*)x\rangle}{\langle x, x\rangle} = \max_{\mu\in\sigma(\frac{1}{2}(A+A^*)} \mu. \tag{143}$$

□

Corollary 1. *Let $t \in \mathbb{R}$, P a stochastic matrix with stationary distribution π, and $\Pi = \mathrm{diag}(\pi)$. The matrix*

$$Q = tP + (1-t)\Pi^{-1}P^T\Pi \tag{144}$$

has $Re(\lambda) \leq 1$ for all its eigenvalues $\lambda \in \mathbb{C}$.

Proof. Consider the inner product $\langle x, x\rangle_\Pi = x^T\Pi x$. With respect to this inner product, the adjoint of Q is

$$Q^* = (1-t)P + t\Pi^{-1}P^T\Pi. \tag{145}$$

Hence,

$$\frac{1}{2}(Q + Q^*) = \frac{1}{2}(P + \Pi^{-1}P^T\Pi) \tag{146}$$

which is a stochastic matrix and therefore has largest eigenvalue 1. □

Lemma 5. *Let ϕ_0, ϕ_1 be random vectors distributed equally. Let furthermore $t \in \mathbb{R}$ and $\gamma \in [0, 1)$. Then the matrix*

$$M = \mathbb{E}[\phi_0\phi_0^T] - \gamma\left(t\mathbb{E}[\phi_1\phi_0^T] + (1-t)\mathbb{E}[\phi_0\phi_1^T]\right) \tag{147}$$

has eigenvalues with all nonnegative real parts. If furthermore $\mathbb{E}[\phi_0\phi_0^T]$ has full rank, then the real parts are all positive.

Proof. Let x be any nonzero vector. Note that we have

$$x^T\mathbb{E}[(\phi_1 - \phi_0)(\phi_1 - \phi_0)^T]x = x^T\,\mathrm{Cov}[\phi_1 - \phi_0]x \geq 0, \tag{148}$$

hence

$$2x^T\mathbb{E}[\phi_0\phi_0^T]x = x^T\mathbb{E}[\phi_0\phi_0^T] + x^T\mathbb{E}[\phi_1\phi_1^T]x^T \geq x^T\mathbb{E}[\phi_1\phi_0^T]x + x^T\mathbb{E}[\phi_0\phi_1^T]x = 2x^T\mathbb{E}[\phi_0\phi_1^T]x. \tag{149}$$

From this follows $x^TMx \geq 0$ with equality only if x is in the kernel of $\mathbb{E}[\phi_0\phi_0^T]$ and hence the claim. □

G Symmetrized distribution is closer to uniform distribution

It is intuitive, that a reversible distribution of transitions is closer to the uniform distribution. The Kullback-Leibler divergence yields a particularly easy way to prove this intuition. Assume our state space $\mathcal{S}$ is a undirected graph $G = (\mathcal{S}, E)$, that is, we are in a deterministic setting, and if one can go from s_1 to s_2 then one can also go back. Then, together with the stationary distribution $\pi(s)$, the transition probabilities $p(s|s')$ define a joint probability on the set of edges (with orientation) of the graph, i.e.

$$p((s_1, s_2)) = p(s_2|s_1)\pi(s_1), \tag{150}$$

with (s_1, s_2) denoting the directed edge from s_1 to s_2. That is, if we take the transition probabilities

$$tp(s_2|s_1) + (1-t)p(s_1|s_2)\frac{\pi(s_2)}{\pi(s_1)}, \tag{151}$$

then the joint distribution would be simply

$$tp(s_2|s_1)\pi(s_1) + (1-t)p(s_1|s_2)\frac{\pi(s_2)}{.} \tag{152}$$

Now, if we were to observe transitions uniformly, then we have

$$p((s_1, s_2)) = \mathbb{I}[(s_1, s_2) \in E]\frac{1}{2|E|} =: q((s_1, s_2)), \tag{153}$$

where the factor 2 appears because we consider each undirected edge twice. Note that this is the joint probability that is obtained when one classically defines the random walk on an undirected graph with

$$p(s_2|s_1) = \mathbb{I}[(s_1, s_2) \in E]\frac{1}{deg(s_1)} \tag{154}$$

with $deg(s_1) = \sum_{s'} \mathbb{I}[(s_1, s') \in E]$ the degree. Indeed, weighting the above equation with the stationary distribution $\pi(s) = \frac{deg(s)}{2|E|}$ yields the uniform distribution over edges.

Now, the Kullback-Leibler divergence between any probability distribution and the uniform distribution is just

$$\mathrm{D_{KL}}(p||q) = \sum_{(s_1,s_2)} \ln p((s_1, s_1))p((s_1, s_2) + c, \tag{155}$$

with c not depending on c -i.e, it is just given by the negative entropy of p. Hence, the following lemma shows that indeed choosing $t = \frac{1}{2}$ in Equation 151 leads us closest to uniform.

Lemma 6. *Let $X \in \mathcal{X}$ be a random variable with density p, where $\mathcal{X}$ admits an involution, that is a bijective, measure-preserving map $\sigma : \mathcal{X} \to \mathcal{X}$ such that $\sigma^2 = Id$. Let $H(\cdot)$ be the entropy. Then for any $\frac{1}{2} \neq t \in [0, 1]$:*

$$H(\frac{1}{2}(p + p \circ \sigma)) \geq H(tp + (1-t)p \circ \sigma) \tag{156}$$

and equality holds iff both densities are equal.

Proof. By the strict concavity of entropy and the convexity of the set $\{tp + (1-t)p \circ \sigma | t \in [0, 1]\}$, there is a unique maximum. Differentiating with respect to t yields

$$\frac{d}{dt}H(tp + (1-t)p \circ \sigma) = \frac{d}{dt}\int_{\mathcal{X}} tp + (1-t)p \circ \sigma \ln tp + (1-t)p \circ \sigma d\mu \tag{157}$$

$$= 2\int_{\mathcal{X}} (p - p \circ \sigma) \ln tp + (1-t)p \circ \sigma d\mu \tag{158}$$

Now at $t = \frac{1}{2}$, the term $\ln \frac{1}{2}(p + p \circ \sigma)$ is invariant under σ, while the term $p - p \circ \sigma$ is skew-symmetric under σ. Hence, the integral vanishes. $\square$

G.1 Proof that symmetrization of optimal policy is stable

Here, we want to show that an optimal policy remains optimal even if the value function is computed under symmetrized transition probabilities. Before we do so, we will however first state an useful identity for the successor representations.

Definition 1. *Under a fixed policy π, for S_t a random walk with transition probabilities $P(s'|s) = \sum_a p(s'|a,s)\pi(a|s)$, we define the first hitting times*

$$\tau^s_{k+1} = \inf\{t > \tau^s_k \in \mathbb{N} | S_t = s\}, \tau^s_0 = 0, \tag{159}$$

where if the case that the set is empty we set the value to ∞. For a given pair of states s, s' we define

$$\mathcal{T}(s'|s) = \sup\{k \geq 0 | \mathbb{P}[\tau^{s'}_k < \infty | S_0 = s] > 0\}. \tag{160}$$

Note that $\mathcal{T}(s'|s)$ only takes the values $\{0, 1, \infty\}$.

It is easy to see that by, the Markovian property, the distribution of these stopping times does not change, in the sense that

$$\tau^s_{k+1} - \tau^s_k | \tau^s_k < \infty \sim \tau^s_1, \tag{161}$$

where for the left hand side we used the convention $\infty - c = \infty$ for any finite c. Using this, we have the following expression for the Successor representation in terms of stopping times:

Proposition 2. *For the successor representation $M = \big(Id - \gamma P)^{-1}\big)$ we have*

$$M_{ss'} = \delta(s,s') + \begin{cases} 0, \mathcal{T}(s'|s) = 0 \\ \mathbb{E}[\gamma^{\tau^{s'}_1} | S_0 = s], \mathcal{T}(s'|s) = 1 \\ \frac{\mathbb{E}[\gamma^{\tau^{s'}_1}] | S_0 = s]}{1 - \mathbb{E}[\gamma^{\tau^{s'}_1}] | S_0 = s']}, \mathcal{T}(s'|s) = \infty \end{cases} \tag{162}$$

Proof. We note that

$$\begin{aligned} M_{ss'} &= \big(Id - \gamma P)^{-1}\big)_{ss'} && (163) \\ &= \sum_{t=0}^{\infty} \gamma^t P^t_{ss'} && (164) \\ &= \sum_{t=0}^{\infty} \gamma^t \mathbb{P}[S_t = s' | S_0 = s] && (165) \\ &= \sum_{t=0}^{\infty} \gamma^t \sum_{k=0}^{\infty} \mathbb{P}[S_t = s', \tau^{s'}_k = t | S_0 = s] && (166) \\ &= \sum_{t=0}^{\infty} \gamma^t \sum_{k=0}^{\mathcal{T}(s'|s)} \mathbb{P}[S_t = s', \tau^{s'}_k = t | S_0 = s] && (167) \end{aligned}$$

where we have simply included a sum over all possible values for the hitting times and then included only the cases where those are finite. In particular, if $\mathcal{T}(s'|s) = 0$, we have $M(s'|s) = 0$.

Now note that in the above sum

$$\mathbb{P}[S_t = s', \tau_k^{s'} = t|S_0 = s] = \begin{cases} 1, k = t = 0, s = s' \\ \mathbb{P}[\tau_k^{s'} = t|S_0 = s], else. \end{cases} \tag{168}$$

Hence we get

$$= \delta(s, s') + \sum_{t=1}^{\infty} \gamma^t \sum_{k=1}^{\mathcal{T}(s'|s)} \mathbb{P}[\tau_k^{s'} = t \| S_0 = s] \tag{169}$$

$$= \delta(s, s') + \sum_{k=1}^{\mathcal{T}(s'|s)} \sum_{t=0}^{\infty} \gamma^t \mathbb{P}[\tau_k^{s'} = t|S_0 = s] \tag{170}$$

$$= \delta(s, s') + \sum_{k=1}^{\mathcal{T}(s'|s)} \mathbb{E}\left[\gamma^{\tau_k^{s'}} |S_0 = s\right]. \tag{171}$$

Thus, in the case that $\mathcal{T}(s'|s) = 1$, we get

$$M_{ss'} = \delta(s, s') + \mathbb{E}\left[\gamma^{\tau_1^{s'}} |S_0 = s\right]. \tag{172}$$

Now, for the remaining case $\mathcal{T}(s'|s) = \infty$ first note that

$$\gamma^{\tau_k^{s'}} = \prod_{l=0}^{k-1} \gamma^{\tau_{l+1}^{s'} - \tau_l^{s'}}, \tag{173}$$

and that for $l > 0$ we have $\tau_{l+1}^{s'} - \tau_l^{s'}$ is independent of S_0 and since at $\tau_l^{s'}, S_l = s'$, we have

$$\mathbb{E}\left[\tau_{l+1}^{s'} - \tau_l^{s'}\right] = \mathbb{E}\left[\tau_{l+1}^{s'} - \tau_l^{s'} |S_0 = s'\right] = \mathbb{E}\left[\tau_1^{s'} |S_0 = s'\right]. \tag{174}$$

Hence,

$$\mathbb{E}\left[\gamma^{\tau_k^{s'}} \| S_0 = s\right] = \mathbb{E}\left[\prod_{l=0}^{k-1} \gamma^{\tau_{l+1}^{s'} - \tau_l^{s'}}\right] \tag{175}$$

$$= \mathbb{E}\left[\gamma^{\tau_1^{s'}} |S_0 = s'\right]^{(k-1)\delta_{k>1}} \mathbb{E}\left[\gamma^{\tau_1^{s'}} |S_0 = s\right]^{(k-1)\delta_{k>0}}. \tag{176}$$

Plugging this into the sum formula yields

$$M_{ss'} = \delta(s,s') + \sum_{k=1}^{\infty} \mathbb{E}\left[\gamma^{\tau_k^{s'}} | S_0 = s\right] \tag{177}$$

$$\delta(s,s') + \mathbb{E}\left[\gamma^{\tau_1^{s'}} | S_0 = s\right] \left(\sum_{k=0}^{\infty} \mathbb{E}\left[\gamma^{\tau_1^{s'}} | S_0 = s\right]^k\right) \tag{178}$$

$$= \delta(s,s') + \mathbb{E}\left[\gamma^{\tau_1^{s'}} | S_0 = s\right] \left(1 - \mathbb{E}\left[\gamma^{\tau_1^{s'}} | S_0 = s\right]\right)^{-1} \tag{179}$$

$$\tag{180}$$

as claimed. □

Corollary 2. *In the setting with a unit reward e_{s^*}, the value function is completely determined by the expected γ to the power of the first hitting time, that is*

$$V(s) \geq V(\tilde{s}) \iff \mathbb{E}[\gamma^{\tau_1^{s^*}} | S_0 = s] \geq \mathbb{E}[\gamma^{\tau_1^{s^*}} | S_0 = \tilde{s}]. \tag{181}$$

In particular, consider the deterministic setting - that is, the transitions $p(s'|a,s)$ are deterministic and hence the state-space becomes a directed graph, where a directed edge exists from state s to state s' if there is an action a which (deterministically) leads from states s to state s'. The task to navigate to state s' is inherently finite/absorbing, but we can extend the time horizion to infinity by simply allowing an action that stays in the same state as one is (which hence is the optimal action once one is at the target state). In this deterministic setting, an optimal policy has to always decrease the distance to the target - hence, we have under an optimal policy, where the target is s^*,

$$(\tau_1^{s^*} | S_0 = s) = \delta(s \neq s^*) d(s,s^*) + \delta(s = s^*), \tag{182}$$

where $d(s,s')$ is the shortest number of steps necessary to get from s to s' (which is not necessarily symmetric). In particular, every non-target state s' should never be returned to, and starting from state s, only states lying on shortest paths from s to s^* should be visited (where some of those shortest paths may still not be visited by the policy, as there might be multiple such paths). In summary:

Corollary 3. *In the directed graph setting navigating to target s^*, for an optimal policy π^*, the successor representation becomes*

$$M_{ss'} = \delta(s,s') + \begin{cases} 0, s' \textit{not on a shortest path from } s \textit{ to } s^* \textit{ which the policy visits} \\ \gamma^{d(s,s')}, s' \neq s^* \\ \frac{\gamma^{d(s,s')}}{1-\gamma}, s' = s^* \end{cases} \tag{183}$$

Having understood this simple structure of the SR under an optimal policy, we can also understand why a symmetrization does not impede solving the

task. First, we have to slightly generalize our definition of the symmetrization of transition probabilities, as the one we used in our biological model only works in the case where the resulting Markov chain is ergodic. This is not the typical situation in navigation problems. Rather, in navigation tasks, under an optimal policy, one will navigate towards a goal on the shortest possible path and then typically the episode ends. Thus one could on the hand imagine an absorbing state at the target, but this then does not lend itself nicely to the generalization setting where the target should change but the environmental dynamics should stay the same. Instead, one could then not introduce an absorbing state in the environment transitions per se, and just include a self-transition at the target state - but still stop the episode accordingly when the reward is reached.

We thus would want to define the reverse process in this case as a process which always moves away from the target state s^*, until it reaches a state with maximal distance to the target, where it stays (or rather, the episode ends) - hence mirroring the behaviour of the optimal policies. That is formally, if P are the transition probabilities under the optimal policy, and $d_{\max} = \max_s d(s, s^*)$

$$P^{reverse}_{s,s'} = \begin{cases} \delta_{s,s'}, & d(s, s^*) = d_{\max} \\ P(S_{\tau_1^s - 1} = s' | \tau_1^s < \infty), & d(s, s^*) < d_{\max} \end{cases} \tag{184}$$

The symmetrized process is then the one with transition probabilities given by

$$q = \frac{1}{2}(P + P_{reverse}. \tag{185}$$

The value function induced by q for the navigational problem (i.e. reward vector e_{s^*}) is then

$$V_q^{s*}(s) = \sum_{k=0}^{\infty} \gamma^k q_k(s^*|s). \tag{186}$$

Recall that a policy π is optimal for a value function V if the policy always selects actions which maximize the value function. We have

Proposition 3. *Let π be an optimal policy in the deterministic navigation problem with target state s^* and let q be the corresponding symmetrized process. Then π is also an optimal policy for V_q^{s*}.*

Proof. First, let us define the sets

$$D_m = \{s \in \mathcal{S} | d(s, s^*) = m\}. \tag{187}$$

It is obvious that under the optimal policy, for the forward process, we have

$$p(S_t \in D_m | S_{t-1} \in D_{m+1}) = 1, m > 0 \tag{188}$$

and

$$p(S_{t-1} \in D_m | S_t \in D_{m-1}, \tau_1^{s^*} > t) = 1, m < d_{\max} \tag{189}$$

This means that for the symmetrized process, with transition probabilities $q = \frac{1}{2}(P + P^{reverse})$, we actually have

$$q(D_{\min(m+1,\max(d(s,s*))}|D_m) = q(D_{\max(m-1,0)}|D_m) = \frac{1}{2}. \tag{190}$$

That is, the symmetrized process with probability 1/2 either increases or decreases the distance to the target state s^* by 1, with exception at the boundaries (i.e. at maximal distance or zero distance), where it stays with probability $\frac{1}{2}$. Now consider the value-function under q, which depends on the sum of the $k-$step probabilities q_k. But the $k-$ step probabilities can be written as

$$q_k(s^*|s) = \sum_{(l_0,l_1,\ldots,l_k)} q(D_0|D_{l_0} q(D_{l_i}|D_{l_{i+1}}) \cdots q(D_{l_k}|s), \tag{191}$$

where the sum runs over all possible alignments such that one can reach s^* after k steps from s. This implies that the value-function can be completely determined by looking at the transition probabilities on the coarse-grained state space of level sets of the distance function, which corresponds to a line graph which self-loops at the ends. To be more precise, the value of s under q is determined simply by

$$V_q(s) = V(D_{d(s,s^*)}). \tag{192}$$

] This tells us that the value of the state is a function only of the distance to the target state. Now we only need to show that this function decreases strictly with distance. To do this, we can simply study the line-graph, or rather the process with transition matrix

$$Q = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & \ldots \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \ldots \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & 0 & 0 & \frac{1}{2} & \frac{1}{2}. \end{pmatrix} \tag{193}$$

Now let $d(s, s*) < d(s', s*)$. It is then clear that $\mathbb{P}[\tau_1^{s^*} = t|S_0 = s] < \mathbb{P}[\tau_1^{s^*} = t|S_0 = s']$, since any path from s' to s^* has to go through s. Hence, recalling proposition 2, also

$$V_q(s) = M^q_{ss*} = \frac{\mathbb{E}[\gamma^{\tau_1^{s^*}}|S_0 = s]}{1 - \mathbb{E}[\gamma^{\tau_1^{s^*}}|S_0 = s^*]} < \frac{\mathbb{E}[\gamma^{\tau_1^{s^*}}|S_0 = s']}{1 - \mathbb{E}[\gamma^{\tau_1^{s^*}}|S_0 = s^*]} = V_q(s'). \tag{194}$$

This shows that the indeed the value-function decreases strictly with distance, and hence the optimal policy π which we started with, also strictly decreases V_q and hence also is an optimal policy for q, which concludes the proof. □

H Supplementary Figures

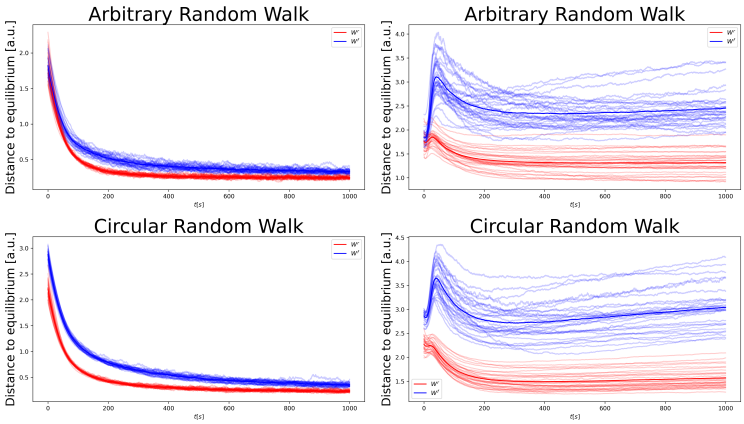

Figure H.1: **Convergence experiments with different activation functions.** We conducted the same experiments as in Figure 3, with the activation functions (tanh,relu) from left to right.

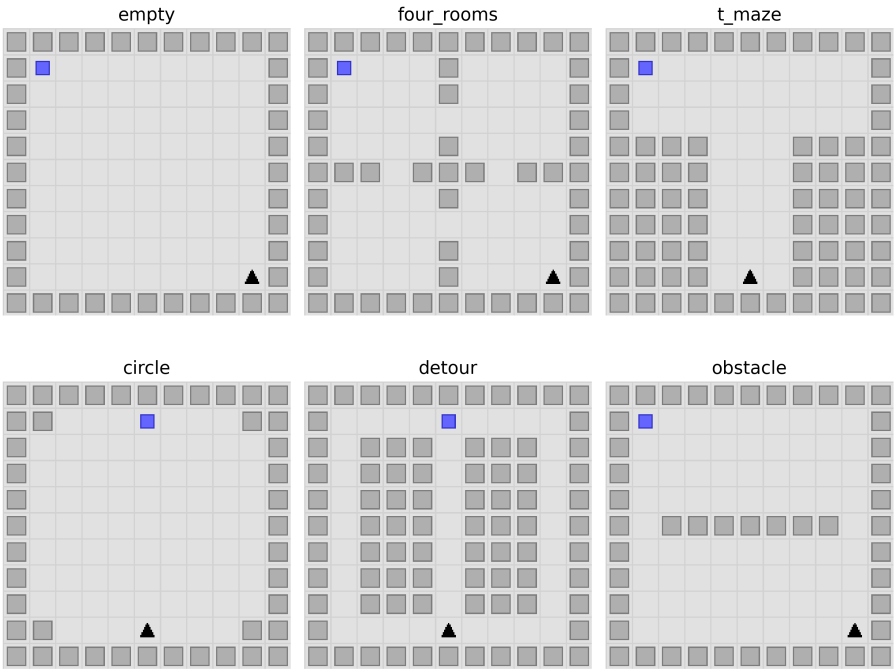

Figure H.2: **Grid environments used in navigation tasks** Outline of the environments used to produce Figure 5. In all environments, agents started at random locations and could choose between four actions, corresponding to moving one step in the respective direction (or staying in place, when the supposed next position would be a wall). Plots were generated using Neuronav package Juliani et al. (2022).

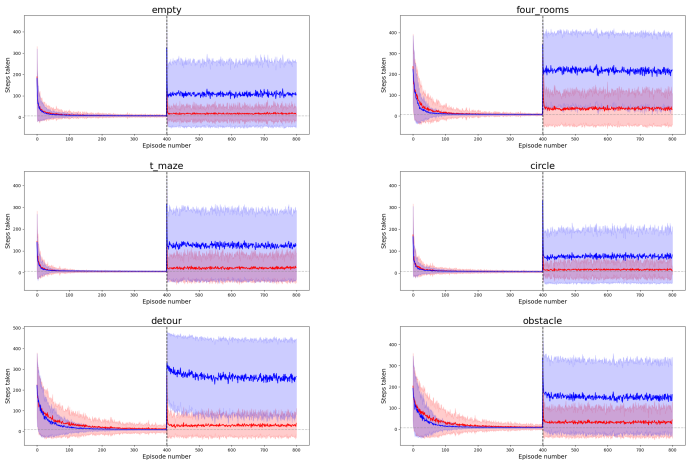

Figure H.3: Generalization performance in the individual environments which are averaged over to generate the left plot in Figure 5.

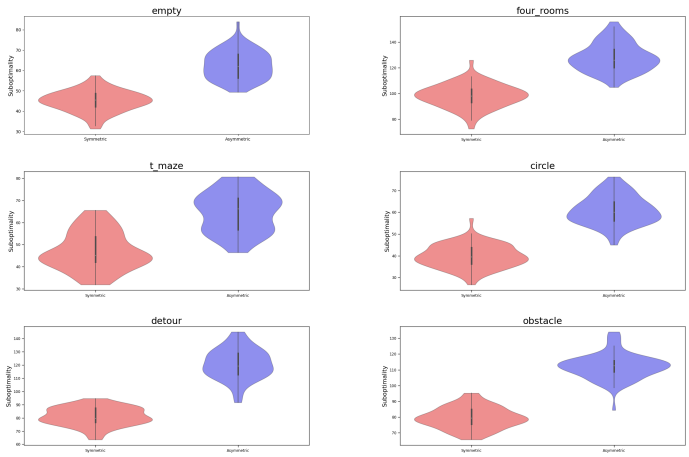

Figure H.4: Generalization performance in the individual environments which are averaged over to generate the right plot in Figure 5.

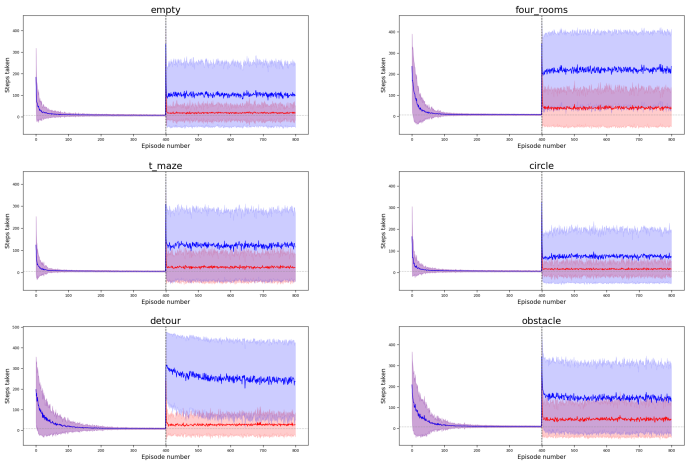

Figure H.5: **Generalization when seeing the same data while training** This figure is identical to Figure H.3, only that here we trained the symmetric agent on the transitions sampled by the asymmetric agent - hence both agents see exactly the same data before generalization.

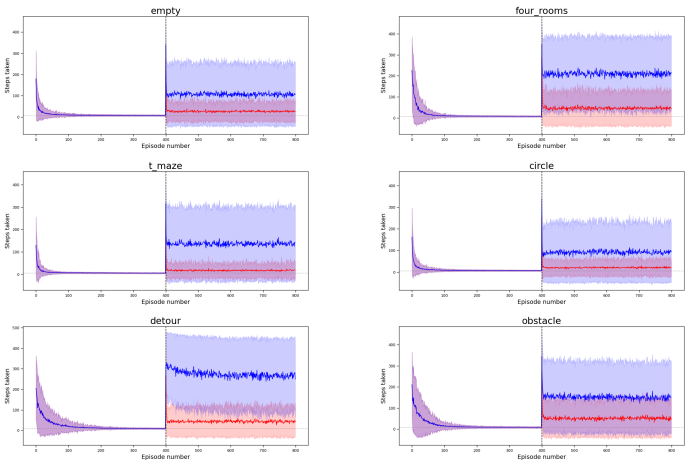

Figure H.6: **Generalization when seeing the same data while training and normalization** This figure is identical to Figure H.5, with the addition that now the updates to the SR are normalized, so that both agents take an equally sized update at each step..

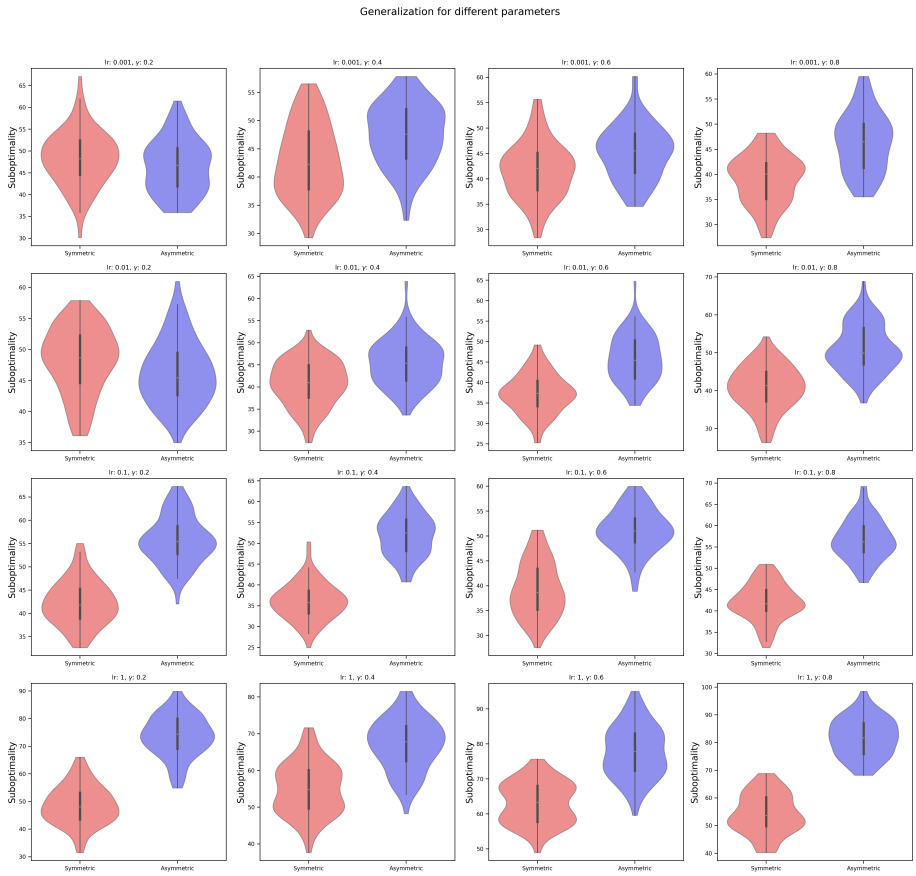

Figure H.7: Generalization performance for varying choices of discount factors and learning rates. All experiments were conducted in the 'empty' environment.Note that in the case were the asymmetric agent outpasses the symmetric agent, generalization is considerably worse than in the regimes where it does not.