Learning the Arrow of Time

Nasim Rahaman[1,2,3] **Steffen Wolf**[1] **Anirudh Goyal**[2] **Roman Remme**[1] **Yoshua Bengio**[2]

[1] Image Analysis and Learning Lab
Ruprecht-Karls-Universität
Heidelberg, Germany

[2] Mila
Montréal, Quebec, Canada

[3] Max-Planck Institute for Intelligent Systems
Tübingen, Germany

Abstract

We humans seem to have an innate understanding of the asymmetric progression of time, which we use to efficiently and safely perceive and manipulate our environment. Drawing inspiration from that, we address the problem of learning an *arrow of time* in a Markov (Decision) Process. We illustrate how a learned arrow of time can capture meaningful information about the environment, which in turn can be used to measure reachability, detect side-effects and to obtain an intrinsic reward signal. We show empirical results on a selection of discrete and continuous environments, and demonstrate for a class of stochastic processes that the learned arrow of time agrees reasonably well with a known notion of an arrow of time given by the celebrated Jordan-Kinderlehrer-Otto result.

1 Introduction

The asymmetric progression of time has a profound effect on how we, as agents, perceive, process and manipulate our environment. Given a sequence of observations of our familiar surroundings (e.g. as photographs or video frames), we possess the innate ability to predict whether the said observations are ordered *correctly*. We use this ability not just to perceive, but also to act: for instance, we know to be cautious about dropping a vase, guided by the intuition that the act of breaking a vase cannot be undone. It is manifest that this profound intuition reflects some fundamental properties of the world in which we dwell, and in this work, we ask whether and how these properties can be exploited to learn a representation that functionally mimics our understanding of the asymmetric nature of time.

In his book *The Nature of Physical World* Eddington (1929), British astronomer Sir Arthur Stanley Eddington coined the term *Arrow of Time* to denote this inherent asymmetry. It was attributed to the non-decreasing nature of the total thermodynamic entropy of an isolated system, as required by the second law of thermodynamics. However, the mathematical groundwork required for its description was already laid by Lyapunov (1892) in the context of dynamical systems. Since then, the notion of an arrow of time has been formalized and explored in various contexts, spanning not only physics (Prigogine, 1978; Jordan et al., 1998; Crooks, 1999) but also algorithmic information theory (Zurek, 1989, 1998), causal inference (Janzing et al., 2016) and time-series analysis (Janzing, 2010) .

Preprint. Under review.

Expectedly, the notion of irreversiblity plays a central role in the discourse. In his Nobel lecture, Prigogine (1978) posits that irreversible processes induce the arrow of time[1]. At the same time, the matter of reversibility has received considerable attention in reinforcement learning, especially in the context of safe exploration (Hans et al.; Moldovan and Abbeel, 2012; Eysenbach et al., 2017), learning backtracking models (Goyal et al., 2018; Nair et al., 2018) and AI-Safety (Amodei et al., 2016; Krakovna et al., 2018). In these applications, *learning* a notion of (ir)reversibility is of paramount importance: for instance, the central premise of safe exploration is to avoid states that prematurely and irreversibly terminate the agent and/or damage the environment. It is related (but not identical) to the problem of detecting and avoiding side-effects, in particular those that adversely affect the environment. In Amodei et al. (2016), the example considered is that of a cleaning robot tasked with moving a box across a room. The optimal way of successfully completing the task might involve the robot doing something disruptive, like knocking a vase over. Such disruptions might be difficult to recover from; in the extreme case, they might be virtually irreversible – say when the vase is broken.

The scope of this work includes detecting and quantifying such disruptions by *learning the arrow of time* of an environment[2]. Concretely, we aim to learn a *potential* (scalar) function on the state space. This function must *keep track of the passage of time*, in the sense that states that tend to occur in the *future* – states with a larger number of broken vases, for instance – should be assigned larger values. To that end, we first introduce a general objective functional (Section 2.1) and study it analytically for toy problems (Section 2.2). We continue by interpreting the solution to the objective (Section 2.3) and highlight its applications (Section 3). To tackle more complex problems, we parameterize the potential function by a neural network and present a stochastic training algorithm (Section 4). Subsequently, we demonstrate results on a selection of discrete and continuous environments and discuss the results critically, highlighting both the strengths and shortcomings of our method (Section 4). Finally, we place our method in a broader context by empirically elucidating connections to the theory of stochastic processes and the variational Fokker-Planck equation (Section 4.1).

2 The h-Potential

2.1 Formalism

Preliminaries. In this section, we will represent the arrow of time as a scalar function h that increases (in expectation) over time. Given a Markov Decision Process (*Environment*), let $\mathcal{S}$ and $\mathcal{A}$ be its state and action space (respectively). A policy π is a function mapping a state $s \in \mathcal{S}$ to a distribution over the action space, $\pi(a|s) \in \mathbb{P}(\mathcal{A})$. Given π and some distribution over the states, we call the sequence $(s_0, s_1, ..., s_N)$ a state-transition trajectory τ_π, where we have $s_{t+1} \sim p_\pi(s_{t+1}|s_t) = \sum_a p(s_{t+1}|s_t, a_t)\pi(a_t|s_t)$ and $s_0 \sim p^0(s)$ for some initial state distribution $p^0(s)$. In this sense, τ_π can be thought of as an instantiation of the Markov (stochastic) process with transitions characterized by p_π.

Methods. Now, for any given function $h : \mathcal{S} \to \mathbb{R}$, one may define the following functional:

$$\mathcal{J}_\pi[h] = \mathbb{E}_t\mathbb{E}_{(s_t \to s_{t+1})\sim\tau^\pi}[h(s_{t+1}) - h(s_t)] = \mathbb{E}_t\mathbb{E}_{s_t}\mathbb{E}_{s_{t+1}}[h(s_{t+1}) - h(s_t)|s_t] \tag{1}$$

where the expectation is over the state transitions of the Markov process τ_π and the time-step t. We now define:

$$h_\pi = \arg\max\{\mathcal{J}_\pi[h] + \lambda\mathcal{T}[h]\} \tag{2}$$

where $\mathcal{T}$ implements some regularizer (e.g. L_2, etc.) weighted by λ. The $\mathcal{J}_\pi$ term is maximized if the quantity $h_\pi(s_t)$ increases in expectation[3] with increasing t, whereas the regularizer $\mathcal{T}$ ensures that h_π is well-behaved and does not diverge to infinity in a finite domain[4]. In what follows, we simplify notation by using h_π and h interchangeably.

[1] Even for systems that are reversible at the microscopic scale, the unified integral fluctuation theorem (Seifert, 2012) shows that the ratio of the probability of a trajectory and its time-reversed counterpart grows exponentially with the amount of entropy the former produces.

[2] Detecting the arrow of time in videos has been studied (Wei et al., 2018; Pickup et al., 2014).

[3] Note that while $\mathcal{J}_\pi[h]$ requires h to increase along all trajectories *in expectation*, it does not guarantee that it must increase along *all* trajectories.

[4] Under certain conditions, h_π resembles the negative stochastic discrete-time Lyapunov function (Li et al., 2013) of the Markov process τ_π.

2.2 Theoretical Analysis

The optimization problem formulated in Eqn 2 can be studied analytically: in Appendix A, we derive the analytic solutions for Markov processes with discrete state-spaces and known transition matrices. The key result of our analysis is a characterization of how the optimal h must behave for the considered regularization schemes. Further, we evaluate the solutions for illustrative toy Markov chains in Fig 1 and 2 to learn the following.

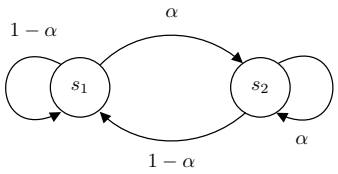


Figure 1: A two variable Markov chain where the reversibility of the transition from the first state to the second is parameterized by α.

First, consider the two variable Markov chain in Fig 1 where the initial state is either s_1 or s_2 with equal probability. If $\alpha > 0.5$, the transition from s_1 to s_2 is more likely than the reverse transition from s_2 to s_1. In this case, one would expect that $h(s_1) < h(s_2)$ and that $h(s_2) - h(s_1)$ increases with α, which is indeed what we find[5] given an appropriate regularizer. Conversely, if $\alpha = 0.5$, the transition between s_1 and s_2 is equally likely in either direction (i.e. it is *fully reversible*), and we obtain $h(s_1) = h(s_2)$. **Second**, consider the four variable Markov chain in Fig 2 where the initial state is s_1 and all state transitions are irreversible. Inituitively, one should expect that $h(s_1) < h(s_2) < h(s_3) < h(s_4)$, and $h(s_2) - h(s_1) = h(s_3) - h(s_2) = h(s_4) - h(s_3)$, given that all state transitions are *equally irreversible*. We obtain this behaviour with an appropriate regularizer[6].

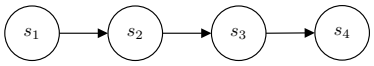


Figure 2: A four variable Markov chain corresponding to a sequence of irreversible state transitions.

While this serves to show that the optimization problem defined in Eqn 2 can indeed lead to interesting solutions, an analytical treatment is not always feasible for complex environments with a large number of states and/or undetermined state transition rules. In such cases, as we shall see in later sections, one may resort to parameterizing h as a function approximator and solve the optimization problem in Eqn 2 with stochastic gradient methods.

2.3 Interpretation and Subtleties

Having defined and analyzed h, we turn to the task of interpreting it. Based on the analytical results presented in Section 2.2, it seems reasonable to expect that even in interesting environments, h should remain constant (in expectation) along reversible trajectories. Further, along trajectories with irreversible transitions, one may hope that h not only increases, but also *quantifies* the irreversibility in some sense. In Section 4, we empirically investigate if this is indeed the case. But before that, there are two conceptual aspects that warrant closer scrutiny.

The first is rooted in the observation that the states s_t are collected by a given but arbitrary policy π. In particular, there may exist *demonic*[7] policies for which the resulting arrow-of-time is unnatural, perhaps even misleading. Consider for instance the actions of a practitioner of Kintsugi, the ancient Japanese art of repairing broken pottery. The corresponding policy might cause the environment to transition from a state where the vase is broken to one where it is not. If we learn h on such *demonic* (or expert) trajectories[8], it might be the case that counter to our intuition, states with a larger number of broken vases are assigned smaller values (and the vice versa). Now, we may choose to resolve this conundrum by defining

$$\mathcal{J}[h] = \mathbb{E}_{\pi\sim\Pi}\mathcal{J}_\pi[h] \tag{3}$$

where Π is the set of all policies defined on $\mathcal{S}$, and $\sim$ denotes uniform sampling. The resulting function $h^* = \arg\max\{\mathcal{J}[h] + \mathcal{T}\}$ would characterize the arrow-of-time w.r.t. all possible policies, and one would expect that for a vast majority of such policies, the transition from broken vase to a intact vase is rather unlikely and/or requires highly specialized policies.

Unfortunately, determining h^* is not feasible for most interesting applications, given the outer expectation over *all* possible policies; we therefore settle for a (uniformly) random policy which we denote by $\pi_\sharp$ (and the corresponding potential as $h_\sharp$). The simplicity (or rather, *clumsiness*) of

[5] cf. Examples 1 and 3 in Appendix A.

[6] cf. Example 4 in Appendix A.

[7] This is indeed an allusion to Maxwell's Demon, cf. Thomson (1874).

[8] By doing so, we solve an inverse RL problem (Ng and Russell, 2000).

$\pi_\sharp$ justifies its adoption, since one would expect a *demonic* policy to be rather complex and not implementable with random actions. In this sense, we ensure that the arrow of time characterizes the underlying dynamics of the environment, and not the peculiarities of a particular agent. However, the price we pay for our choice is the lack of adequate exploration in complex enough environments, although this problem plagues most model-based reinforcement learning approaches[9] (cf. Ha and Schmidhuber (2018)).

The second aspect concerns what we require of environments in which the arrow of time is informative. To illustrate the matter, we consider a class of *Hamiltonian* systems[10], a typical instance of which could be a billiard ball moving on a frictionless arena and bouncing (elastically) off the edges[11]. The state space comprises the ball's velocity and its position constrained to a *billiard table* (without holes!), where the ball is initialized at a random position on the table. For such a system, it can be seen by time-reversal symmetry that when averaged over a large number of trajectories, the state transition $s \to s'$ is just as likely as the reverse transition $s' \to s$. In this case, one should expect the arrow of time to be constant[12] (see Eqn 2). A similar argument can be made for systems that identically follow closed trajectories in their respective state space (e.g. a frictionless and undriven pendulum). It follows that h must remain constant along the trajectory and that the arrow of time is uninformative. However, for so-called *dissipative* systems, the notion of an arrow of time is pronounced and well studied (Prigogine, 1978; Willems, 1972). In MDPs, dissipative behaviour may arise in situations where certain transitions are irreversible by design (e.g. bricks disappearing in Atari Breakout), or due to partial observability (e.g. for a damped pendulum, the state space does not track the microscopic processes that give rise to friction[13]).

Therefore, a central premise underlying the practical utility of learning the arrow of time is that the considered MDP is indeed dissipative. Operating under this assumption, we now discuss a few applications of the arrow of time and experimentally demonstrate its learnability on non-trivial environments.

3 Applications with Related Work

3.1 Measuring Reachability

Given two states s and s' in $\mathcal{S}$, the reachability of s' from s measures how difficult it is for an agent at state s to reach state s'. The prospect of learning reachability from state-transition trajectories has been explored: in Savinov et al. (2018), the approach taken involves learning a logistic regressor network $g^\theta : \mathcal{S} \times \mathcal{S} \to [0, 1]$ to predict the probability of states s' and s being reachable to one another within a certain number of steps (of a random policy), in which case $g(s, s') \approx 1$. However, the model g is not *directed*: it does not learn whether s' is more likely to follow s, or the vice versa. Instead, we propose to learn a function $\eta_\pi : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ such that $\eta_\pi(s \to s') \mapsto h_\pi(s') - h_\pi(s)$ where $\eta_\pi(s \to s')$ is said to measure the *directed reachability* of state s' from state s by following some reference policy π. In the following, we take the reference policy as given (e.g. a random policy) and drop the π for notational clarity. Now, η has the following properties.

First, consider the case where the transition between states s and s' is fully reversible, i.e. when state s is exactly as reachable from state s' as is s' from s. In expectation, we obtain $h(s') = h(s)$ and consequently, $\eta(s \to s') = \eta(s' \to s)$. We denote such reversible transitions with $s \leftrightarrow s'$. Now, if instead the state s' is more likely to occur after state s than state s after s', we say s' is *more reachable*[14] from s than s from s'. It follows in expectation that $h(s') > h(s)$, and consequently, $\eta(s \to s') > 0$ along with $\eta(s' \to s) = -\eta(s \to s') < 0$. **Second**, it can easily be seen that the reachability measure implemented by η is additive by design: given three states $s_0, s_1, s_2 \in \mathcal{S}$, we have that $\eta(s_0 \to s_2) = \eta(s_0 \to s_1) + \eta(s_1 \to s_2)$. As a special case, consider when $s_0 \leftrightarrow s_1$ and

[9] While this is a fundamental problem, powerful methods for off-policy learning exist (cf. Munos et al. (2016) and references therein); however, a full analysis is beyond the scope of the current work.

[10] Systems where Liouville's theorem holds. Further, the Hamiltonian is assumed to be time-independent.

[11] This is well studied in the context of dynamical systems and chaos theory (keyword: *dynamic billiards*); see (Bunimovich, 2007) and references therein.

[12] Prigogine (1978) (page 783 et seq.) provides a more physical treatment.

[13] Note that while a damped pendulum can be expressed as a Hamiltonian system (McDonald, 2015), the Hamiltonian is time dependent.

[14] With respect to the reference (random) policy, which is implicit in our notation.

$s_1 \leftrightarrow s_2$: it follows that $s_0 \leftrightarrow s_2$. In words, if both transitions, from s_0 to s_1 and from s_1 and s_2, are fully reversible, it automatically follows that the transition from s_0 to s_2 is also fully reversible. **Third**, η allows for a *soft* measure of reachability. As we shall see in Section 4, it measures not only *whether* a state s' is reachable from another state s, but also quantifies *how* reachable the former is from the latter. For instance: if the state $s_{(0)}$ is one with all vases intact, $s_{(1)}$ with one vase broken, and $s_{(100)}$ with a hundred vases broken, we find that $\eta(s_{(0)} \to s_{(100)}) \approx 100 \cdot \eta(s_{(0)} \to s_{(1)})$. This behaviour is sought-after in the context of AI-Safety (Krakovna et al., 2018; Leike et al., 2017).

While these properties are satisfactory, the following aspect should be considered to prevent potential confusion. Namely, while we expect $\eta(s' \to s) = \eta(s \to s')$ if the transition between states s and s' is fully reversible, the converse is not guaranteed, especially for non-ergodic environments. For instance, if a Markov chain does not admit a trajectory between states s and s', it might still be the case that $h(s) = h(s')$, and consequently, $\eta(s \to s') = \eta(s' \to s) = 0$.

3.2 Detecting and Penalizing Side Effects for Safe Exploration

The problem of detecting and avoiding side-effects is well known and crucially important for safe exploration (Moldovan and Abbeel, 2012; Eysenbach et al., 2017; Krakovna et al., 2018; Armstrong and Levinstein, 2017). Broadly, the problem involves detecting and avoiding state transitions that permanently and irreversibly damage the agent or the environment. As such, it is not surprising that it is fundamentally related to reachability, as in the agent is prohibited from taking actions that drastically reduce the reachability between the resulting state and some predefined *safe* state. In Eysenbach et al. (2017), the authors learn a reset policy responsible for resetting the environment to some initial state after the agent has completed its trajectory. The resulting value function of the reset policy indicates when the actual (*forward*) policy executes an irreversible state transition. In contrast, Krakovna et al. (2018) propose to attack the problem by measuring reachability relative to a baseline state. However, determining it requires counterfactual reasoning, which in turn requires a known causal model.

We propose to directly use the reachability measure η defined in Section 3.1 to derive a Lagrangian for safe-exploration. Let r_t be the reward (potentially including an exploration bonus) at time-step t. The augmented reward is given by:

$$\hat{r}_t = r_t - \beta \cdot \max\{\eta(s_{t-1} \to s_t), 0\} \tag{4}$$

where β is a scaling coefficient. In practice, one may replace η with $\sigma(\eta)$, where σ is a monotonically increasing transfer function (e.g. a step function).

Intuitively, transitions $s \to s'$ that are *less reversible* cause the h-potential to increase, and the resulting reachability measure $\eta(s \to s') > 0$ in expectation. This in-turn incurs a penalty, which is reflected in the value function of the agent. Conversely, transitions that are reversible should have the property that $\eta(s \to s') = 0$ (also in expectation), thereby incurring no penalty.

3.3 Rewarding Curious Behaviour

In many environments where reinforcement learning methods shine, the reward function is assumed to be given; however, shaping a good reward function can often prove to be a challenging endeavour. It is in this context that the notion of *curiosity* comes to play an important role (Schmidhuber, 2010; Chentanez et al., 2005; Pathak et al., 2017; Burda et al., 2018; Savinov et al., 2018). One typical approach towards encouraging curious behaviour is to seek *novel* states that surprise the agent (Schmidhuber, 2010; Pathak et al., 2017; Burda et al., 2018) and use the error in the agent's prediction of future states is used as a curiosity reward. This approach is, however, susceptible to the so-called noisy-TV problem, wherein an uninteresting source of entropy like a noisy-TV can induce a large curiosity bonus because the agent cannot predict its future state. Savinov et al. (2018) propose to define novelty in terms of (undirected) reachability - states that are easily reachable from the current state are considered less novel.

The h-potential and the corresponding reachability measure η affords another way of defining a curiosity reward: namely, states that are difficult to access by a simple reference policy (e.g. a random policy) should incur a larger reward. In other words, it encourages an agent to *do hard things*, i.e. to seek states that are otherwise difficult to reach just by chance. The general form of the corresponding reward is given by: $\hat{r}_t = -\eta(s_{t-1} \to s_t)$.

Despite the above being independent of the reward function defined by the environment, the latter might often align with the former: in many environments, the task at hand is to reach the least reachable state. This is readily recognized in classical control tasks like Pendulum, Cartpole and Mountain-Car, where the goal state is often the least reachable. However, if the environment's specified task requires the agent to inadvertently execute irreversible trajectories, we expect our proposed reward to be less applicable.

4 Algorithm and Experiments

In this section, we introduce a learning algorithm for the parameterized h-potential and empirically validate it on a selection of discrete and continuous environments (more experiments can be found in Appendix C).

For interesting MDPs with a large number of states and unknown state transition models, an analytic solution like in Section 2.2 is not feasible. In such cases, the h-potential can be parameterized by a neural network h^θ with parameters θ, reducing the optimization problem in Eqn 2 to:

$$\arg\max_\theta \left\{\mathcal{J}_\pi[h^\theta] + \lambda\mathcal{T}[h^\theta]\right\} \tag{5}$$

For stochastic training, the expectation in Eqn 1 can be replaced by its Monte-Carlo estimate, and optimization problem in Eqn 5 can be solved via stochastic gradient descent – the details are given in Algorithm 1 (Appendix B).

Now, we turn to the question of what regularizer to use. Perhaps the simplest candidate is early stopping, wherein the network h^θ is simply not trained to convergence. In combination with weight-decay and/or gradient clipping, we find it to work surprisingly well in practice. Another good regularizer is the so-called *trajectory regularizer* (cf. Eqn 17 in Appendix A):

$$\mathcal{T}[h] = -\mathbb{E}_t\mathbb{E}_{s_t}\mathbb{E}_{s_{t+1}}[(h(s_{t+1}) - h(s_t))^2|s_t] \tag{6}$$

In words, the trajectory regularizer penalizes all changes in h, whereas the primary objective $\mathcal{J}$ encourages h to increase along a trajectory; for an appropriate coefficient λ, a balance is found.

2D World with Vases.[15] The environment considered is a 7×7 2D world, where cells can be occupied by the agent, the goal and/or a vase (their respective positions are randomly sampled in each episode). If the agent enters a cell with a vase in it, the vase disappears without compromising the agent. We use a random policy to generate state-transition trajectories, which we then use to train the h-potential. In Fig 9 (in Appendix C.1.1), we plot the h-potential along a trajectory (parameterized by t) generated by a random policy. We find that $h(s_t)$ increases step-wise when the agent breaks a vase, but remains constant as it moves around – consequently, we observe that the breaking of a vase corresponds to a spike in the $\eta(s_t \to s_{t+1})$ signal (Fig 3). Indeed, the latter is reversible whereas the former irreversibly changes the environment. Moreover, we find that the spikes in Fig 3 are of roughly similar heights, indicating that the model has learned to measure the number of vases broken, i.e. it has learned to *quantify* irreversiblity, instead of merely detecting it[16].

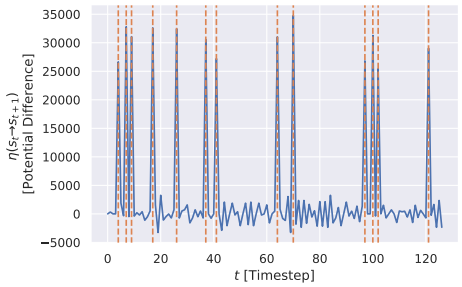


Figure 3: The potential difference (i.e. change in h-potential) between consecutive states along a trajectory. The dashed vertical lines denote when a vase is broken. **Gist:** the h-potential increases step-wise when the agent irreversibly breaks a vase (corresponding to the spikes), but remains constant as it reversibly moves about. Further, the spikes are all of roughly the same height, indicating that the h-potential has learned to count the number of destroyed vases.

Now, to study the robustness of the h-potential to noise, we carry out the following two experiments. In the first of the two, we append a uniformly-random temporally uncorrelated noise signal to the state, which serves as an entropy source (i.e. a noisy-TV). In the second, we append a *clock* to the state,

[15] Experimental details and additional plots can be found in Appendix C.1.1.

[16] The trained h-potential can be utilized to derive a safety reward from the trained model, as elaborated in Section 3.2 (cf. Fig 11 in Appendix C.1.1).

i.e. a temporally-correlated signal that increases in constant intervals as the trajectory progresses. Fig 8a and 8b (Appendix C.1.1) show the respective plots for the corresponding reachability η. While the former noises the background in the $\eta(s_t \rightarrow s_{t+1})$ signal, the spikes remain clearly visible, suggesting that the h-potential is fairly robust to temporally uncorrelated sources of entropy. The latter has a more interesting effect - the h-potential latches on to the clock signal, which results in the baseline η shifting up by a constant. While the spikes remain visible for the most part, this experiment shows that the model might be susceptible to spurious causal signals in the environment.

Sokoban[17] ("warehouse-keeper") is a classic puzzle video game, where an agent must push a number of boxes to set goal locations placed on a map. We use a 2D-world like implementation[18], where each cell can be occupied by a wall, the agent or a box. Additionally, a goal marker may co-occupy a cell with all sprites except a wall. The agent may only push boxes (and not pull), rendering certain moves irreversible - for instance, when a box is pushed against a wall. Solving Sokoban requires long-term planning, precisely due to the existence of such irreversible moves. To further exacerbate the problem, the task of even determining whether a move is irreversible might be non-trivial.

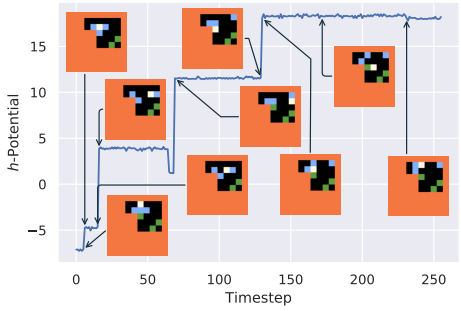


Figure 4: The h-potential along a trajectory from a random policy, annotated with the corresponding state images. The white sprite corresponds to the agent, orange to a wall, blue to a box and green to a goal. **Gist:** the h-potential increases sharply as the agent pushes a box against the wall. While it may decrease (for a given trajectory) if the agent manages to move a box away from the wall (in this case), it increases in expectation over trajectories (cf. Fig 15 in Appendix C.1.3).

We train the h-potential on trajectories generated by a random policy, wherein we generate a random (solvable) map for each trajectory. Fig 4 shows the evolution of h with timesteps for a randomly sampled (validation) map. We find that h increases sharply when a box is pushed against a wall, but remains constant as the agent moves about (potentially pushing a box around). Indeed, the latter is reversible whereas the former is not. Further, we confirm that h does not necessarily increase along all trajectories, but only in expectation (Fig 15). We therefore learn that the h-potential can be used to extract useful information from the environment, all without any external supervision (via rewards) or specialized policies.

Mountain-Car with Friction.[19] The environment considered shares its dynamics with the well known (continuous) Mountain-Car environment (Sutton and Barto, 2011), but with a crucial amendment: the car is subject to friction[20]. Friction is required to make the environment dissipative and thereby induce an arrow of time (cf. Section 2.3). Moreover, we initialize the system in a uniform-randomly sampled state to avoid exploration issues.

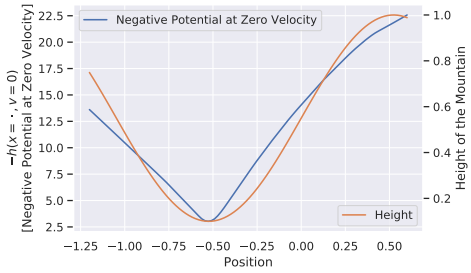


Figure 5: The h-potential (for Mountain Car) at zero-velocity plotted against position. Also plotted (orange) is the height profile of the mountain. **Gist:** the h-potential approximately recovers the height-profile of the mountain with just trajectories from a random policy.

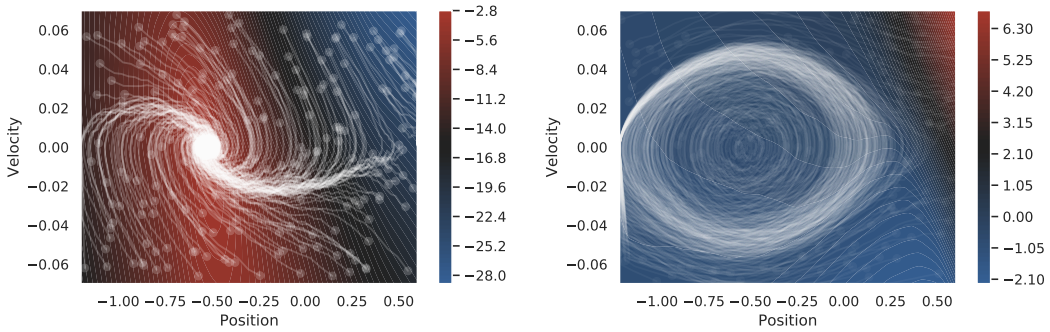
Fig 6a shows the output of the h-potential trained with trajectory regularization overlayed with random trajectories, whereas Fig 5 plots the negative potential at zero-velocity together with the height of the mountain. We not only find that h is at its maximum around the valley, but also that $-h$ at zero velocity largely recovers the terrain just from random trajectories. In addition, we also train the h-potential under identical conditions but without friction. The resulting environment is not dissipative, and in Fig 6b we accordingly find that the corresponding h-potential is not informative (and an order of magnitude smaller), highlighting the practical importance of dissipation.

[17] Experimental details and additional plots can be found in Appendix C.1.3.

[18] Our implementation is adapted from Schrader (2018).

[19] Experimental details and additional plots can be found in Appendix C.2.2

[20] Technically, this is achieved by subtracting a velocity dependent term from the acceleration.



(a) With friction.

(b) Without friction.

Figure 6: The h-potential as a function of state (position and velocity) for (continuous) Mountain-Car with and without friction. The overlay shows random trajectories (emanating from the dots). **Gist:** with friction, we find that the state with largest h is one where the car is stationary at the bottom of the valley. Without friction, there is no dissipation and the car oscillates up and down the valley. Consequently, we observe that the h-potential is constant (up-to edge effects) and thereby uninformative.

Appendices C.1.2 and C.2.1 present additional experiments. The former shows for a discrete environment that the h-potential can be used to derive a reward signal that correlates well with what one might engineer; in the latter, we learn the h-potential for an under-damped pendulum.

4.1 Connection to Stochastic Processes

In this section, we empirically study the link between our method and the theory of stochastic processes. Concretely: our goal is to investigate whether a learned arrow of time behaves *as expected* by comparing it with a known notion of an arrow of time due to Jordan, Kinderlehrer, and Otto (1998). Experimental details are provided in Appendix C.3.

Following the notation of Jordan et al. (1998), we consider the spatial distribution $\rho(\mathbf{x}, t)$ at time t of a particle undergoing Brownian motion in the presence of a potential Ψ. The Ito stochastic differential equation (associated with the Fokker-Planck-Equation) is given by:

$$d\mathbf{X}(t) = -\nabla\Psi(\mathbf{X}(t))dt + \sqrt{2\beta^{-1}}d\mathbf{W}(t) \quad (7)$$

where $\mathbf{X}(t)$ is the random variable corresponding to the distribution $\rho(\mathbf{x}, t)$, the initial spatial distribution $\rho^0(\mathbf{x}) = \rho(\mathbf{x}, t = 0)$ is fixed, β is a parameter and $\mathbf{W}$ is the standard Wiener process (or equivalently, $d\mathbf{W}$ is uncorrelated white-noise). The celebrated Jordan-Kinderlehrer-Otto result (Jordan et al., 1998) shows that the dynamics of a distribution satisfying the Ito SDE (Eqn 7) has the property that the following Free-Energy functional $F[\rho(\cdot, t)]$ can only decrease with time[21]:

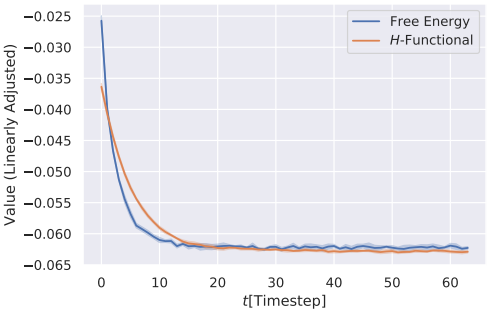


Figure 7: The *true* arrow of time (the Free-Energy functional, in blue) plotted against the learned arrow of time (the H-functional, plotted in orange) after linear scaling and shifting. We find the two to be in good (albeit not perfect) agreement.

$$F[\rho(\cdot,t)] = \underbrace{\int_{\mathbb{R}^n} \Psi\rho(\cdot,t)\,\mathbf{dx}}_{E[\rho(\cdot,t)]} + \beta^{-1}\underbrace{\int_{\mathbb{R}^n} \rho(\cdot,t)\log\rho(\cdot,t)\,\mathbf{dx}}_{-S[\rho(\cdot,t)]} = \mathbb{E}_{\mathbf{x}\sim\rho(\cdot,t)}[\Psi + \beta^{-1}\log\rho(\cdot,t)] \quad (8)$$

where $E[\rho]$ is the energy functional and $S[\rho]$ is the (Gibbs-Boltzmann) entropy functional. It follows that the free-energy functional F induces an arrow of time for the stochastic process generated by Eqn 7.

Given that background, the question we now ask is the following: given just samples from the stochastic process $\mathbf{X}(t)$, how well does a learned h-potential agree with the true Free-Energy

[21] In other words, $F[\rho(\cdot, t)]$ is a Lyapunov functional.

functional? To answer that, we first define the H-functional as:

$$H[\rho(\cdot, t)] = -\mathbb{E}_{\mathbf{x}\sim\rho(\cdot,t)}[h(\mathbf{x})] \tag{9}$$

This allows us to compare $H[\rho(\cdot, t)]$ with $F[\rho(\cdot, t)]$, modulo a linear scaling and shift. To that end, we train h with realizations of two-dimensional random walks under an elliptic paraboloid potential Ψ. Further, $\mathbb{E}_{\mathbf{x}\sim\rho(\cdot,t)}[\Psi]$ is estimated via Monte-Carlo sampling, the differential entropy $\mathbb{E}_{\mathbf{x}\sim\rho(\cdot,t)}[\log \rho(\cdot, t)]$ via a non-parametric estimator (Kozachenko and Leonenko, 1987; Kraskov et al., 2004; Gao et al., 2015), and the linear transform coefficients for H via linear regression. Fig 16 (in Appendix C.3) plots h as a function of state $\mathbf{x} \in \mathbb{R}^2$, whereas Fig 7 shows that after appropriate (linear) scaling, the learned H largely agrees with the true F.

5 Conclusion

Over the course of the paper, we addressed the problem of learning the arrow of time in Markov (Decision) Processes. Having formulated an objective (Eqn 2) and analyzed the corresponding optimization problem for discrete state-spaces (Section 2.2 and Appendix A), we laid out the fundamental challenges that arise – namely the presence of *demonic* policies and the requirement that the environment be *dissipative* (Section 2.3). Under appropriate assumptions, we discussed how the arrow of time can be used to measure reachability, detect side-effects and define a curiosity reward (Section 3). Subsequently, we demonstrated the process of learning the arrow of time on a selection of discrete and continuous environments (Section 4). Finally, we showed for random walks that the learned arrow of time agrees well with the Free-Energy functional, which acts as the *true* arrow of time. Future work could draw connections to algorithmic independence of cause and mechanism (Janzing et al., 2016) and explore applications in causal inference (Janzing, 2010; Peters et al., 2017).

Acknowledgements

The authors would like to acknowledge Min Lin for the initial discussions, Simon Ramstedt, Zaf Ahmed and Maximilian Puelma Touzel for their feedback on the draft.

References

Arthur Stanley Eddington. *The nature of the physical world / by A.S. Eddington*. Cambridge University Press Cambridge, England, 1st ed. edition, 1929.

Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *Kharkov Mathematical Society*, 1892.

Ilya Prigogine. Time, structure, and fluctuations. *Science*, 201(4358):777–785, 1978.

Richard Jordan, David Kinderlehrer, and Felix Otto. The variational formulation of the fokker–planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

Gavin E. Crooks. Entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. *Phys. Rev. E*, 60:2721–2726, Sep 1999. doi: 10.1103/PhysRevE.60.2721. URL `https://link.aps.org/doi/10.1103/PhysRevE.60.2721`.

Wojciech H Zurek. Algorithmic randomness and physical entropy. *Physical Review A*, 40(8):4731, 1989.

Wojciech H Zurek. Decoherence, chaos, quantum-classical correspondence, and the algorithmic arrow of time. *Physica Scripta*, 1998(T76):186, 1998.

Dominik Janzing, Rafael Chaves, and Bernhard Schölkopf. Algorithmic independence of initial condition and dynamical law in thermodynamics and causal inference. *New Journal of Physics*, 18 (9):093052, 2016.

Dominik Janzing. On the entropy production of time series with unidirectional linearity. *Journal of Statistical Physics*, 138(4-5):767–779, 2010.

Udo Seifert. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Reports on progress in physics*, 75(12):126001, 2012.

Alexander Hans, Daniel Schneegaß, Anton Maximilian Schäfer, and Steffen Udluft. Safe exploration for reinforcement learning.

Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.4810*, 2012.

Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine. Leave no trace: Learning to reset for safe and autonomous reinforcement learning. *arXiv preprint arXiv:1711.06782*, 2017.

Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. Recall traces: Backtracking models for efficient reinforcement learning. *arXiv preprint arXiv:1804.00379*, 2018.

Suraj Nair, Mohammad Babaeizadeh, Chelsea Finn, Sergey Levine, and Vikash Kumar. Time reversal as self-supervision. *arXiv preprint arXiv:1810.01128*, 2018.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Victoria Krakovna, Laurent Orseau, Miljan Martic, and Shane Legg. Measuring and avoiding side effects using relative reachability. *CoRR*, abs/1806.01186, 2018. URL `http://arxiv.org/abs/1806.01186`.

Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018.

Lyndsey C Pickup, Zheng Pan, Donglai Wei, YiChang Shih, Changshui Zhang, Andrew Zisserman, Bernhard Scholkopf, and William T Freeman. Seeing the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2035–2042, 2014.

Yan Li, Weihai Zhang, and Xikui Liu. Stability of nonlinear stochastic discrete-time systems. *Journal of Applied Mathematics*, 2013, 2013.

William Thomson. Kinetic theory of the dissipation of energy, 1874.

Andrew Y Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*. Citeseer, 2000.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

L. Bunimovich. Dynamical billiards. *Scholarpedia*, 2(8):1813, 2007. doi: 10.4249/scholarpedia.1813. revision #91212.

Jan C Willems. Dissipative dynamical systems part i: General theory. *Archive for rational mechanics and analysis*, 45(5):321–351, 1972.

Kirk T McDonald. A damped oscillator as a hamiltonian system. 2015.

Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *arXiv preprint arXiv:1810.02274*, 2018.

Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

Stuart Armstrong and Benjamin Levinstein. Low impact artificial intelligences. *arXiv preprint arXiv:1705.10720*, 2017.

Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

Nuttapong Chentanez, Andrew G Barto, and Satinder P Singh. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 1281–1288, 2005.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

Max-Philipp B. Schrader. gym-sokoban. `https://github.com/mpSchrader/gym-sokoban`, 2018.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 2011.

LF Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

Shuyang Gao, Greg Ver Steeg, and Aram Galstyan. Efficient estimation of mutual information for strongly dependent variables. In *Artificial Intelligence and Statistics*, pages 277–286, 2015.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

Zhang Shangtong. Modularized implementation of deep rl algorithms in pytorch. `https://github.com/ShangtongZhang/DeepRL`, 2018.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

A Theoretical Analysis

In this section, we present a theoretical analysis of the optimization problem formulated in Eqn 2, and analytically evaluate the result for a few toy Markov processes to validate that the resulting solutions are indeed consistent with intuition. To simplify the exposition, we consider the discrete case where the state space $\mathcal{S}$ of the MDP is finite.

Consider a discrete Markov chain with enumerable states $s_i \in \mathcal{S}$. At an arbitrary (but given) time-step t, we let $p_i^t = p(s_t = s_i)$ denote the probability that the Markov chain is in state s_i, and $\mathbf{p}^t$ the corresponding vector (over states). With T_{ij} we denote the probability of the Markov chain transitioning from state s_i to s_j under some policy π, i.e. $T_{ij} = p_\pi(s_{t+1} = s_j | s_t = s_i)$. One has the transition rule:

$$\mathbf{p}^{t+1} = \mathbf{p}^t T \qquad \mathbf{p}^t = \mathbf{p}^0 T^t \tag{10}$$

where T^t is the t-th matrix power of T. Now, we let h_i denote the value h_π takes at state s_i, i.e. $h_i = h_\pi(s_i)$, and the corresponding vector (over states) becomes $\mathbf{h}$. This reduces the expectation of the function (now a vector) $\mathbf{h}$ w.r.t any state distribution (now also a vector) $\mathbf{p}$ to the scalar product $\mathbf{p} \cdot \mathbf{h}$. In matrix notation, the optimization problem in Eqn 2 simplifies to:

$$\arg\max_{\mathbf{h}} \frac{1}{N} \sum_{t=0}^{N-1} \left[\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h} \right] + \lambda \mathcal{T}(\mathbf{h}) \tag{11}$$

For certain $\mathcal{T}$, the discrete problem in Eqn 11 can be handled analytically. We consider two candidates for $\mathcal{T}$, the first being the norm of $\mathbf{h}$, and the second one being the norm of change in h_i, in expectation along a trajectory.

Proposition 1. *If $\mathcal{T}(\mathbf{h}) = -(2N)^{-1}\|\mathbf{h}\|^2$, the solution to the optimization problem in Eqn 11 is given by:*

$$\mathbf{h} = \frac{\mathbf{p}^0 T^N - \mathbf{p}^0}{\lambda} \tag{12}$$

Proof. First, note that the objective in Eqn 11 becomes:

$$\mathcal{L}[\mathbf{h}] = \frac{1}{N} \sum_{t=0}^{N-1} \left[\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h} \right] - \frac{1}{2N} \|\mathbf{h}\|^2 \tag{13}$$

To solve the maximization problem, we must differentiate $\mathcal{L}$ w.r.t. its argument $\mathbf{h}$, and set the resulting expression to zero. This yields:

$$\nabla_{\mathbf{h}} \mathcal{L} = \frac{1}{N} \left[\sum_{t=0}^{N-1} (\mathbf{p}^t T - \mathbf{p}^t) - \lambda \mathbf{h} \right] = 0 \tag{14}$$

Now, the summation (over t) is telescoping, and evaluates to $\mathbf{p}^{N-1} T - \mathbf{p}^0$. Substituting $\mathbf{p}^{N-1}$ with the corresponding expression from Eqn 10 and solving for $\mathbf{h}$, we obtain Eqn 12. □

Proposition 1 has an interesting implication: if the Markov chain is initialized at equilibrium, i.e. if $\mathbf{p}^0 = \mathbf{p}^0 T$, we obtain that $\mathbf{h} = \mathbf{0}$ identically. Given the above, we may now consider simple Markov chains to explore the implications of Eqn 12.

Example 1. Consider a Markov chain with two states and reversible transitions, parameterized by $\alpha \in [0, 1]$ such that $T_{11} = T_{21} = 1 - \alpha$ and $T_{12} = T_{22} = \alpha$. If $\mathbf{p}^0 = (1/2, 1/2)$, one obtains:

$$\mathbf{h} \propto (-\gamma, \gamma) \tag{15}$$

where $\gamma = \alpha - 1/2$. To see how, consider that for all $N > 0$, one obtains $\mathbf{p}^0 T^N = (1 - \alpha, \alpha)$. Together with Proposition 1, Eqn 15 follows.

The above example illustrates two things. On the one hand, if $\alpha = 1/2$, one obtains a Markov chain with *perfect reversibility*, i.e. the transition $s_1 \to s_2$ is equally as likely as the transition $s_2 \to s_1$. In this case, one indeed obtains $h(s_1) = h(s_2) = 0$, as mentioned above. On the other hand, if one sets $\alpha = 1$, the transition from $s_2 \to s_1$ is never sampled, and that from $s_1 \to s_2$ is irreversible; consequently, $h(s_2) - h(s_1)$ takes the largest value possible. Now, while this aligns well with our intuition, the following example exposes a weakness of the L2-norm-penalty used in Proposition 1.

Example 2. Consider two Markov chains, both always initialized at s_1. For the first Markov chain, the dynamics admits the following transitions: $s_1 \to s_2 \to s_3 \to s_4$, whereas for the second chain, one has $s_1 \to s_3 \to s_2 \to s_4$. Now, for both chains and $N \geq 4$, it's easy to see that $(\mathbf{p}^0 T^N)_i = 1$ if $i = 4$, but 0 otherwise. From Eqn 12, one obtains:

$$\mathbf{h} \propto (-1, 0, 0, 1) \tag{16}$$

The solution for h given by Eqn 16 indeed increases (non-strictly) monotonously with timestep. However, we obtain $h(s_2) = h(s_3) = 0$ for both Markov chains. In particular, h does not increase between the $s_2 \to s_3$ transition in the former and the $s_3 \to s_2$ transition in the latter, even though both transitions are irreversible. It is in general apparent from 1 that the solution for h depends only on the initial and final state distribution, and not the intermediate trajectory.

Now, consider the following regularizer that penalizes not just the function norm, but the change in h in expectation along trajectories:

$$\mathcal{T}(\mathbf{h}) = -\frac{1}{2N} \sum_{t=0}^{N-1} (\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h})^2 - \frac{\omega}{2N} \|\mathbf{h}\|^2 \tag{17}$$

where ω is the relative weight of the L2 regularizer. This leads to the result:

Proposition 2. *The solution to the optimization problem in Eqn 11 with the regularizer in Eqn 17 is the solution to the following matrix-equation:*

$$\sum_{t=0}^{N-1} \mathbf{p}^0 (T^{t+1} - T^t) \mathbf{h}\, \mathbf{p}^0 (T^{t+1} - T^t) + \omega \mathbf{h} = \frac{\mathbf{p}^0 T^N - \mathbf{p}^0}{2\lambda} \tag{18}$$

Proof. Analogous to Eqn 13, we may write the objective in Eqn 11 as (by substituting Eqn 17 in Eqn 11):

$$\mathcal{L}[\mathbf{h}] = \frac{1}{N} \sum_{t=0}^{N-1} \left[\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h}\right] - \frac{\lambda}{2N} \sum_{t=0}^{N-1} (\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h})^2 - \frac{\lambda\omega}{2N} \|\mathbf{h}\|^2 \tag{19}$$

Like in Proposition 1, we maximize it by setting the gradient of $\mathcal{L}$ w.r.t. $\mathbf{h}$ to zero. This yields:

$$\nabla_{\mathbf{h}} \mathcal{L} = \frac{1}{N} \left[\sum_{t=0}^{N-1} (\mathbf{p}^t T - \mathbf{p}^t) - \frac{\lambda}{2} \nabla_{\mathbf{h}} \sum_{t=0}^{N-1} (\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h})^2 - \omega\lambda \mathbf{h} \right] = 0 \tag{20}$$

The first term in the RHS is again a telescoping sum; it evaluates to: $\mathbf{p}^0 T^N - \mathbf{p}^0$ (cf. proof of Proposition 1). The second term can be expressed as (with I as the identity matrix):

$$\frac{\lambda}{2} \nabla_{\mathbf{h}} \sum_{t=0}^{N-1} (\mathbf{p}^t T \mathbf{h} - \mathbf{p}^t \cdot \mathbf{h})^2 = \frac{\lambda}{2} \sum_{t=0}^{N-1} \nabla_{\mathbf{h}} (\mathbf{p}^t (T - I) \mathbf{h})^2 \tag{21}$$

$$= \lambda \sum_{t=0}^{N-1} (\mathbf{p}^t (T - I) \mathbf{h})(\mathbf{p}^t (T - I)) \tag{22}$$

$$= \lambda \sum_{t=0}^{N-1} \mathbf{p}^0 (T^{t+1} - T^t) \mathbf{h}\, \mathbf{p}^0 (T^{t+1} - T^t) \tag{23}$$

where the last equality follows from Eqn 10. Substituting the above in Eqn 20 and rearranging terms yields Eqn 18. □

While Eqn 18 does not yield an explicit expression for $\mathbf{h}$, it is sufficient for analysing individual cases considered in Examples 1 and 2.

Example 3. Consider the two-state Markov chain in Example 1 and the associated transition matrix T and initial state distribution $\mathbf{p}^0 = (1/2, 1/2)$. Using the regularization scheme in Eqn 17 and the associated solution Eqn 18, one obtains:

$$\mathbf{h} = (-\tilde{\gamma}, \tilde{\gamma}) \tag{24}$$

where:

$$\tilde{\gamma} = \frac{2\alpha - 1}{\lambda(4\alpha^2 - 4\alpha + 2\omega + 1)} \tag{25}$$

To obtain this result[22], we use that $T^t = T$ for all $t \geq 1$ and truncate the sum without loss of generality at $N = 1$.

Like in Example 1, we observe $h(s_1) = h(s_2) = 0$ if $\alpha = 1/2$ for all $\omega > 0$ (i.e. at equilibrium). In addition, if $\omega \geq 1/2$, it can be shown that $h(s_2) - h(s_1)$ increases monotonously with α and takes the largest possible value at $\alpha = 1$. We therefore find that for the simple two-state Markov chain of Example 1, the regularization in Eqn 17 indeed leads to intuitive behaviour for the respective solution $\mathbf{h}$. Now:

Example 4. Consider the four-state Markov chain with transitions $s_1 \to s_2 \to s_3 \to s_4$ and the corresponding transition matrix T, where $T_{12} = T_{23} = T_{34} = T_{44} = 1$, $T_{ij} = 0$ for all other i, j. Set $\mathbf{p}^0 = (1, 0, 0, 0)$, i.e. the chain is always initialized at s_1. Now, the summation over t in Eqn 18 can be truncated w.l.o.g when $N = 4$, given that $T^{t+1} = T^t$ for all $t \geq 3$. At $\omega = 0$, one solution is:

$$\mathbf{h} \propto (-3/2, -1/2, 1/2, 3/2) \tag{26}$$

Further, for all finite ω, one obtains $h(s_1) < h(s_2) < h(s_3) < h(s_4)$, where the inequality is strict. This is unlike Eqn 16 where $h(s_2) = h(s_3)$, and consistent with the intuitive expectation that the arrow of time must increase along irreversible transitions.

In conclusion: we find that the functional objective defined in Eqn 2 may indeed lead to analytical solutions that are consistent with the notion of an arrow of time in certain toy Markov chains. However, in most interesting real world environments, the transition model T is not known and or or the number of states is infeasibly large, rendering an analytic solution intractable. In such cases, as we see in Section 4, it is possible to parameterize h as a neural network and train the resulting model with stochastic gradient descent to optimize the functional objective defined in Eqn 2.

B Algorithm

Algorithm 1 Training the h-Potential

Require: Environment `Env`, random policy $\pi_\sharp$, trajectory buffer `B`

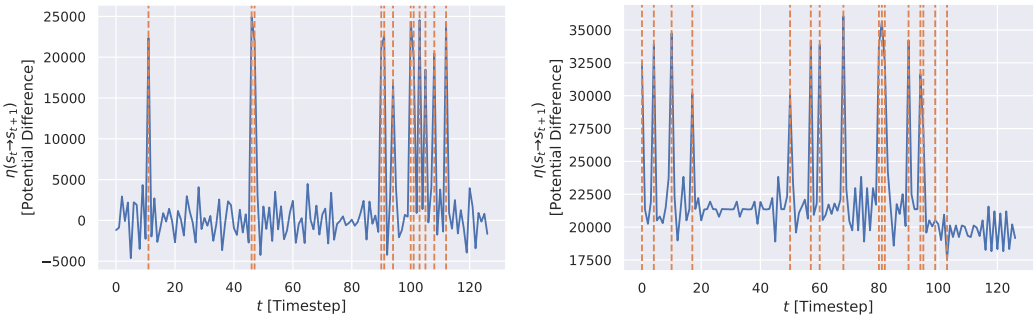
Require: Model h^θ, regularizer $\mathcal{T}$, optimizer.

for $k = 1...M$ **do**
  `B[k, :]` $\leftarrow (s_0, ..., s_N) \sim$ `Env`$[\pi_\sharp]$ {Sample a trajectory of length N with the random policy and write to k-th position in the buffer.}
end for
loop
  Sample trajectory index $k \sim \{1, ..., M\}$ and time-step $t \sim \{0, ..., N-1\}$. {In general, one may sample multiple k's and t's for a larger mini-batch.}
  Fetch states $s_t \leftarrow$ `B[k, t]` and $s_{t+1} \leftarrow$ `B[k, t + 1]` from buffer.
  Compute loss as $L(\theta) = -[h^\theta(s_{t+1}) - h^\theta(s_t)]$.
  if using trajectory regularizer **then**
    Compute regularizer term as $[h^\theta(s_{t+1}) - h^\theta(s_t)]^2$ and add to $L(\theta)$.
  else
    Apply the regularizer as required. If early-stopping, break out of the loop if necessary.
  end if
  Compute parameter gradients $\nabla_\theta L(\theta)$ and update parameters with the optimizer.
end loop

C Experimental Details

All experiments were run on a workstation with 40 cores, 256 GB RAM and 2 nVidia GTX 1080Ti.

[22] Interested readers may refer to the attached SymPy computation.



(a) TV-Noise

(b) Causal Noise

Figure 8: The potential difference η plotted along trajectories, where the state-space is augmented with temporally uncorrelated (TV-) and correlated (causal) noise. The dashed vertical lines indicate time-steps where a vase is broken. **Gist:** while our method is fairly robust to TV-noise, it might get distracted by causal noise.

C.1 Discrete Environments

C.1.1 2D World with Vases

The environment state comprises three 7×7 binary images (corresponding to agent, vases and goal), and the vases appear in a different arrangement every time the environment is reset. The probability of sampling a vase at any given position is set to $1/2$.

We use a two-layer deep and 256-unit wide ReLU network to parameterize the h-potential. It is trained on 4096 trajectories of length 128 for 10000 iterations of stochastic gradient descent with Adam optimizer (learning rate: 0.0001). The batch-size is set to 128, and we use a weight decay of 0.005 to regularize the model. We use a validation trajectory to generate the plots in Fig 9 and 3. Moreover, Fig 10 shows histograms of the values taken by h at various time-steps along the trajectory. We learn that h takes on larger values (on average) as t increases.

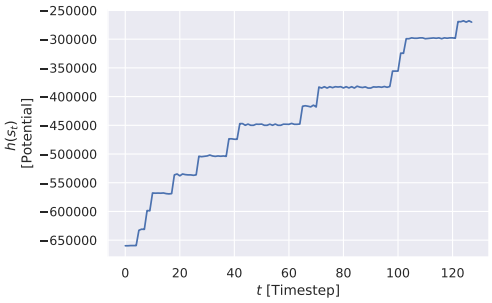
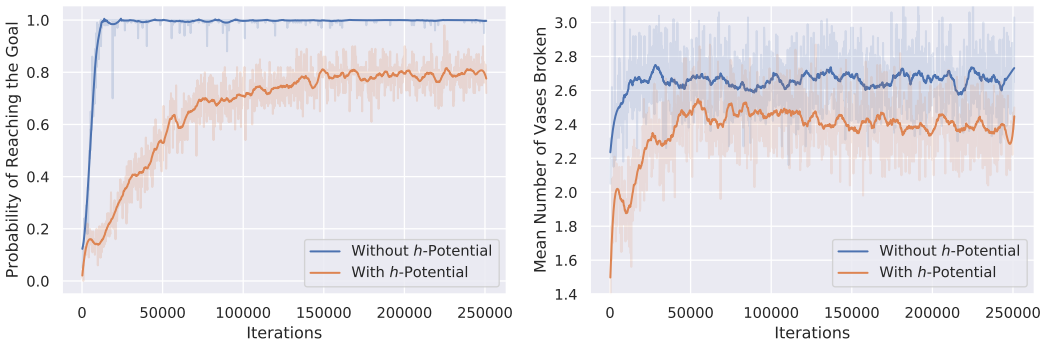


Figure 9: The h-potential along a trajectory sampled from a random policy. **Gist:** The h-potential increases step-wise along the trajectory every time an agent (irreversibly) breaks a vase. It remains constant as the agent (reversibly) moves about.

To test the robustness of our method, we conduct experiments where the environment state is augmented with one of: (a) a 7×7 image with uniform-randomly sampled pixel values (*TV-noise*) and (b) a 7×7 image where every pixel takes the value $t/128$, where t is the time-step[23] of the corresponding state (*Causal Noise*). Fig 8a and 8b plot the corresponding $\eta = \Delta h$ along randomly sampled trajectories.

Given a learned arrow of time, we now present an experiment where we use it to derive a safe-exploration penalty (in addition to the environment reward). To that end, we now consider the situation where the agent's policy is not random, but specialized to reach the goal state (from its current state). For both the baseline and the safe agents, every action is rewarded with the change in Manhattan norm of the agent's position to that of the goal – i.e. an action that moves the agent closer to the goal is rewarded $+1$, one that moves it farther away from the goal is penalized -1, and one that keeps the distance unchanged is neither penalized nor rewarded (0). Further, every step is penalized by -0.1 (so as to keep the trajectories short), and exceeding the available time limit (30 steps) incurs a termination penalty (-10). In addition, the reward function of the safe agent is augmented with the reachability, i.e. it takes the form described in Eqn 4. We use $\beta = 4$ and a transfer function σ such that $\sigma(\eta) = 0$ if $\eta < 5000$ (cf. Fig 3), and 1 otherwise.

[23] Recall that the trajectory length is set to 128.



(a) Probability of reaching the goal.

(b) Number of vases broken.

Figure 11: Probability of reaching the goal and the expected number of vases broken, obtained over 100 evaluation episodes (per step). **Gist:** while the safety Lagrangian results in fewer vases broken, the probability of reaching the goal state is compromised. This trade-off between safety and efficiency is expected (cf. Moldovan and Abbeel (2012)).

The policy is parameterized by a 3-layer deep 256-unit wide (fully connected) ReLU network and trained via Duelling Double Deep Q-Learning[24] (Van Hasselt et al., 2016; Wang et al., 2015). The discount factor is set to 0.99 and the target network is updated once every 200 iterations. For exploration, we use a $1-\epsilon$ greedy policy, where ϵ is decayed linearly from 1 to 0.1 in the span of the first 10000 iterations. The replay buffer stores 10000 experiences and the batch-size used is 10. Fig 11a shows the probability of reaching the goal (in an episode of 30 steps) over the iterations (sample size 100), whereas Fig 11b shows the expected number of vases broken per episode (over the same 100 episodes). Both curves are smoothed by a Savitzky-Golay filter (Savitzky and Golay, 1964) of order 3 and window-size 53 (the original, unsmoothed curves are shaded). As expected, we find that using the safety penalty does indeed result in fewer vases broken, but also makes the task of reaching the goal difficult (we do not ensure that the goal is reachable without breaking vases).

C.1.2 2D World with Drying Tomatoes

The environment considered comprises a 7×7 2D world where each cell is initially occupied by watered tomato plant[25]. The agent waters the cell it occupies, restoring the moisture level of the plant in the said cell to 100%. However, for each step the agent does not water a plant, it loses some moisture (by 2% of maximum in our experiments). If a plant loses all moisture, it is considered dead and no amount of watering can resurrect it. The state-space comprises two 7×7 images: the first image is an indicator of the agent's position, whereas the pixel values of the second image quantifies the amount of moisture held by the plant[26] at the corresponding location.

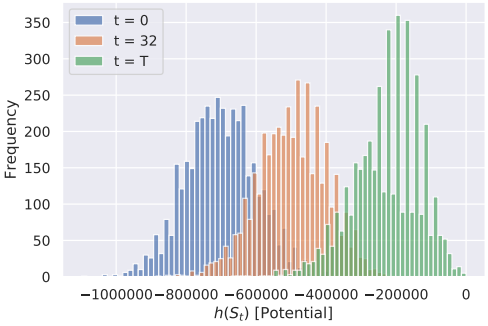


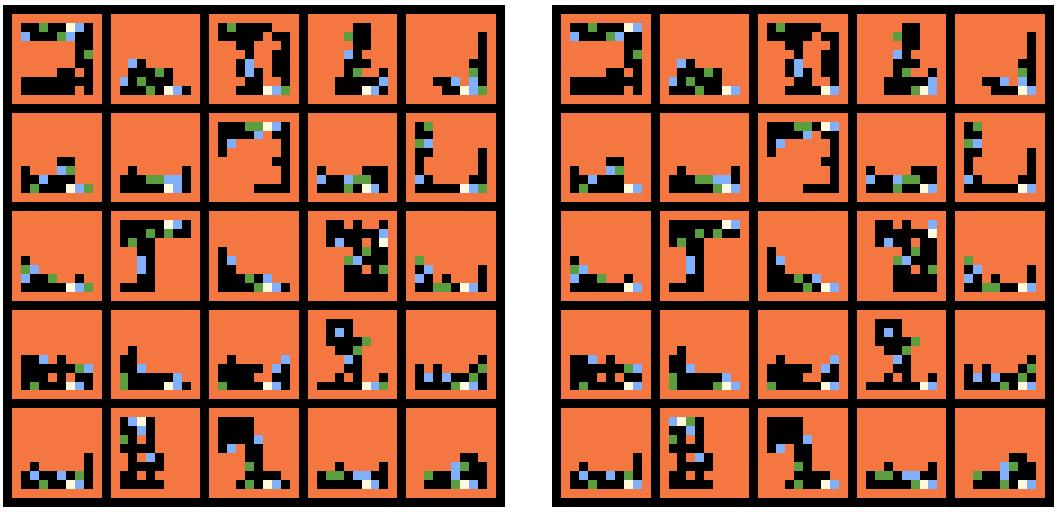
Figure 10: Histogram (over trajectories) of values taken by h at time-steps $t=0$, $t=32$ and $t=T=128$.

We show that it is possible to recover an intrinsic reward signal that coincides well with one that one might engineer. To that end, we parameterize the h-potential as a two-layer deep 256-unit wide ReLU network and train it on 4096 trajectories (generated by a random policy) of length 128 for 10000 iterations of Adam (learning rate: 0.0001). The batch-size is set to 128 and the model is regularized with the trajectory regularizer ($\lambda = 0.5$).

[24] We adapt the implementation due to Shangtong (2018).

[25] We draw inspiration from the tomato-watering environment described in Leike et al. (2017).

[26] This is a strong causal signal which may distract the model. We include it nonetheless to make the task more challenging.



(a) States at time t.

(b) States at time $t+1$.

Figure 13: Random samples from 200 transitions that cause the largest increase in the h-potential (out of a sample size of 8000 transitions). The orange, white, blue and green sprites correspond to a wall, the agent, a box and a goal marker respectively. **Gist:** pushing boxes against the wall increases the h-potential.

Unsurprisingly, we find that h increases as the plants lose moisture. But conversely, when the agent waters a plant, it causes the h-potential to decrease by an amount that strongly correlates with the amount of moisture the watered plant gains. This can be used to define a *dense* reward signal for the agent:

$$\hat{r}_t = -\{\eta(s_{t-1} \to s_t) - \text{RunningAverage}_t[\eta]\} \tag{27}$$

where we use a momentum of 0.95 to evaluate the running average.

In Fig 12, we plot for a random trajectory the intrinsic reward $\hat{r}_t$ against a reference reward, which in this case is the moisture gain of the plant the agent just watered. Further, we observe the reward function dropping significantly at around the 90-th iteration - this is precisely when all plants have died. This demonstrates that the h-potential can indeed be useful for defining intrinsic rewards.

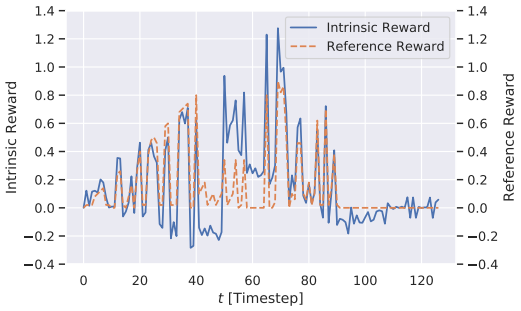


Figure 12: The intrinsic reward (Eqn 27) plotted against an engineered reward, which in this case is the amount of moisture gained by the tomato plant the agent just watered. **Gist:** the h-Potential captures useful information about the environment, which can then be utilized to define intrinsic rewards.

C.1.3 Sokoban

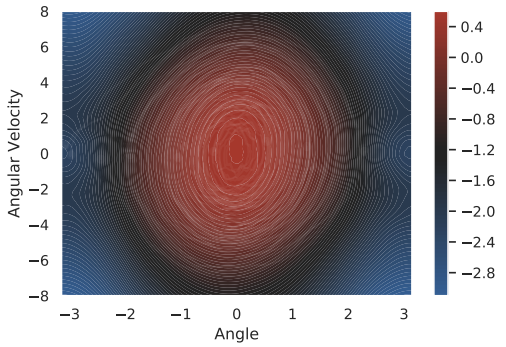
The environment state comprises five 10×10 binary images, where the pixel value at each location indicates the presence of the agent, a box, a goal, a wall and empty space. The layout of all sprites are randomized at each environment reset, under the constraint that the game is still solvable (Schrader, 2018). The h-potential is parameterized by a two-layer deep and 512-unit wide network, which is trained on 4096 trajectories of length 512 for 20000 steps of Adam (learning rate: 0.0001). The batch-size is set to 256 and we use the trajectory regularizer ($\lambda = 0.05$) to regularize our model.

C.2 Continous Environments

C.2.1 Under-damped Pendulum

Under-damped Pendulum. The environment considered simulates an under-damped pendulum, where the state space comprises the angle[27] θ and angular velocity $\dot{\theta}$ of the pendulum. The dynamics

[27] θ is commonly represented as $(\cos(\theta), \sin(\theta))$ instead of a scalar.



(a) Learned h-Potential as a function of the state-space $(\theta, \dot{\theta})$. Overlaid are trajectories from a random policy.

(b) Negative of the learned h-Potential as a function of θ when $\dot{\theta} = 0$.

Figure 14: **Gist:** the learned h-Potential takes large values around $(\theta, \dot{\theta}) = \mathbf{0}$, since that is where most trajectories terminate due to the effect of damping.

are governed by the following differential equation where τ is the (time-dependent) torque applied by the agent and m, l, g are constants:

$$\ddot{\theta} = \frac{-3g}{2l}\sin(\theta) + \frac{3\tau}{ml^2} - \alpha\dot{\theta} \tag{28}$$

We adapt the implementation in OpenAI Gym (Brockman et al., 2016) to add an extra term $\alpha\dot{\theta}$ to the dynamics to simulate friction. In our experiments, we set $g = 10$, $m = l = 1$, $\alpha = 0.1$ and the torque τ is uniformly sampled iid. from the interval $[-2, 2]$.

The h-Potential is parameterized by a two-layer 256-unit wide ReLU network, which is trained on 4096 trajectories of length 256 for 20000 steps of stochastic gradient descent with Adam (learning rate: 0.0001). The batch-size is set to 1024 and we use the trajectory regularizer with $\lambda = 1$ to regularize the network. Fig 14a plots the learned h-potential (trained with trajectory regularizer) as a function of the state $(\theta, \dot{\theta})$ whereas Fig 14b shows the negative potential for all angles θ at zero angular velocity, i.e. $\dot{\theta} = 0$. We indeed find that states in the vicinity of $\theta = 0$ have a larger h-potential, owing to the fact that all trajectories converge to $(\theta, \dot{\theta}) = \mathbf{0}$ for large t due to the dissipative action of friction.

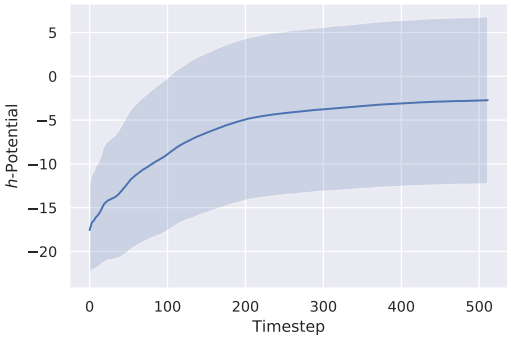


Figure 15: h-Potential averaged over 8000 trajectories, plotted against timestep t; shaded band shows the standard deviation. **Gist:** as required by its objective (Eqn 1), the h-Potential must increase in expectation along trajectories.

C.2.2 Continuous Mountain Car

The environment[28] considered is a variation of Mountain Car (Sutton and Barto, 2011), where the state-space is a tuple $(x, \dot{x})$ of the position and velocity of a vehicle on a mountainous terrain. The action space is the interval $[-1, 1]$ and denotes the *force* f applied by the vehicle. The dynamics of the modified environment is given by the following equation of motion:

$$\ddot{x} = \zeta f - 0.0025\cos 3x - \alpha\dot{x} \tag{29}$$

where ζ and α are constants set to 0.0015 and 0.1 respectively, and the velocity $\dot{x}$ is clamped to the interval $[-0.07, 0.07]$. Our modification is the last $\alpha\dot{x}$ term to simulate friction. Further, the

[28] We adapt the implemetation due to Brockman et al. (2016), available here: `github.com/openai/gym/blob/master/gym/envs/classic_control/continuous_mountain_car.py`

initial state $(x, \dot{x})$ is sampled uniformly from the state space $\mathcal{S} = [-1.2, 0.6] \times [-0.07, 0.07]$. This can potentially be avoided if an exploratory policy is used (instead of the random policy) to gather trajectories, but we leave this for future work.

The h-potential is parameterized by a two-layer 256-unit wide ReLU network, which is trained on 4096 trajectories of length 256 for 20000 steps of stochastic gradient descent with Adam (learning rate: 0.0001). The batch-size is set to 1024 and we use the trajectory regularizer with $\lambda = 1$.

C.3 Stochastic Processes

The environment state comprises two scalars, the x_1 and x_2 coordinates of the particle's position $\mathbf{x}$. The potential is given by:

$$\Psi(\mathbf{x}) = \frac{x_1^2}{20} + \frac{x_2^2}{40} \tag{30}$$

corresponding to a two dimensional Ornstein-Uhlenbeck process, and $\sqrt{2\beta^{-1}}$ is set to 0.3.

We train a two-layer deep, 512-unit wide network on 8092 trajectories of length 64 for 20000 steps of stochastic gradient descent with Adam (learning rate: 0.0001). The batch-size is set to 1024 and the network is regularized by weight decay (with coefficient 0.0005). Fig 16 shows the learned h-potential as a function of position $\mathbf{x}$. Fig 7 compares the free-energy functional with the learnt arrow of time given by the linearly scaled H-functional. To obtain the linear scaling parameters for the H, we find parameters w and b such that $\sum_{t=0}^{N}(wH[\rho(\cdot, t)] + b - F[\rho(\cdot, t)])^2$ is minimized (constraining w to be positive).

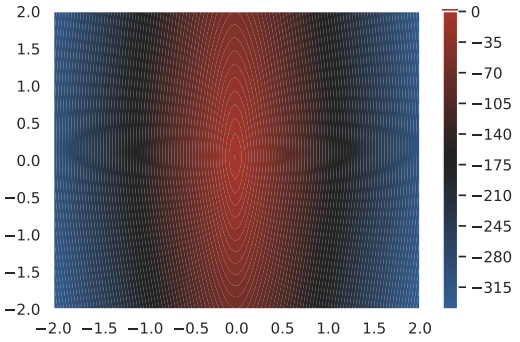


Figure 16: Learned h-Potential as a function of position $\mathbf{x}$. Observe the qualitative similarity to the potential Ψ defined in Eqn 30.