A Harmonic Mean Formulation of Average Reward Reinforcement Learning in SMDPs

Erel Shtossel
Bar Ilan University
Ramat Gan, Israel

Alicia Vidler
Bar Ilan University
Ramat Gan, Israel

Uri Shaham
Bar Ilan University
Ramat Gan, Israel

Gal A. Kaminka
Bar Ilan University
Ramat Gan, Israel

ABSTRACT

Recent research has revived and amplified interest in algorithms for undiscounted average reward reinforcement learning in infinite-horizon, non-episodic (continuing) tasks. Semi-Markov decision processes (SMDPs) are of particular interest. In SMDPs, discrete actions stochastically generate both *rewards* and *durations*, and the objective is to optimize the average *reward rate*. Existing algorithms approach this by optimizing the ratio of rewards to durations. However, when rewards and durations are non-stationary (in the infinite horizon), this can be incorrect. This paper presents a novel modified harmonic mean operator that correctly computes reward rates even under such conditions. This yields model-free learning algorithms that can work with SMDPs, while maintaining robustness to non-stationary reward and duration distributions over time. We prove theoretical properties of the modified harmonic mean operator, and empirically demonstrate its efficacy in comparison to existing algorithms.

KEYWORDS

Reinforcement Learning, Semi-Markov Decision Processes, R learning, Bitcoin, harmonic RL, financial markets

1 INTRODUCTION

Recent research has revived and amplified interest in undiscounted average reward reinforcement learning [7, 13, 16]. Motivated by real-world applications, average reward reinforcement learning (ARL) addresses infinite-horizon, non-episodic (continuing) tasks. While less popular than the familiar discounted-rewards formulation for reinforcement learning, it is nonetheless a well known optimization criteria [15].

While ARL has been studied from multiple perspectives in Markov Decision Processes (MDPs), there are almost no investigations of ARL in semi-Markov settings (SMDP), where discrete actions have *durations*. In every decision step, an action is selected. It then takes a (stochastically) varying duration before the next state is known, and the rewards received. The objective of ARL in SMDPs is to optimize the policy's average *reward rate*.

To the best of our knowledge, only two algorithms were proposed for ARL in SMDP settings. The first algorithm, SMART [5], uses the *long-term sample average* to track the optimal reward rate. This makes it brittle when the rate is non-stationary, i.e., rewards and/or durations are not stationary. Relaxed-SMART [9] handles the non-stationary by approximating the reward rate as the ratio of the *Exponential Moving Average* (EMA) of the rewards, to the EMA of the durations. This approximation is perfect when the rewards and durations are independent of each other, but faces limitations otherwise (as we show in Section 3).

We note that rather than approximating the average rate as the ratio of arithmetic average of the rewards to the arithmetic average of the durations, the average rate can be directly calculated using the harmonic mean, i.e., as the harmonic average of the reward rates received in different steps. For example, the harmonic mean is the correct averaging operator to use in averaging velocities (distance/time), or throughput (units/time). Unfortunately, the harmonic mean is not well-defined for zero or mixed-sign rewards, and therefore ill-suited to general reinforcement learning (see Section 3).

To address this, we first present a novel modified harmonic mean operator that correctly computes reward rates (including when rewards are negative or zero), and admits stochastic approximation: an exponential moving harmonic mean, rather than the exponential moving arithmetic mean so commonly used in reinforcement-learning literature. We prove theoretical properties of modified harmonic mean operator, showing that it generalizes the familiar harmonic mean, and that it obeys axiomatic definitions of means. We also show that it is not a member of the class of quasi-linear means, which includes arithmetic, geometric, and harmonic means.

Use of the modified harmonic mean operator yields a novel model-free reinforcement learning algorithm for SMDPs—Harmonic R-Learning—that directly generalizes existing algorithms, and does not suffer from the same weaknesses as SMART or Relaxed-SMART. We discuss the conditions under which modified harmonic mean operator is better than the alternative (approximation by the ratio). We empirically demonstrate the improvements offered by Harmonic R-Learning over existing algorithms in two domains with non-stationary, volatile reward rates: a simple two-state SMDP, and Bitcoin trading (using real data).

2 BACKGROUND: AVERAGE REWARD RL

We begin with a brief reminder of the problem of optimizing the average undiscounted rewards in Markov Decision Processes (MDPs), and how it has been addressed in reinforcement learning literature (in Section 2.1). We then discuss such learning in Semi-Markov Decision Processes (SMDPs), where transition durations (*sojourn* times) must be taken into account (Section 2.2).

Proc. of the Adaptive and Learning Agents Workshop (ALA 2026), Aydeniz, Delgrange, Mohammedalamen, Yang (eds.), May 25 – 26, 2026, Paphos, Cyprus, https://alaworkshop2026.github.io/. 2026.

2.1 Maximizing Average Undiscounted Reward

Markov Decision Processes (MDPs) are a popular formulation of sequential decision making problems. An infinite-horizon MDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$. $\mathcal{S}$ is a set of states (at any time t, the agent is in a state $s_t \in \mathcal{S}$.). $\mathcal{A}$ is a set of actions that the agent can take to transition between states. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is a transition probability function, determining the probability that an action $a \in \mathcal{A}$, taken in a state $s \in \mathcal{S}$, will result in a transition to a new state $s' \in \mathcal{S}$.

The reward function kernel $\mathcal{R} : \mathbb{N} \times \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathcal{P}(\mathbb{R})$, maps a transition taken on step $t \in \mathbb{N}$ to a probability distribution over rewards in $\mathbb{R}$. We write $\mathcal{R}_t(s, a, s')$ to define the distribution from which the system samples the reward received by the agent on taking the action a in state s, and transitioning to the state s' at time t. We use R_t to denote the random variable drawn from this distribution. The use of the index t in the definition allows discussing non-stationary rewards, where the distribution of rewards may change with t.

A solution to an MDP is a deterministic policy $\pi : \mathcal{S} \mapsto \mathcal{A}$, which assigns an action to each state, i.e., $\forall s_t \in \mathcal{S}, \exists a_t \in \mathcal{A}$ such that $\pi(s_t) = a_t$.

The value of R_t is determined considering both $\mathcal{T}$ (which determines the probability of all possible next states s_{t+1}, given the action) and the reward distribution $\mathcal{R}_t(s, a, s')$ (which determines the reward for reaching any specific s_{t+1} from s_t, given the action determined by the policy π). On taking an action a_t in state s_t and transitioning to s_{t+1}, the agent receives a realized reward

$$r_t \sim \mathcal{R}_t(s_t, a_t, s_{t+1}),$$

Optimality Criteria. Given a starting state s_0, a policy π^* is *optimal* in some optimality criterion $O(\cdot)$, if $O(\pi^*(s_0)) \geq O(\pi(s_0)), \forall \pi \in \Pi$, where Π is the set of all possible policies. Many readers are familiar with the *discounted rewards* criterion.

Note: In the following equations we make use of the law of iterated expectations, namely; that we state $E[R_t]$ instead of $E_\pi[R_t]$ relying on iterated expectation over trajectories and the inner on R_t, conditioned on those trajectories.

$$O_d(\pi(s_0)) := \lim_{T \to \infty} \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t R_t\right] \tag{1}$$

However, a different criterion, often useful in infinite-horizon tasks, is the *average undiscounted rewards* criterion [7, 13, 15, 16, 19]. It is particularly useful in cycling and continuing tasks, where an optimal policy repeatedly goes through the same states:

$$O_a(\pi(s_0)) := \lim_{T \to \infty} \mathbb{E}\left[\frac{1}{T}\sum_{t=0}^{T-1} R_t\right] \tag{2}$$

Average Rewards Reinforcement Learning (ARL). In a seminal paper, Schwartz [16] presented the R-Learning algorithm, the basic tabular model-free reinforcement learning algorithm that optimizes policies for the average reward criterion (Eq. (2)). The paper introduces the notation ρ^π to denote the average reward $O_a(\pi(s_0))$ of a policy π. The R-Learning algorithm seeks to optimize ρ, by learning an optimal policy. R-Learning is structurally identical to Q-Learning (e.g., in terms of selecting actions).

It then takes two update steps:

First, it updates the Q table entry using Eq. (3):

$$Q_{t+1}(s_t, a_t) \leftarrow \alpha\left(r_t - \rho_t + \max_{a'} Q_t(s_{t+1}, a') - Q_t(s_t, a_t)\right). \tag{3}$$

Then, and *only if the action a_t was selected on-policy* (i.e., was not an exploratory action), R-Learning also updates the parameter ρ, which maintains the average reward of the learned policy across all states:

$$\rho_{t+1} \leftarrow \beta\left(r_t + \max_{a'} Q_t(s_{t+1}, a') - \max_a Q_{t+1}(s_t, a) - \rho_t\right). \tag{4}$$

In essence, the Q table is updated using the *marginal reward* with respect to the average policy reward ρ, which is actually the optimization goal of the learning process. The reason for this is to prevent Q from increasing without bound. Instead, Q is bounded by the difference with ρ (see [7, 13, 16, 19] for details).

Over the years, there have been investigations of R-Learning variants and novel algorithms for model-based average reward learning [20], deep neural-network approaches [12, 17], and theoretical investigations. Mahadevan [13] provides an early synthesis; Dewanto et al. [7] provides a recent comprehensive review.

The use of two stochastic approximators in R-Learning, α in Eq. (3) and β in Eq. (4) is a noted weakness of R-Learning [7]. Many variants focus on modifying the ρ update steps to improve the practical performance of R-Learning. However, to the best of our knowledge, all variants—with one exception, below—have assumed that the underlying environment is an MDP. This allows them to estimate the average reward ρ using the standard arithmetic exponential moving average.

2.2 Undiscounted Rewards in Semi-Markov Decision Processes

Many real world decision processes unfold over infinite horizons with unknown stopping times, where neither the termination point nor the duration between actions is fixed or known in advance. Such settings are naturally modeled by *semi-Markov decision processes* (SMDPs), which generalize MDPs by introducing stochastic, state-action-dependent holding times (*sojourn* times) between decisions.

An SMDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega \rangle$, where $\mathcal{S}$, $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{R}$ are defined as above (MDP). SMDPs relax the unit-time MDP assumption by admitting *stochastic holding times*, captured by the function $\Omega : \mathbb{N} \times \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathcal{P}(\mathbb{R}^+)$. Analogous to the definition of the distribution $\mathcal{R}_t(s, a, s')$, and the random variable R_t, we define the distribution $\Omega_t(s, a, s')$ and random variable Υ_t.

When an action $a \in A$ is taken in state s at time $t \in \mathbb{N}$, the new state s' is stochastically determined by $\mathcal{T}$, and the *sojourn time* τ_t is sampled from by $\Omega_t(s, a, s')$, analogous to how the reward r_t is drawn.

The average reward optimization criterion is a natural fit for SMDPs. Rewards are naturally weighted by the time, and are therefore best understood as a reward rate (lump sum reward, divided by the sojourn time). It is then straightforward to consider ρ as the *average reward rate*—reward *per unit time*—of a stationary policy π (Eq. (5)), and the optimal policy yielding an optimal ρ: $\rho^* = \sup_\pi \rho^\pi$.

$$\rho^{\pi} \;=\; \lim_{T\to\infty} \mathbb{E}\left[\frac{\sum_{t=0}^{T-1} R_t}{\sum_{t=0}^{T-1} \Upsilon_t}\right]. \tag{5}$$

Estimation of the average reward *rate* ρ is the key to model-free average reward reinforcement learning in SMDP settings.

Average Reward RL in SMDPs: *R*-Learning with Reward Rates. To the best of our knowledge, only two reinforcement learning algorithms for average reward optimization in SMDPs: SMART and Relaxed-SMART [9].

SMART [5] builds on *R*-Learning, modifying it as follows: First, the *R*-Learning Q update step (Eq. (3)) is modified to compute the marginal update to Q based on the average *rate* ρ, rather than the average reward. This is done using τ_t:

$$Q_{t+1}(s_t, a_t) \leftarrow \alpha\left(r_t - \rho_t \cdot \tau_t + \max_{a'} Q_t(s_{t+1}, a') - Q_t(s_t, a_t)\right). \tag{6}$$

Essentially, this allows a comparison between the reward received r_t (with holding time τ_t), and the expected reward, given the reward rate ρ_t.

If (and only if) action a_t was selected on-policy, SMART also incrementally updates ρ so as to track the average reward rate of the policy. It does this by tracking the total sum of the rewards received so far X_t, and the total sum of the holding times received so far Y_t:

$$Y_{t+1} \leftarrow Y_t + \tau_t, \qquad X_{t+1} \leftarrow X_t + r_t,$$
$$\rho_{t+1} \leftarrow \frac{X_{t+1}}{Y_{t+1}}. \tag{7}$$

This calculation uses the long-term sample average rate $\frac{\sum_{i=0}^{t} r_t}{\sum_{i=0}^{t} \tau_t}$ to approximate the expected reward rate ρ in Eq. (5).

The choice to approximate the average reward rate ρ by the time sample average (Eq. (7)) is unusual in the context of reinforcement learning, because it makes a strong assumption as to the stationarity of the rewards and duration. As long as the rewards and holding times are stationary, the approximation is generally correct. Otherwise, Eq. (7) can be dominated by rare outcomes (outliers), expand dramatically with unbounded rewards (or holding times), or, be affected by latent regime shifts in rewards or holding times. SMART will then fail to estimate ρ correctly.

The Relaxed-SMART algorithm [9] attempts to address this strong assumption of stationary rewards and holding times, by estimating ρ in a different manner, which facilitates assigning more weight to recent values. Rather than using the long-term average (Eq. (7)), Relaxed-SMART uses the ratio of average rewards to average durations (Eq. (8)):

$$\bar{\tau}_{t+1} \leftarrow \beta\bar{\tau}_t + (1-\beta)\tau_t, \qquad \bar{r}_{t+1} \leftarrow \beta\bar{r}_t + (1-\beta)r_t,$$
$$\rho_{t+1} \leftarrow \frac{\bar{r}_{t+1}}{\bar{\tau}_{t+1}}. \tag{8}$$

3 HARMONIC *R*-LEARNING

We develop a novel *R*-Learning variant that can be used with reward rates, i.e., in SMDP settings. Section 3.1 presents the idea of using the harmonic mean as a an alternative basis for estimating the average rate from the reward and holding time generated from each step. We show that this approach is not reliant on the assumption that rewards and holding times are independent of each other, as previous approaches assume. Then, in Section 3.2 we address the limitations of using the standard harmonic mean (e.g., not defined for rates of mixed signs). We introduce a novel alternative, the modified harmonic mean operator $\mathbb{H}$, and investigate its axiomatic properties. Section 3.3 then shows how it can be used in *R*-Learning.

3.1 The Harmonic Mean (Is Not Enough)

We observe that there is a different approach to averaging rates, using the harmonic mean. The *harmonic* mean (Definition 1 below) is the correct averaging calculation to be used with rates and flows (and indeed used in engineering and science in areas focusing on such variables, e.g., electric resistance, liquid flow rates, etc.).

Definition 1 (Harmonic mean, $\mathcal{H}$)**.** Let $X = (x_1, \ldots, x_n)$ be a multiset of non-zero numbers $x_i \in \mathbb{R}$. The *harmonic mean* of X is given by:

$$\mathcal{H}(X) \;=\; \mathcal{H}(x_1, \ldots, x_n) \;=\; \frac{n}{\sum_{k=1}^{n} \frac{1}{x_k}}. \tag{9}$$

For example, given data about the speeds at which a fixed distance D is traveled—first at 20kph and then again at 40kph—one cannot infer the average speed is 30kph, as that would be incorrect: Traveling D at 20kph takes twice the amount of time it takes to travel D at 40kph. Instead, the correct average speed is 26.6kph, which is the harmonic mean of 20 and 40.

Based on this observation, we may want to estimate ρ as the incrementally computed harmonic mean of the sampled reward rates $\frac{r_1}{\tau_1}, \frac{r_2}{\tau_2}, \ldots$, where each rate $\frac{r_t}{\tau_t}$ resulting from a step t is used as single atomic datum (x_t in Eq. (9)).

Calculating ρ without separating the aggregations of the reward and holding time can yield the same value as the ratio of averages in some specific cases. We compare the behavior of aggregating the values using the ratio of the arithmetic means vs. using the harmonic mean.

We begin with an intuition. Suppose we have taken n steps in the SMDP, resulting in a sequence of rewards $(r_1, r_2, \ldots r_n)$ and an associated sequence of holding times $(\tau_1, \tau_2, \ldots \tau_n)$. Intuitively, when the rewards and the holding times are completely independent of each other, then a change in the internal order of the sequence of rewards does not matter when assessing the average reward $\bar{r}$. But if the rewards and holding times are not independent, then changing the order of reward (e.g., exchanging r_1 and r_n) necessitates changing the order of the holding times (τ_1 and τ_n), to preserve the coupling between the rewards and their duration. The ratio of averages ignores this coupling and would produce an incorrect average rate, while the harmonic mean (which averages the rates $\frac{r_1}{\tau_1}, \ldots \frac{r_n}{\tau_n}$ directly) respects this coupling.

Theorem 1 formalizes this intuition. It shows that the harmonic mean and the ratio of arithmetic means will differ under specific conditions.

Theorem 1. *Let $r_t > 0$ and $\tau_t > 0$ for $t = 1, \ldots, n$. Define the sample mean*

$$A(x) := \frac{1}{n}\sum_{t=1}^{n} x_t,$$

and the (mean-based) covariance

$$\mathrm{Cov}(X,Y) := A(XY) - A(X)\,A(Y). \tag{10}$$

Define

$$Q := \frac{A(r)}{A(\tau)} = \frac{\sum_{t=1}^{n} r_t}{\sum_{t=1}^{n} \tau_t}, \qquad H := \frac{n}{\sum_{t=1}^{n} \frac{\tau_t}{r_t}} = \frac{1}{A(\tau/r)}.$$

Then

$$Q = H \iff \mathrm{Cov}\Big(r, \frac{\tau}{r}\Big) = 0.$$

Equivalently, $Q \neq H$ iff the covariance is nonzero.

Proof Sketch. Using the definition of covariance,

$$\mathrm{Cov}(X,Y) = A(XY) - A(X)A(Y) \iff$$
$$A(XY) = A(X)A(Y) + \mathrm{Cov}(X,Y).$$

Take $X = r$ and $Y = \tau/r$. For generality, we drop the time t subscript for the purpose of this proof in r_t and since $r_t \neq 0$,

$$XY = r \cdot (\tau/r) = \tau,$$

hence

$$A(\tau) = A(r)\,A(\tau/r) + \mathrm{Cov}\Big(r, \frac{\tau}{r}\Big).$$

By the definition of H,

$$H = \frac{n}{\sum_{t=1}^{n} \frac{\tau_t}{r_t}} = \frac{1}{A(\tau/r)} \implies A(\tau/r) = \frac{1}{H}.$$

Substituting into the previous identity gives

$$A(\tau) = \frac{A(r)}{H} + \mathrm{Cov}\Big(r, \frac{\tau}{r}\Big),$$

so

$$H = \frac{A(r)}{A(\tau) - \mathrm{Cov}\big(r, \frac{\tau}{r}\big)}.$$

Therefore,

$$H = \frac{A(r)}{A(\tau)} \iff \mathrm{Cov}\Big(r, \frac{\tau}{r}\Big) = 0.$$

Since $Q = \frac{A(r)}{A(\tau)}$, this is exactly $Q = H \iff \mathrm{Cov}\big(r, \frac{\tau}{r}\big) = 0$. □

In other words, as long as the relationship between rewards and time induces the Covariance in equation 10 to be **non zero**, we expect to observe a difference in ρ between SMART and Relaxed-Smart, relative to the $\mathbb{H}$. Under finite second moments, such non zero covariance between r and $\frac{\tau}{r}$ implies that the variables are not independently distributed. In short, nonzero covariance forces the two mean measures to differ.

In other words, as long as the relation between the reward and time is non-linear, we expect to observe a difference in the ρ value between SMART and Relaxed-SMART in comparison to $\mathbb{H}$.

Unfortunately, the harmonic mean has strict domain restrictions: First, it is not defined for zero reward values (as they have no defined reciprocals). Second, if values in X have mixed signs, $\mathcal{H}$ loses critical properties expected of means. Most glaringly, it loses the *internality* property (value between the minimum and maximum datum, see below) expected with any measure of central tendency: $\mathcal{H}(-20, 40) = -80$, for example.

These restrictions severely limit the applicability of the harmonic mean to the use of reinforcement learning using reward rates in the model-free R-Learning algorithm, as in many environments, rewards can be zero (though their associated sojourn times are not zero), or negative. Below, therefore, we propose a new mean operator, modified harmonic mean operator, to address these restrictions.

3.2 The modified harmonic mean operator $\mathbb{H}$

We formalize a mixed-sign operator $\mathbb{H}$ that admits zero and mixed-sign datums (Definition 2). It is the weighted arithmetic mean of (i) the zero-valued datums, and the harmonic means of (ii) the positive datums, and of (iii) the negative datums.

Definition 2 (modified harmonic mean operator $\mathbb{H}$)**.** Let $X = (x_1, \ldots, x_n)$ be a multi-set of numbers $x_i \in \mathbb{R}$. Let X^+, X^-, X^0 be partitions of X such that $X^+ = (x_i | x_i \in X, x_i > 0)$, $X^- = (x_i | x_i \in X, x_i < 0)$, and $X^0 = (x_i | x_i \in X, x_i = 0)$. The modified harmonic mean operator $\mathbb{H}$ is defined as:

$$\begin{aligned}\mathbb{H}(X) :=& \frac{|X^+| \cdot \mathcal{H}(X^+) \;+\; |X^-| \cdot \mathcal{H}(X^-) \;+\; |X^0| \cdot 0}{|X^+| + |X^-| + |X^0|} \\ =& \frac{|X^+| \cdot \mathcal{H}(X^+) \;+\; |X^-| \cdot \mathcal{H}(X^-)}{|X|},\end{aligned} \tag{11}$$

where $\mathcal{H}$ is the harmonic mean (Eq. (9)):

$$\mathcal{H}(X^+) = \frac{|X^+|}{\sum\limits_{x_i \in X^+} \left(\frac{1}{x_i}\right)}, \qquad \mathcal{H}(X^-) = \frac{|X^-|}{\sum\limits_{x_j \in X^-} \left(\frac{1}{x_j}\right)}. \tag{12}$$

We consider theoretical properties of modified harmonic mean operator. There are several axiomatic definitions of means, or general *mean-type aggregation functions* [2, 4, 10, 18], which we refer to as *general means* for brevity. We will show that the modified harmonic mean operator is a general mean, but not a quasi-arithmetic mean (the class that includes the familiar arithmetic, geometric, harmonic means, and many others).

Definition 3 (General Mean)**.** Following [1, 4], an operator $\mathcal{M}$ is a *general mean* if it satisfies the following properties, with respect to all multisets X:

Internality. $\min_i x_i \leq \mathcal{M}(x_1, \ldots, x_n) \leq \max_i x_i$.

Idempotence (reflexivity). $\mathcal{M}(x, \ldots, x) = x$ for any x.

Symmetry (permutation invariance). $\mathcal{M}$ depends only on the values in the finite multiset X, but not their order: for any permutation σ of $(1, \ldots, n)$,

$$\mathcal{M}(a_1, \ldots, a_n) = \mathcal{M}\big(a_{\sigma(1)}, \ldots, a_{\sigma(n)}\big).$$

Monotonicity. If one datum value increases (others fixed), the mean should not decrease. Given $Y = (x_1, \ldots, x_i + k, \ldots, x_n)$, where $k > 0$, $\mathcal{M}(Y) \geq \mathcal{M}(X)$.

An important and familiar subclass of means is *quasi-arithmetic means*, which obeys the above definition, but in addition, has one more property (Definition 4), where by any subgroup of datums $Y \in X$ may be replaced by their subgroup mean $\mathcal{M}(Y)$, without affecting the $\mathcal{M}(X)$ [11, 14, as presented in [1, 4]].

Definition 4 (n-Associativity (replacement invariance, consistency))**.** A mean operator $\mathcal{M}$ is *consistent* if for any multiset $X = (x_1, \ldots, x_n)$, and any *block* $B = (x_i, \ldots, x_j) \subset X$ (for any $1 \leq i, j \leq n$), the values $x_i, \ldots, x_j$ can be replaced by $|B|$ copies of the block mean $\mathcal{M}(B)$, without changing $\mathcal{M}(X)$:

$$\mathcal{M}(x_1, \ldots x_i, \ldots x_j, \ldots, x_n) = \mathcal{M}(x_1, \ldots \mathcal{M}(B), \ldots \mathcal{M}(B), \ldots x_n).$$

The familiar arithmetic mean is a quasi-arithmetic mean. The harmonic mean $\mathcal{H}$ is a quasi-arithmetic mean under the domain restriction that either $\forall x \in X, x > 0$ or $\forall x \in X, x < 0$. If the restriction is removed, the harmonic mean loses the internality property at the very least (for mixed signs), or becomes undefined (for zero values).

We show that where the domain restrictions for the harmonic mean are satisfied, $\mathbb{H}$ generalizes $\mathcal{H}$ (Theorem 2). We then use this result to show that $\mathbb{H}$ is a general mean (Theorem 3), but not a quasi-arithmetic mean (Theorem 4). For brevity, we present proof sketches only.

Theorem 2 ($\mathbb{H}$ generalizes $\mathcal{H}$). *Given a multiset of positive (respectively, negative) values X^+ (X^-), $\mathbb{H}(X^+) = \mathcal{H}(X^+)$ ($\mathbb{H}(X^-) = \mathcal{H}(X^-)$.*

Proof sketch. Assume $X = (x_1, \ldots, x_n)$, where $\forall i, x_i > 0$. Then by Definition 2,

$$\begin{aligned}\mathbb{H}(X) =& \frac{|X^+| \cdot \mathcal{H}(X^+) \ + \ |X^-| \cdot \mathcal{H}(X^-)}{|X|} && \text{(by Eq. (11).)}\\ =& \frac{|X| \cdot \mathcal{H}(X) + 0 \cdot \mathcal{H}(\emptyset)}{|X|} && \text{(since } X^+ = X.)\\ =& \mathcal{H}(X).\end{aligned}$$

The same argument holds when $\forall i, x_i < 0$. □

Theorem 3 shows that unlike $\mathcal{H}$, $\mathbb{H}$ is a general mean across all values ($\forall i, x_i \in \mathbb{R}$):

Theorem 3. *$\mathbb{H}$ is a general mean.*

Proof sketch. We show $\mathbb{H}$ satisfies the properties listed in Definition 3: internality, idempotence, permutation invariance, and monotonicity.

Internality (satisfied). $\mathbb{H}$ is bounded between the harmonic mean of the positives and that of the negatives. Each of these, by itself, maintains internality. So $\mathbb{H}$ always lies between the dataset minimum and maximum.

Idempotence (satisfied). If $x = 0$, $\mathbb{H}(x, \ldots, x) = 0 = x$. If $x > 0$, or $x < 0$, then $\mathbb{H} = \mathcal{H}$ (see Theorem 2) and thus maintains idempotence (this can also be proven directly).

Symmetry (satisfied). $\mathbb{H}(X)$ is unchanged by reordering datums in X, as the partitioning of X into the multisets X^+, X^-, X^0 is not dependent on the ordering.

Monotonicity (satisfied). Note the denominator $|X|$ does not change. Pick any x_i in X^+ (respectively, X^-). Add a constant $k > 0$. If $x_i + k$ remained in X^+ (X^-), then by the monotonicity of $\mathcal{H}$, $\mathcal{H}(X^+)$ (or $\mathcal{H}(X^-)$) will increase and thus $\mathbb{H}(X)$ will increase. If $x_i + k = 0$, then x_i is eliminated from X^-. The negative-valued $\mathcal{H}(X^-)$ will increase, and thus $\mathbb{H}(X)$ will increase.

Conclusion. As $\mathbb{H}$ satisfies Definition 3, it is a general mean. □

The modified harmonic mean operator $\mathbb{H}$ generalizes the harmonic mean $\mathcal{H}$ where the latter is defined, but $\mathbb{H}$ is defined where $\mathcal{H}$ is not (for zero-value datums). Also, unlike $\mathcal{H}$, $\mathbb{H}$ maintains the centrality property when datums of mixed signs are present. It is therefore a general mean. However, it is not a quasi-arithmetic mean (Theorem 4).

Theorem 4. *$\mathbb{H}$ is not a quasi-arithmetic mean.*

Proof sketch. We note that consistency is maintained within pure-sign blocks, as a function of the consistency of $\mathcal{H}$ with respect to pure-sign blocks. However, if a block B has mixed signs, consistency is not guaranteed. Assume for contradiction that $\mathbb{H}$ is a quasi-arithmetic mean. It is therefore consistent (per Definition 4). Let $X = (1, 1, -1, 4)$, and $B = (1, -1) \subset X$. Then,

$$\mathbb{H}(1, 1, -1, -4) = -0.3, \qquad \mathbb{H}(1, -1) = 0.$$

However, replacing B by $\mathbb{H}(B)$ yields $\mathbb{H}(1, 0, 0, -4) = -0.75$. This contradicts the assumption that $\mathbb{H}$ is consistent, and hence it cannot be a quasi-arithmetic mean. □

Finally, it remains to show that $\mathbb{H}$ maintains the property discussed in Theorem 1, which distinguishes it from the ratio of averages. We show in Theorem 5 that if the distribution of all rewards is not independent of all times, then when rewards are decomposed into their positive, negative, and zero values, at least one of three reward-collections must also not be independent of time, hence maintaining the distinction.

Theorem 5. *If the distributions of all rewards are **not independent** of the distribution of all holding times, then when rewards are partitioned based on their sign (positive, negative, zero), at least one of three reward partition collections must also be dependent on the respective partition of holding times.*

Proof. Skipped due to space limitations. See Appendix A. □

3.3 Using $\mathbb{H}$ in Average Reward RL

We introduce a novel average reward RL algorithm, called Harmonic R-Learning. The Q update rule for Harmonic R-Learning is the same as for SMART (ρ is multiplied by τ_t, Eq. (6)). We develop an incremental approximator of $\mathbb{H}$, to be used in updating the average reward rate ρ.

We use p (respectively, n) to maintain the exponential moving *arithmetic* means of the positive rewards (negative rewards, resp.). Analogously, we use w_p, w_n, w_z to maintain the exponential moving *arithmetic* approximation of $|H^+|, |H^-|, |H^0|$, resp. We denote by E^+, E^- the exponentially moving approximations of the harmonic means $\mathcal{H}(H^+), \mathcal{H}(H^-)$, resp.

We set r, the reciprocal of the rate $\frac{r_t}{\tau_t}$ (when $r_t \neq 0$)

$$r := \begin{cases} 0, & r_t = 0, \\ \frac{\tau_t}{r_t}, & \text{otherwise.} \end{cases}$$

Then, we set the reciprocals p, n and the associated weights w_p, w_n, using the indicator function $\mathbf{1}$ (1 when the condition holds, 0 otherwise). We also set w_z:

$$p \leftarrow \beta \cdot (\mathbf{1}\{r > 0\} \cdot r - p), \qquad w_p \leftarrow \beta \cdot (\mathbf{1}\{r > 0\} - w_p), \quad (13)$$

$$n \leftarrow \beta \cdot (\mathbf{1}\{r < 0\} \cdot r - n), \qquad w_n \leftarrow \beta \cdot (\mathbf{1}\{r < 0\} - w_n), \quad (14)$$

$$w_z \leftarrow \beta \cdot (\mathbf{1}\{r = 0\} - w_z). \quad (15)$$

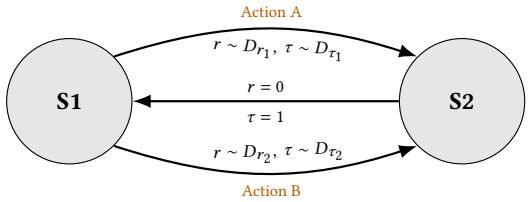


Figure 1: Two-state SMDP with stochastic reward and sojourn time for the transition $s1 \rightarrow s2$, and deterministic return for the transition $s2 \rightarrow s1$.

Now we can compute E^+, E^- by inverting the reciprocals n, p (accounting for the weight w_p, w_n)

$$E^+ \leftarrow \begin{cases} 0, & p = 0, \\ \dfrac{w_p}{p}, & \text{otherwise.} \end{cases}, \qquad E^- \leftarrow \begin{cases} 0, & n = 0, \\ \dfrac{w_n}{n}, & \text{otherwise.} \end{cases} \tag{16}$$

Finally, the exponentially moving $\mathbb{H}$ of ρ is given by

$$\rho_{t+1} \leftarrow \frac{w_p \cdot E^+ + w_n \cdot E^-}{w_p + w_n + w_z}. \tag{17}$$

The Harmonic R-Learning algorithm is an R-Learning variant, where ρ is a stochastically-estimated average reward rate. In that, it is completely analogous to R-Learning, but where the latter is appropriate only for MDP environments, Harmonic R-Learning is intended for use in the more general SMDP case. Note that by setting $\tau_t = 1$, one gets back the familiar ρ computation via the arithmetic mean (Eq. (4)). It is therefore a proper generalization of R-Learning.

4 EMPIRICAL EVALUATION

In addition to the theoretical analysis of modified harmonic mean operator, we also investigate it empirically in extensive experiments, using deceptively simple—and very challenging—SMDPs (Section 4.1), and using real-world bitcoin data (Section 4.2).

4.1 A Family of Deceptively Simple SMDP

We use simulated two-state SMDPs to empirically evaluate the difference between SMART and Relaxed SMART algorithms, and Harmonic R-Learning.

4.1.1 Simulation Setup. Fig. 1 shows the small, continuing two-state SMDP environment that we use to evaluate the algorithms. The environment is constructed from states s_1, s_2 and actions A, B. From s_1, action A deterministically transitions to s_2 with a reward r_t and duration (sojourn time) τ_t. Action B behaves similarly but uses a different (r_t, τ_t) generator. From s_2, the process deterministically returns to s_1 with reward 0 and duration 1, yielding a continuing task (no terminal states). Each episode starts at s_1, and the reward/duration generators are reset at episode boundaries to ensure identical sampling across runs.

As shown in Section 3.1, the algorithms' different approaches to estimating ρ lead to identical results when the rewards and durations are independent of each other. We therefore focus on the case where the reward and durations are not independent. Furthermore, had the rate itself been stationary, then SMART would already be sufficient, and thus we choose a non-stationary rate.

Specifically, we set up the rewards and durations such that action A may appear promising within the episode length, though its rewards and durations are independent. Action B may appear less promising initially, but its rewards and durations are not independent. The objective of the algorithms is to discover that the actual expected rate of Action B is much better in the long term.

In the experiments, action A's rewards are taken from a function linear in t, with a slope of 0.05. The durations of action A are stochastically sampled from the normal distribution $\mathcal{N}(\mu = 1, \sigma = 0.1)$, capped at 0.001 to prevent zero or negative durations. The rewards and durations of action A are clearly independent (in the long run).

Action B's rewards use a *drifting log-scaled sine* (denoted *SinLogD*)—a drifting sine wave that scales and amplifies stronger as time passes (Eq. (18)). The durations use a *drifting log-scaled cosine* (denoted *CosLogD) (Eq.* (19)):

$$SinLogD(t, \text{offset}, \text{log_scale}) = \big(\sin(t) + \text{offset}\big)\, 10^{\,t \cdot \text{log_scale}}. \tag{18}$$

$$CosLogD(t, \text{offset}, \text{log_scale}) = \big(\cos(t) + \text{offset}\big)\, 10^{\,t \cdot \text{log_scale}}. \tag{19}$$

We ensure the rewards and durations of action B are not independent by using a hyperparameter to set the log_scale in the rewards and durations. Given a baseline value l, the reward function uses log_scale $= v$, and the duration function uses log_scale $= \frac{v}{2}$. For simplicity, the value of the offset throughout the experiment is set to 10 for both functions. While offset of zero still works, the addition of fast-iterating positive and negative values is hard to grasp visually, hence we keep the values positive only using the offset.

Fig. 2 shows the first thousand steps of each of the reward (*SinLogD*) and duration (*CosLogD*) functions and their ratio: $\frac{\text{SinLogD}(t,10,0.001)}{\text{CosLogD})\ t,10,0.0005)}$. The figure illustrates how quickly the non-stationary rewards and durations expand, while their non-stationary ratio expands much more slowly.

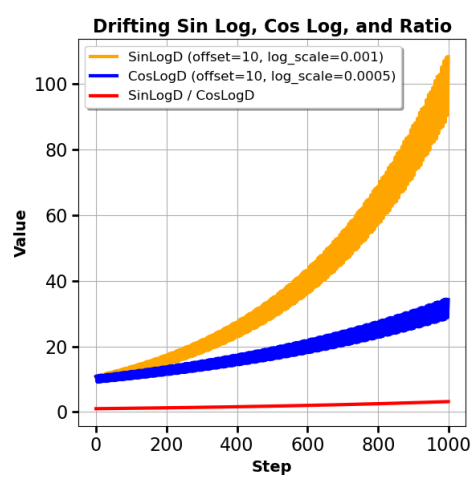


Figure 2: A thousand-step sample of $\text{SinLogD}(t, 10, 0.001)$ **and** $\text{CosLogD}(t, 10, 0.0005)$ **with their ratio.**

Fig. 3 shows a specific setting, where action B's rewards and durations are defined as in Fig. 2. The learning algorithms would be exposed to the first 1000 steps (shown as a dotted vertical line). Up to that point, action A seems superior. However, plotting ahead to 10,000 steps, we see that the rate resulting from action B overtakes that of action A. Action B is the optimal choice. These settings are a good test for a learning algorithm's ability to uncover the underlying trend of the non-stationary average rate.

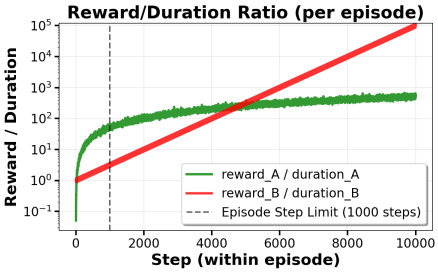
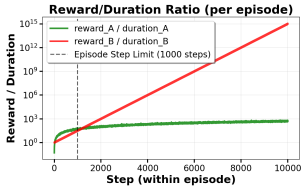


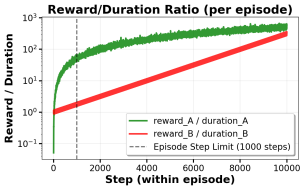
Figure 3: Ratio of $\mathrm{SinLogD}(t, 10, 0.001)/\mathrm{CosLogD}(t, 10, 0.0005)$ **and** $\mathrm{Linear}(0.05)/\mathrm{Normal}(1, 0.1)$ **for the first** 10,000 **steps. The dotted line shows the learning cutoff, meaning that the algorithms don't see the cross at** 4,500 **where Action B starts to result in a better value than Action A.**

In the experiments, we vary the difficulty of the settings by changing only the baseline log_scale parameter controlling the relation between the rewards and durations of action B. We test baseline values between 10^{-5} and 10^{-1} with 30 log-scale steps in this range.

Fig. 4 shows that smaller values lead to more difficult problems. log_scale values greater than 0.003 are "easier" because Action B dominates within the 1,000 steps of the learning. Smaller values make the settings harder (moving the crossover point further and further away).



(a) $\mathrm{SinLogD}(t, 10, 0.003)$ / $\mathrm{CosLogD}(t, 10, 0.0015)$



(b) $\mathrm{SinLogD}(t, 10, 0.0005)$ / $\mathrm{CosLogD}(t, 10, 0.00025)$

Figure 4: Different log scales in relation to $\mathrm{Linear}(0.05)/\mathrm{Normal}(1, 0.1)$ **for the first** 10,000 **steps. As the log scale value lowers—e.g., (b) above—the cross-over point takes place later, at a step unobserved by the learning algorithm.**

Each algorithm is run on 4 episodes, each of 1000 steps, with a fixed exploration rate of 0.2. To assure ourselves of some robustness to the learning rate parameters α, β, we use $\alpha \in [10^{-4}, 0.1]$ and $\beta \in [10^{-4}, 10^{-1}]$. For both, we use 20 even log-scale steps, i.e., a total of 400 α, β combinations. Note that SMART does not use a β parameter.

4.1.2 Simulation Results. Fig. 5 plots the results from the experiments. The horizontal axis measures the difficulty of the SMDP environment, using a log-scale. Each point along the axis is a baseline value used for the log_scale parameter for action B, as described. As we move from left to right, the learning settings become harder and harder, as the optimality of action B is not observable by the algorithms, and the cross-over point is further away in the horizon. The vertical axis measures the percentage of successful trials to learn the optimal policy (action B), out of the 400 attempts (with different α, β parameters).

The figure shows that Harmonic R-Learning is more robust and successful in these experiments. It is shown to be more robust, because its success percentage is higher in all cases (i.e., it is less sensitive to specific α, β learning rates). It is shown to be more successful, as it continues to maintain a high percentage of successes, even as the environment becomes too difficult for SMART and Relaxed SMART, which drop to 0% success around the middle of the horizontal axis.

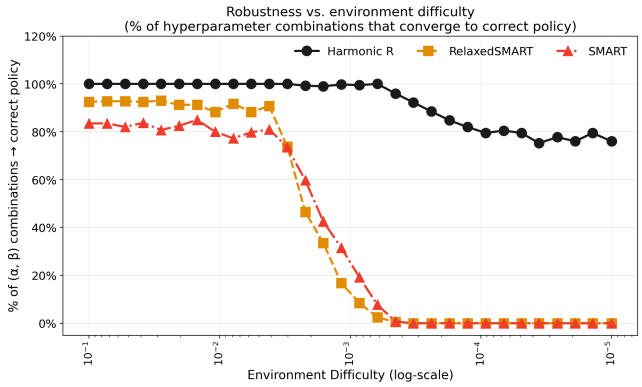


Figure 5: Comparing the success rate of the different algorithms, as learning problems become harder (left to right). The Harmonic R**-Learning algorithm clearly dominates (shown in black).**

These results are very promising, but this simple environment has only positive values and is artificially structured. In the next set of experiments, we deal with a challenging environment that was not constructed manually and has negative and zero values.

4.2 Bitcoin environment

We test the algorithms on highly volatile, generally non-stationary data: real-world Bitcoin (BTC) market trading data. Bitcoin is uniquely suitable: it trades continuously (24/7) with high liquidity and dense minute-level observations, enabling rate-sensitive, infinite-horizon analysis without episodic boundaries. Publicly available data provide over 500,000 one-minute observations per year, with gapless bars where each close equals the next open[1].

[1]Data source: https://www.kaggle.com/datasets/swaptr/bitcoin-historical-data, licensed for open use.

BTC exhibits documented non-stationary features: heavy tails, volatility clustering, regime shifts, and explicit ruin barriers (which would adversarially stress robustness) [3, 6, 8, 21]. These properties make simulated Bitcoin trading a very demanding RL testbed. In addition, publicly-available data permits precise ablations.

4.2.1 Experimental Setup. We compare the behavior of the three algorithms: Relaxed-SMART, SMART, and Harmonic R-Learning. As a reminder, SMART addresses rates, but uses the long-term sample arithmetic average to approximate the expected optimal reward rate ρ—brittle when rewards are non-stationary. Relaxed-SMART uses EMA to handle non-stationarity, but bases the approximation on the ratio of averages, which assumes the rewards and durations are independent. The arithmetic mean. Harmonic R-Learning uses the stochastically estimated modified harmonic mean operator (Eq. (17)) to target the rate directly.

Data Preparation. We use the entire set of minute-by-minute BTC historical prices (2012 to mid-September 2025). To eliminate dependency on starting conditions, we partitioned the data into 21 non-overlapping segments, each (except for the last one) with 350,000 one-minute observations. This supports walk-forward evaluation without look-ahead leakage.

At each decision point (every minute), the algorithm is given the choice of either buying a bitcoin or selling one. It gets a reward depending not only on the direction of the decision, but also on the difference between the price at the moment the action was executed and the price at the end of the one-minute interval.

An MDP version of this task would assume a fixed duration of action execution, e.g., at every minute, a decision is made, and the action is taken. So, for instance, the reward depends on the entire one-minute price difference. An SMDP version of this task would instead account for the duration of the action itself. If an action takes τ seconds, the reward is computed by the difference between the end price and the price τ seconds from the 1-minute interval beginning (determined in these experiments by a linear interpolation between the opening and closing minute prices). Quicker actions thus use more of the 1-minute price difference.

We generate two SMDP versions of the data. In the *random* version, the duration of actions is selected uniformly from the interval of 5 to 45 seconds. In the second version (*scaled*), the duration is *scaled* by the value of the reward using min-max scaling according to the minimum and maximum reward values. Larger rewards impose larger durations, and vice versa (within 5–45 seconds).

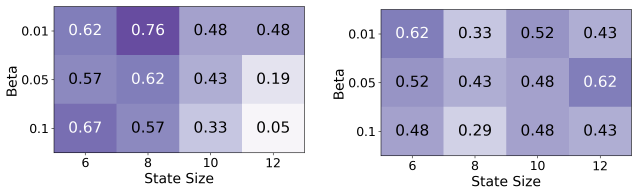
Experiment Parameters. At each timestamp, the value of BTC can either increase or decrease. Therefore, the state space is defined as a window capturing the recent trend directions of BTC. For example, with a state size of three, the state represents the last three movement directions of BTC, such as $\{up, up, down\}$ or $\{down, up, up\}$. We investigate four values: 3, 6, 9, and 12.

We sweep over combinations of state sizes (affecting all algorithms) and mean smoothing parameter β with values: 0.01, 0.05, and 0.1 (affecting only Harmonic and Relaxed-SMART). Each (state, β) pair is evaluated independently on all 21 segments for both delay time windows. All algorithms used an exploration rate of 0.2 with a ϵ decay rate of 0.999 and a learning rate of 0.001. For each of the 21 files and their 350,000 data points, every algorithm was run with 30 different seeds.

Evaluation Metric. We measure the accumulating rewards (profits and losses) for each algorithm instance, at each step. The final result of the accumulated value is therefore the accumulated reward over an entire segment. The results from 30 runs are averaged and used to represent the performance of each agent.

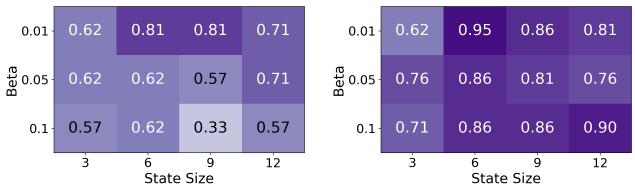
For each window size, method, and (state, β), we aggregate the values across all agents, keeping the mean and std of the accumulated reward at each timestamp. Only on-policy actions are taken into account, any exploration action is ignored for fairness.

4.2.2 Results. To compare all β and state size values for all segments, we summarize the results as heatmaps in two axes: the state size and the β values. These are shown in Fig. 6 for the random SMDP version, and in Fig. 7 for the scaled SMDP version. In both figures, subfigure (a) shows the heatmap of wins (percentage in which Harmonic R-Learning was better, out of the 21 cases) against SMART, and subfigure (b) shows the same against Relaxed SMART. Values around 0.5 hint at equivalent performance.



(a) Harmonic vs SMART **(b) Harmonic vs Relaxed-SMART**

Figure 6: Win-Ratio of harmonic for all 21 segments of the *random* version. The rewards and durations are independent. Harmonic is not performing better than SMART and Relaxed-SMART.



(a) Harmonic vs SMART **(b) Harmonic vs Relaxed-SMART**

Figure 7: Win-Ratio of harmonic for all 21 segments of the *scaled* version. The dependency of the reward and time gives the Harmonic algorithm a significant advantage, creating a clear win in most cases.

5 CONCLUSIONS AND FUTURE WORK

This paper makes several contributions. First, it analyzes the requirements for an average rewards learning algorithm for semi-Markov decision processes. It shows that SMART and Relaxed-SMART, the only algorithms for this task, as a result of using the sample average, can fail when rewards are not in a linear relation to time. To address this, we introduce a modified harmonic mean operator, a general mean operator that generalizes the harmonic mean where the latter is applicable, and obeys key axiomatic properties of general means.

We proved the generalization and the inclusion of the modified harmonic mean operator in the class of general means. We also proved that it is not a member of the quasi-arithmetic means class. As the modified harmonic mean operator generalizes the harmonic mean, it is mathematically correct for use in averaging reward rates, as needed. We demonstrate empirically, using a proof-of-concept simulation and a highly volatile real-world Bitcoin trading data (millions of data points), that use of Harmonic R-Learning leads to improved results over existing algorithms for average reward maximization.

REFERENCES

[1] János Aczél. 1948. On mean values. *Bull. Amer. Math. Soc.* 54, 4 (1948), 392–400.

[2] Gleb Beliakov, Andrea Pradera, and Tomasa Calvo. 2007. *Aggregation Functions: A Guide for Practitioners.* Springer, Berlin.

[3] Haim Bodek. 2013. *The Problem of HFT: Collected Writings on High Frequency Trading and Stock Market Structure Reform.* Decimus Capital Markets, LLC, CT, USA.

[4] Peter S. Bullen. 2003. *Handbook of Means and Their Inequalities.* Kluwer Academic Publishers, Dordrecht.

[5] Tapas K. Das, Abhijit Gosavi, Sridhar Mahadevan, and Nicholas Marchalleck. 1999. Solving Semi-Markov Decision Problems Using Average Reward Reinforcement Learning. *Management Science* 45, 4 (1999), 560–574. https://doi.org/10.1287/mnsc.45.4.560 arXiv:https://doi.org/10.1287/mnsc.45.4.560

[6] Nassim Dehouche. 2021. Scale matters: The daily, weekly and monthly volatility and predictability of Bitcoin, Gold, and the S& P 500. arXiv:2103.00395 [q-fin.ST] https://arxiv.org/abs/2103.00395

[7] Vektor Dewanto, George Dunn, Ali Eshragh, Marcus Gallagher, and Fred Roosta. 2020. Average-reward model-free reinforcement learning: a systematic review and literature mapping. *CoRR* abs/2010.08920 (2020). arXiv:2010.08920 https://arxiv.org/abs/2010.08920

[8] Stanisław Drożdż, Ludovico Minati, Paweł Oświęcimka, Michał Stanuszek, and Marcin Watorek. 2018. Bitcoin Market Route to Maturity? Evidence from Return Fluctuations, Temporal Correlations and Multiscaling Effects. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 28, 7 (2018), 071101.

[9] Abhijit Gosavi. 2004. Reinforcement learning for long-run average cost. *European Journal of Operational Research* 155 (06 2004), 654–674. https://doi.org/10.1016/S0377-2217(02)00874-3

[10] John E. Gray and Andrew Vogt. 2013. The Mean: Axiomatics, Generalizations, Applications. arXiv:1210.3908 [math.PR] https://arxiv.org/abs/1210.3908

[11] Andrei N. Kolmogorov. 1930. Sur la notion de la moyenne. *Atti della Accademia Nazionale dei Lincei. Rendiconti, Classe di Scienze Fisiche, Matematiche e Naturali* 12 (1930), 388–391.

[12] Xiaoteng Ma, Xiaohang Tang, Li Xia, Jun Yang, and Qianchuan Zhao. 2021. Average-Reward Reinforcement Learning with Trust Region Methods. arXiv:2106.03442 [cs.LG] https://arxiv.org/abs/2106.03442

[13] Sridhar Mahadevan. 1996. Average Reward Reinforcement Learning: Foundations, Algorithms, and Empirical Results. *Machine Learning* 22, 1–3 (1996), 159–195. https://doi.org/10.1007/BF00114727

[14] Mitio Nagumo. 1930. Über eine Klasse der Mittelwerte. *Japanese Journal of Mathematics* 7 (1930), 71–79.

[15] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley, New York.

[16] Anton Schwartz. 1993. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning* (Amherst, MA, USA) *(ICML'93)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 298–305.

[17] Rishi Shah, Yuqian Jiang, Justin Hart, and Peter Stone. 2020. Deep R-Learning for Continual Area Sweeping. arXiv:2006.00589 [cs.LG] https://arxiv.org/abs/2006.00589

[18] Nozer D. Singpurwalla and Boya Lai. 2020. What Does the "Mean" Really Mean? arXiv:2003.01973 [stat.OT] https://arxiv.org/abs/2003.01973

[19] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (2 ed.). MIT Press, http://incompleteideas.net/book/the-book-2nd.html. http://incompleteideas.net/book/the-book-2nd.html

[20] Prasad Tadepalli and DoKyeong Ok. 1994. *H-Learning: A Reinforcement Learning Method for Optimizing Undiscounted Average Reward.* Technical Report 94-30-1. Oregon State University, Department of Computer Science.

[21] Yaoyue Tang, Karina Arias-Calluari, M. N. Najafi, Michael S. Harré, and Fernando Alonso-Marroquin. 2025. Stylized Facts of High-Frequency Bitcoin Time Series. arXiv:2402.11930 [q-fin.ST] https://arxiv.org/abs/2402.11930

A PROOF OF THEOREM 5

In real world scenarios, we need an average rate method that can extend to treat scenarios of: positive, negative and zero value rewards with non zero time. Extending our proof of Theorem 4, below we show that this proof holds when the joint distribution of rewards (where positive, negative and zero values are commingled) is decomposed into its three components, as needed to generalize equation 11. To do this, we work with σ-algebras in this proof. By way of brief mention, a σ-algebra is a neat mathematical representation of a set of variables where the requirement that once we can show a property about a certain variable (such as reward or time), we can then extrapolate it to also talking about any combinations of variables (e.g., "all rewards" or "all time periods"). In this way, the σ-algebra defines a property (or properties) of a collection of events where unbounded random variables r and τ generate the event collections $\sigma(X)$ and $\sigma(T)$ that contain all measurable statements about r and τ, respectively. In this way this extends to infinite horizon R-learning tasks naturally.

The motivation for this proof is to show that, even when harmonic means are decomposed into their mutually exclusive partitions based on sign, the resulting distributions of rewards and time, retain any dependence structure (or more formally, remain not independent). This lingering dependence structure in effect, guarantees that harmonic mean will exhibit a ρ value that differs from that produced by SMART and Relaxed-SMART even when decomposed into a calculation using the positive, negative and zero components of r from equation 12. Hence we posit, as we did in stating Theorem 5: *if the distributions of all rewards is* ***not independent*** *of the distribution of all holding times, then when rewards are partitioned based on their sign (positive, negative, zero), at least one of three reward partition collections must also be dependent on the respective partition of holding times.*

To expand further consider the following. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. Let

$$r : \Omega \to \mathbb{R}, \qquad \tau : \Omega \to \mathbb{R}$$

be random variables (with arbitrary distributions), where r denotes the rewards and τ denotes the holding time. In an SMDP, a single execution of the policy from one decision epoch to the next produces a random *experience segment* (also called a sojourn), e.g. a tuple

$$\omega := (s_0, a_0, r_0, \tau_0,\ s_1, a_1, r_1, \tau_1,\ \ldots,\ s_N, a_N, r_N, \tau_N)$$

containing the visited states, chosen actions, intermediate rewards, and elapsed times until the next decision epoch (or termination). The set Ω is the collection of all such possible segments ω; in other words, Ω is the sample space of all outcomes that could be generated by the environment dynamics together with the (possibly stochastic) policy.

The σ-algebra $\mathcal{F}$ is the collection of events (subsets of Ω) to which we assign probabilities, such as "the segment terminates within 2 seconds" or "the total reward exceeds 10". The probability measure $\mathbb{P}$ is the distribution over segments induced by the environment's stochastic transition-and-holding-time kernel together with the policy.

A random variable is a measurable function of the realized segment $\omega \in \Omega$. In particular, we define

$$r : \Omega \to \mathbb{R}, \qquad \tau : \Omega \to (0, \infty)$$

by letting $r(\omega)$ be the total reward accrued during the segment ω (e.g. cumulative reward until the next decision epoch) and letting $\tau(\omega)$ be the corresponding holding time (elapsed physical time) of that segment.

Now, assume we decompose the rewards space into three disjoint collections (positive, negative and zero rewards). Formally, let

$$H_+,\ H_-,\ H_0 \subseteq \mathbb{R},$$

$$H_+ \cap H_- = \emptyset, H_+ \cap H_0 = \emptyset, H_- \cap H_0 = \emptyset, H_+ \cup H_- \cup H_0 = \mathbb{R}.$$

Define the three reward-collection events

$$A_+ := \{r \in H_+\}, \qquad A_- := \{r \in H_-\}, \qquad A_0 := \{r \in H_0\}.$$

Equivalently, define the reward-collection label

$$Z : \Omega \to \{+, -, 0\}, \qquad Z(\omega) = \begin{cases} + & \text{if } r(\omega) \in H_+, \\ - & \text{if } r(\omega) \in H_-, \\ 0 & \text{if } r(\omega) \in H_0. \end{cases}$$

Then, if the reward-collection label is not independent of time, i.e.

$$Z \not\perp\!\!\!\perp T.$$

it follows that *at least one* of the three collections is itself not independent of time. That is, there exists $\kappa \in \{+, -, 0\}$ and there exists an event $B \in \sigma(T)$ such that

$$\mathbb{P}(A_\kappa \cap B) \neq \mathbb{P}(A_\kappa)\,\mathbb{P}(B).$$

Note: the proof below relies on the property of *finite additivity* of probability, which is a direct consequence of Kolmogorov's axioms (Kolmogorov, 1933): for pairwise disjoint events $E_1, \ldots, E_m \in \mathcal{F}$,

$$\mathbb{P}\left(\bigcup_{j=1}^{m} E_j\right) = \sum_{j=1}^{m} \mathbb{P}(E_j).$$

Proof. We prove by contradiction. Assume that $Z \neq \perp\!\!\!\perp T$, but that nevertheless *each* of the three (disjoint) reward collections, is independent of time. In σ-algebra notation, this means that for every

$$B \in \sigma(T),$$

we have

$$\mathbb{P}(A_+ \cap B) = \mathbb{P}(A_+)\mathbb{P}(B) \tag{20}$$

$$\mathbb{P}(A_- \cap B) = \mathbb{P}(A_-)\mathbb{P}(B) \tag{21}$$

$$\mathbb{P}(A_0 \cap B) = \mathbb{P}(A_0)\mathbb{P}(B) \tag{22}$$

Now, note that $\sigma(Z)$ is generated by the three atoms A_+, A_-, A_0. In particular, every event $C \in \sigma(Z)$ can be written as a union of a subcollection of these three events. That is, for every $C \in \sigma(Z)$ there exists a set $S \subseteq \{+, -, 0\}$ such that

$$C = \bigcup_{\kappa \in S} A_\kappa. \tag{23}$$

Fix any $C \in \sigma(Z)$ and any $B \in \sigma(T)$. Using (23) and the distributive property of intersection,

$$C \cap B = \left(\bigcup_{\kappa \in S} A_\kappa\right) \cap B = \bigcup_{\kappa \in S} (A_\kappa \cap B).$$

Since the events A_+, A_-, A_0 are disjoint, the events $(A_\kappa \cap B)$ are also disjoint across $\kappa \in S$. Therefore, by finite additivity (Kolmogorov, 1933),

$$\mathbb{P}(C \cap B) = \sum_{\kappa \in S} \mathbb{P}(A_\kappa \cap B).$$

Applying (20) to each term,

$$\mathbb{P}(C \cap B) = \sum_{\kappa \in S} \mathbb{P}(A_\kappa)\mathbb{P}(B) = \mathbb{P}(B) \sum_{\kappa \in S} \mathbb{P}(A_\kappa).$$

Finally, since the A_κ are disjoint, finite additivity gives

$$\sum_{\kappa \in S} \mathbb{P}(A_\kappa) = \mathbb{P}\left(\bigcup_{\kappa \in S} A_\kappa\right) = \mathbb{P}(C),$$

and therefore

$$\mathbb{P}(C \cap B) = \mathbb{P}(C)\mathbb{P}(B) \qquad \text{for all } C \in \sigma(Z) \text{ and all } B \in \sigma(T). \tag{24}$$

Equation (24) is exactly the definition of independence of the σ-algebras $\sigma(Z)$ and $\sigma(T)$, i.e.

$$\sigma(Z) \perp\!\!\!\perp \sigma(T),$$

which implies

$$Z \perp\!\!\!\perp T.$$

This contradicts our assumption that $Z \neq \perp\!\!\!\perp T$. Hence, our supposition that all three collections are independent of time must be false. Therefore, at least one of the three collections must be dependent on time, i.e. there exists $\kappa \in \{+, -, 0\}$ and $B \in \sigma(T)$ such that

$$\mathbb{P}(A_\kappa \cap B) \neq \mathbb{P}(A_\kappa)\,\mathbb{P}(B).$$

Q.E.D

□