Optimizing for the Future in Non-Stationary MDPs

Yash Chandak [1] **Georgios Theocharous** [2] **Shiv Shankar** [1]
Martha White [3] **Sridhar Mahadevan** [1 2] **Philip S. Thomas** [1]

Abstract

Most reinforcement learning methods are based upon the key assumption that the transition dynamics and reward functions are fixed, that is, the underlying Markov decision process is stationary. However, in many real-world applications, this assumption is violated and using existing algorithms may result in a performance lag. To proactively search for a good *future* policy, we present a policy gradient algorithm that maximizes a forecast of *future* performance. This forecast is obtained by fitting a curve to the counter-factual estimates of policy performance over time, without explicitly modeling the underlying non-stationarity. The resulting algorithm amounts to a non-uniform reweighting of past data, and we observe that *minimizing* performance over some of the data from past episodes can be beneficial when searching for a policy that *maximizes* future performance. We show that our algorithm, called Prognosticator, is more robust to non-stationarity than two online adaptation techniques, on three simulated problems motivated by real-world applications.

1. Introduction

Policy optimization algorithms in RL are promising for obtaining general purpose control algorithms. They are designed for Markov decision processes (MDPs), which model a large class of problems (Sutton & Barto, 2018). This generality can facilitate application to a variety of real-world problems. However, most existing algorithms assume that the environment remains stationary over time.

This assumption is often violated in practical problems of interest. For example, consider an assistive driving system. Over time, tires suffer from wear and tear, leading to increased friction and thus, change in the system dynamics. Similarly, in almost all human-computer interaction applications, e.g., automated medical care, dialogue systems, and marketing, human behavior changes over time. In such scenarios, if the automated system is not adapted to take such changes into account, or if it is adapted only after observing such changes, then the system might quickly become sub-optimal, incurring severe loss (Moore et al., 2014). This raises our main question: *how do we build systems that proactively search for a policy that will be good for the future MDP?*[1]

In this paper we present a policy gradient based approach to search for a policy that maximizes the forecasted future performance when the environment is non-stationary. To capture the impact of changes in the environment on a policy's performance, first, the performance of the policy during the past episodes is estimated using counter-factual reasoning. Subsequently, a regression curve is fit to these estimates to model the performance trend of the policy over time, thereby enabling the forecast of future performance. By differentiating this performance forecast with respect to the parameters of the policy being evaluated, we obtain a gradient-based optimization procedure that proactively searches for a policy that will perform well in the future.

Recently, Al-Shedivat et al. (2017) and Finn et al. (2019) also presented methods that search for initial parameters that are effective when the objective is changing over time. These approaches are complementary to our own, as they could be additionally applied to set the initial parameters of our algorithms. In our empirical study, we show how the adaptation procedure of their methods alone can result in a performance lag that is mitigated by our method, which explicitly captures the trend of the objective due to non-stationarity. A detailed survey on other approaches can be found in the work by Padakandla (2020).

[1]University of Massachusetts, MA, USA. [2]Adobe Research, CA, USA. [3]University of Alberta, AB, Canada. Correspondence to: Yash Chandak <ychandak@cs.umass.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

[1]Note that even when an RL agent is being trained on a stationary environment, the observed transition tuples come from a 'non-stationary' distribution. This is due to the changing state distribution induced by updates in the policy parameters over the course of the training. While such non-stationarity exists in our setup as well, it is not the focus of this work. Here, 'non-stationarity' refers to the transition dynamics and reward function of an environment changing across episodes as described in Section 3.

Advantages: The proposed method has the following advantages: (a) It does not require modeling the transition function, reward function, or how either changes in a non-stationary environment, and thus scales gracefully with respect to the number of states and actions in the environment. (b) Irrespective of the complexity of the environment or the policy parameterization, it concisely models the *effect* of changes in the environment on a policy's performance using a *univariate* time-series. (c) It is data-efficient in that it leverages all available data. (d) It mitigates performance lag by proactively optimizing performance for episodes in both the immediate and near future. (e) It degenerates to an estimator of the ordinary policy gradient if the system is stationary, meaning that there is little reason not to use our approach if there is a possibility that the system *might* be non-stationary.

Limitations: The method that we propose is limited to settings where (a) non-stationarity is governed by an exogenous process (i.e., past actions do not impact the underlying non-stationarity), which has no auto-correlated noise, and (b) performance of every policy changes smoothly over time and has no abrupt breaks/jumps. Further, we found that our method is sensitive to a hyper-parameter that trades off exploration and exploitation in the non-stationary setting. We conclude the paper with a discussion of these limitations.[2]

2. Notation

An MDP $\mathcal{M}$ is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, d^0)$, where $\mathcal{S}$ is the set of possible states, $\mathcal{A}$ is the set of actions, $\mathcal{P}$ is the transition function, $\mathcal{R}$ is the reward function, γ is the discount factor, and d^0 is the start state distribution. Let $\mathcal{R}(s,a)$ denote the expected reward of taking action a in state s. For any given set $\mathcal{X}$, we use $\Delta(\mathcal{X})$ to denote the set of distributions over $\mathcal{X}$. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ is a distribution over the actions conditioned on the state. When π is parameterized using $\theta \in \mathbb{R}^d$, we denote it as π^θ. In a non-stationary setting, as the MDP changes over time, we use $\mathcal{M}_k$ to denote the MDP during episode k. In general, we will use sub-script k to denote the episode number and a super-script t to denote the time-step within an episode. S_k^t, A_k^t, and R_k^t are the random variables corresponding to the state, action, and reward at time step t, in episode k. Let H_k denote a trajectory in episode k: $(s_k^0, a_k^0, r_k^0, s_k^1, a_k^1, ..., s_k^T)$, where T is the finite horizon. The value function evaluated at state s, during episode k, under a policy π is $v_k^\pi(s) = \mathbb{E}[\sum_{j=0}^{T-t} \gamma^j R_k^{t+j} | S_k^t = s, \pi]$, where conditioning on π denotes that the trajectory in episode k was sampled using π. The start state objective for a policy π, in episode k, is defined as $J_k(\pi) := \sum_s d_0(s) v_k^\pi(s)$. Let $J_k^* = \max_\pi J_k(\pi)$ be the performance of an optimal policy for $\mathcal{M}_k$. Often we write θ in place of π^θ when the dependence on θ is important.

[2]Note that there is a source of confusion that stems from the misconception that a non-stationary MDP can be converted into a stationary (PO)MDP by considering an 'expanded' (PO)MDP consisting of all possible variants of the given problem. However, this perspective only shifts the source of non-stationarity from the transition function and the reward function to the starting-state distribution.

3. Problem Statement

To model non-stationarity, we let an exogenous process change the MDP from $\mathcal{M}_k$ to $\mathcal{M}_{k+1}$, i.e., between episodes. Let $\{\mathcal{M}_k\}_{k=1}^\infty$ represent a sequence of MDPs, where each MDP $\mathcal{M}_k$ is denoted by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}_k, \mathcal{R}_k, \gamma, d^0)$.

In many problems, like adapting to friction in robotics, human-machine interaction, etc., the transition dynamics and rewards function change, but every other aspect of the MDP remains the same throughout. Therefore, we assume that for any two MDPs, $\mathcal{M}_k$ and $\mathcal{M}_{k+1}$, the state set $\mathcal{S}$, the action set $\mathcal{A}$, the starting distribution d^0, and the discount factor γ are the same.

If the exogenous process changing the MDPs is arbitrary and changes it in unreasonable ways, then there is little hope of finding a good policy for the future MDP as $\mathcal{M}_{k+1}$ can be wildly different from everything the agent has observed by interacting with the past MDPs, $\mathcal{M}_1, ..., \mathcal{M}_k$. However, in many practical problems of interest, such changes are smooth and have an underlying (unknown) structure. To make the problem tractable, we therefore assume that both the transition dynamics $(\mathcal{P}_1, \mathcal{P}_2, ...)$, and the reward functions $(\mathcal{R}_1, \mathcal{R}_2, ...)$ vary smoothly over time in a way that ensures there are no abrupt jumps in the performance of any policy.

Problem Statement. We seek to find a sequence of policies that minimizes lifelong regret:

$$\underset{\{\pi_1, \pi_2, ... \pi_k, ...\}}{\arg\min} \sum_{k=1}^{\infty} J_k^* - \sum_{k=1}^{\infty} J_k(\pi_k).$$

4. Related Work

The problem of non-stationarity has a long history and no effort is enough to thoroughly review it. Here, we briefly touch upon the most relevant work and defer a more detailed literature review to the appendix. A more exhaustive survey can be found in the work by Padakandla (2020).

Perhaps the work most closely related to ours is that of Al-Shedivat et al. (2017). They consider a setting where an agent is required to solve test tasks that have different transition dynamics than the training tasks. Using meta-learning, they aim to use training tasks to find an initialization vector for the policy parameters that can be quickly fine-tuned when facing tasks in the test set. In many real-world prob-

lems, however, access to such independent training tasks may not be available *a priori*. In this work, we are interested in the continually changing setting where there is no boundary between training and testing tasks. As such, we show how their proposed online adaptation technique that fine-tunes parameters, by discarding past data and only using samples observed online, can create performance lag and can therefore be data-inefficient. In settings where training and testing tasks do exist, our method can be leveraged to better adapt during test time, starting from any desired parameter vector.

Recent work by Finn et al. (2019) aims at bridging both the continuously changing setting and the train-test setting for supervised-learning problems. They propose continuously improving an underlying parameter initialization vector and running a Follow-The-Leader (FTL) algorithm (Shalev-Shwartz et al., 2012) every time new data is observed. A naive adaption of this for RL would require access to all the underlying MDPs in the past for continuously updating the initialization vector, which would be impractical. Doing this efficiently remains an open question and our method is complementary to choosing the initialization vector. Additionally, FTL based adaptation always lags in tracking optimal performance as it uniformly maximizes performance over all the past samples that might not be directly related to the future. Further, we show that by explicitly capturing the trend in the non-stationarity, we can mitigate this performance lag resulting from the use of an FTL algorithm during the adaptation process.

The problem of adapting to non-stationarity is also related to continual learning (Ring, 1994), lifelong-learning (Thrun, 1998), and meta-learning (Schmidhuber, 1999). Several meta-learning based approaches for fine-tuning a (mixture of) trained model(s) using samples observed during a similar task at test time have been proposed (Nagabandi et al., 2018a;b). Other works have also shown how models of the environment can be used for continual learning (Lu et al., 2019) or using it along with a model predictive control (Wagener et al., 2019). Concurrent work by Xie et al. (2020) also demonstrates how modeling the changes in a *dynamic-parameter* MDP can be useful. We focus on the model-free paradigm and our approach is complementary to these model-based methods.

More importantly, in many real-world applications, it can be infeasible to update the system frequently if it involves high computational or monetary expense. In such a case, even optimizing for the immediate future might be greedy and sub-optimal. The system should optimize for a longer term in the future, to compensate for the time until the next update is performed. None of the prior approaches can efficiently tackle this problem.

5. Optimizing for the Future

The problem of minimizing lifelong regret is straightforward if the agent has access to sufficient samples, in advance, from the future environment, $\mathcal{M}_{k+1}$, that it is going to face (where k denotes the current episode number). That is, if we could estimate the start-state objective, $J_{k+1}(\pi)$, for the future MDP $\mathcal{M}_{k+1}$, then we could search for a policy π whose performance is close to J^*_{K+1}. However, obtaining even a single sample from the future is impossible, let alone getting a sufficient number of samples. This necessitates rethinking the optimization paradigm for searching for a policy that performs well when faced with the future unknown MDP. There are two immediate challenges here:

1. *How can we estimate $J_{k+1}(\pi)$ without any samples from $\mathcal{M}_{k+1}$?*

2. *How can gradients, $\partial J_{k+1}(\pi)/\partial\theta$, of this future performance be estimated?*

In this section we address both of these issues using the following idea. When the transition dynamics $(\mathcal{P}_1, \mathcal{P}_2, ...)$, and the reward functions $(\mathcal{R}_1, \mathcal{R}_2, ...)$ are changing smoothly the performances $(J_1(\pi), J_2(\pi), ...)$ of any policy π will also vary smoothly over time. The impact of smooth changes in the environment thus manifests as smooth changes in the performance of any policy, π. In cases where there is an underlying, unknown, structure in the changes of the environment, one can now ask: *if the performances $J_{1:k}(\pi) \coloneqq (J_1(\pi), ..., J_k(\pi))$ of π over the course of past episodes were known, can we analyze the trend in its past performances to find a policy that maximizes future performance $J_{k+1}(\pi)$?*

5.1. Forecasting Future Performance

In this section we address the first challenge of estimating future performance $J_{k+1}(\pi)$ and pose it as a time series forecasting problem.

Broadly, this requires two components: (a) A procedure to compute past performances, $J_{1:k}(\pi)$, of π. (b) A procedure to create an estimate, $\hat{J}_{k+1}(\pi)$, of π's future performance, $J_{k+1}(\pi)$, using these estimated values from component (a). An illustration of this idea is provided in Figure 1.

Component (a). As we do not have access to the past MDPs for computing the true values of past performances, $J_{1:k}(\pi)$, we propose computing estimates, $\hat{J}_{1:k}(\pi)$, of them from the observed data. That is, in a non-stationary MDP, starting with the fixed transition matrix $\mathcal{P}_1$ and the reward function $\mathcal{R}_1$, we want to estimate the performance $J_i(\pi)$ of a given policy in episode $i \leq k$. Leveraging the fact that the changes to the underlying MDP are due to an exogenous

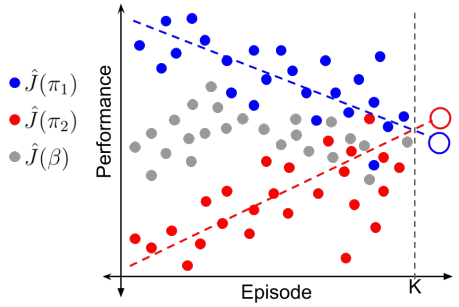

Figure 1. An illustration, where the blue and red filled circles represent counter-factually reasoned performance estimates of policies π_1 and π_2, respectively, using data collected from a given policy β. The open circles represent the forecasted performance of π_1 and π_2 estimated by fitting a curve on the counter-factual estimates represented by filled circles.

processes, we can estimate $J_i(\pi)$ as,

$$J_i(\pi) = \sum_{t=0}^{T} \gamma^t \mathbb{E}\left[R_i^t \middle| \pi, \mathcal{P}_i, \mathcal{R}_i\right], \tag{1}$$

where $\mathcal{P}_i$ and $\mathcal{R}_i$ are also random variables. Next we describe how an estimate of $J_i(\pi)$ can be obtained from (1) using information only from the i^{th} episode.

To get an unbiased estimate, $\hat{J}_i(\pi)$, of π's performance during episode i, consider the past trajectory H_i of the i^{th} episode that was observed when executing a policy β_i. By using counter-factual reasoning (Rosenbaum & Rubin, 1983) and leveraging the per-decision importance sampling (PDIS) estimator (Precup, 2000), an unbiased estimate of $J_i(\pi)$ is thus given by:[3]

$$\hat{J}_i(\pi) := \sum_{t=0}^{H} \left(\prod_{l=0}^{t} \frac{\pi(A_i^l|S_i^l)}{\beta_i(A_i^l|S_i^l)} \right) \gamma^t R_i^t. \tag{2}$$

It is worth noting that computing (2) does not require storing all the past policies β_i, one needs to only store the actions and the probabilities with which these actions were chosen.

Component (b). To obtain the second component, which captures the structure in $\hat{J}_{1:k}(\pi) := (\hat{J}_1(\pi), ..., \hat{J}_k(\pi))$ and predicts future performances, we make use of a forecasting function Ψ that estimates future performance $\hat{J}_{k+1}(\pi)$ conditioned on the past performances:

$$\hat{J}_{k+1}(\theta) := \Psi(\hat{J}_1(\pi), \hat{J}_2(\pi), ..., \hat{J}_k(\pi)). \tag{3}$$

While Ψ can be any forecasting function, we consider Ψ to be an ordinary least squares (OLS) regression model with parameters $w \in \mathbb{R}^{d\times 1}$, and the following input and output variables,

$$X := [1, 2, ..., k]^\top \qquad \in \mathbb{R}^{k\times 1},$$
$$Y := [\hat{J}_1(\pi), \hat{J}_2(\pi), \hat{J}_2(\pi), ..., \hat{J}_k(\pi)]^\top \qquad \in \mathbb{R}^{k\times 1}.$$

For any $x \in X$, let $\phi(x) \in \mathbb{R}^{1\times d}$ denote a d-dimensional basis function for encoding the time index. For example, an identity basis $\phi(x) := \{x, 1\}$, or a Fourier basis, where

$$\phi(x) := \{\sin(2\pi n x | n \in \mathbb{N}_{>0})\} \cup \{\cos(2\pi n x) | n \in \mathbb{N}_{>0}\} \cup \{1\}.$$

Let $\Phi \in \mathbb{R}^{k\times d}$ be the corresponding basis matrix. The solution to above least squares problem is $w = (\Phi^\top \Phi)^{-1}\Phi^\top Y$ (Bishop, 2006) and the forecast of the future performance can be obtained using,

$$\hat{J}_{k+1}(\pi) = \phi(k+1)w = \phi(k+1)(\Phi^\top\Phi)^{-1}\Phi^\top Y. \tag{4}$$

This procedure enjoys an important advantage – by just using a univariate time-series to estimate future performance, it bypasses the need for modeling the environment, which can be prohibitively hard or even impossible. Further, note that $\Phi^\top\Phi \in \mathbb{R}^{d\times d}$, where $d << k$ typically, and thus the cost of computing the matrix inverse is negligible. These advantages allows this procedure to gracefully scale to more challenging problems, while being robust to the size, $|\mathcal{S}|$, of the state set or the action set $|\mathcal{A}|$.

5.2. Differentiating Forecasted Future Performance

In the previous section, we addressed the first challenge and showed how to proactively estimate future performance, $\hat{J}_{k+1}(\theta)$, of a policy π^θ by explicitly modeling the trend in its past performances $\hat{J}_{1:k}(\theta)$. In this section, we address the second challenge to facilitate a complete optimization procedure. A pictorial illustration of the idea is provided in Figure 2.

Gradients for $\hat{J}_{k+1}(\theta)$ with respect to θ can be obtained as follows,

$$\frac{d\hat{J}_{k+1}(\theta)}{d\theta} = \frac{d\Psi(\hat{J}_1(\theta), ..., \hat{J}_k(\theta))}{d\theta}$$
$$= \sum_{i=1}^{k} \underbrace{\frac{\partial\Psi(\hat{J}_1(\theta), ..., \hat{J}_k(\theta))}{\partial\hat{J}_i(\theta)}}_{\text{(a)}} \cdot \underbrace{\frac{d\hat{J}_i(\theta)}{d\theta}}_{(b)}. \tag{5}$$

The decomposition in (5) has an elegant intuitive interpretation. The terms assigned to (a) in (5) correspond to how the future prediction would change as a function of past outcomes, and the terms in (b) indicate how the past outcomes would change due to changes in the parameters of the policy π^θ. In the next paragraphs, we discuss how to obtain the terms (a) and (b).

[3] We assume that $\forall i \in \mathbb{N}$ the distribution of H_i has full support over the set of all possible trajectories of the MDP $\mathcal{M}_i$.

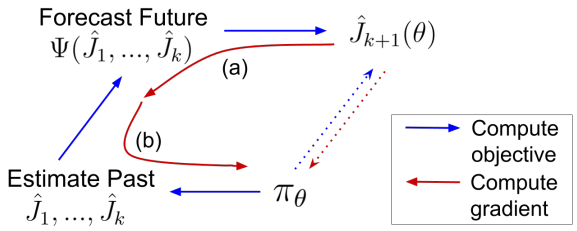

Figure 2. The proposed method from the lens of differentiable programming. At any time k, we aim to optimize policy's parameters, θ, to maximize its performance in the future, $J_{k+1}(\theta)$. However, conventional methods (dotted arrows) can not be used to directly optimize for this. In this work, we achieve this as a composition of two programs: one which connects the policy's parameters to its past performances, and the other which forecasts future performance as a function of these past performances. The optimization procedure then corresponds to taking derivatives through this composition of programs to update policy parameters in a direction that maximizes future performance. Arrows (a) and (b) correspond to the respective terms marked in (5).

To obtain term (a), note that in (4), $\hat{J}_i(\theta)$ corresponds to the i^{th} element of Y, and so using (3) the gradients of the terms (a) in (5) are,

$$\frac{\partial \hat{J}_{k+1}(\theta)}{\partial \hat{J}_i(\theta)} = \frac{\partial \phi(k+1)(\Phi^\top \Phi)^{-1}\Phi^\top Y}{\partial Y_i} = [\phi(k+1)(\Phi^\top \Phi)^{-1}\Phi^\top]_i, \tag{6}$$

where $[Z]_i$ represents the i^{th} element of a vector Z. Therefore, (6) is the *gradient of predicted future performance with respect to an estimated past performance*.

The term (b) in (5) corresponds to the gradient of the PDIS estimate $\hat{J}_i(\theta)$ of the past performance with respect to policy parameters θ. The following Property provides a form for (b) that makes its computation straightforward.

Property 1 (PDIS gradient)**.** *Let* $\rho_i(0,l) := \prod_{j=0}^{l} \frac{\pi^\theta(A_i^j|S_i^j)}{\beta_i(A_i^j|S_i^j)}$.

$$\frac{d\hat{J}_i(\theta)}{d\theta} = \sum_{t=0}^{T} \frac{\partial \log \pi^\theta(A_i^t|S_i^t)}{\partial \theta} \left(\sum_{l=t}^{T} \rho_i(0,l)\gamma^l R_i^l \right).$$

Proof. See Appendix B. □

5.3. Algorithm

We provide a sketch of our proposed *Prognosticator* procedure for optimizing the future performance of the policy in Algorithm 1. To make the method more practical, we incorporated two additional modifications to reduce computational cost and variance.

Algorithm 1: Prognosticator

Input Learning-rate η, time-duration δ, entropy-regularizer λ
Initialize Forecasting function Ψ, Buffer $\mathbb{B}$
while *True* **do**
  # Record a new batch of trajectories using π^θ
  for $episode = 1, 2, ..., \delta$ **do**
    $h = \{(s_{0:T}, a_{0:T}, \Pr(a_{0:T}|s_{0:T}), r_{0:T})\}$
    $\mathbb{B}$.insert(h)
  # Update for future performance
  for $i = 1, 2, ...$ **do**
    # Evaluate past performances
    for $k = 1, 2, ..., |\mathbb{B}|$ **do**
      $\hat{J}_k(\theta) = \sum_{t=0}^{T} \rho(0,t)\gamma^t R_k^t$ ▷ (2)
    # Future forecast and its gradient
    $\mathcal{L}(\theta) = \frac{1}{\delta}\sum_{\Delta=1}^{\delta} \hat{J}_{k+\Delta}(\theta)$ ▷ (4)
    $\theta \leftarrow \theta + \eta \frac{\partial}{\partial \theta}(\mathcal{L}(\theta) + \lambda \mathcal{H}(\theta))$ ▷ (5)

First, it is often desirable to perform an update only after a certain episode interval δ to reduce computational cost. This raises the question: if a newly found policy will be executed for the next δ episodes, should we choose this new policy to maximize performance on just the single next episode, or to maximize the average performance over the next δ episodes? An advantage of our proposed method is that we can easily tune how far in the future we want to optimize for. Thus, to minimize lifelong regret, we propose optimizing for the mean performance over the next δ episodes. That is, $\arg\max_\theta (1/\delta) \sum_{\Delta=1}^{\delta} \hat{J}_{k+\Delta}(\theta)$.

Second, notice that if the policy becomes too close to deterministic, there would be two undesired consequences. (a) The policy will not cause exploration, precluding the agent from observing any changes to the environment in states that it no longer revisits—changes that might make entering those states worthwhile to the agent. (b) In the future when estimating $\hat{J}_{k+1}(\theta)$ using the *past* performance of θ, importance sampling will have high variance if the policy executed during episode $k+1$ is close to deterministic. To mitigate these issues, we add an entropy regularizer $\mathcal{H}$ during policy optimization. More details are available in Appendix D.

6. Understanding the Behavior of Prognosticator

Notice that as the scalar term (a) is multiplied by the PDIS gradient term (b) in (5), the gradient of future performance can be viewed as a weighted sum of off-policy policy gradients. In Figure 3, we provide visualization of the weights

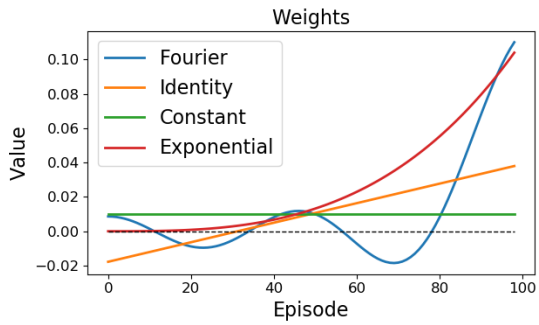

Figure 3. The value of $\frac{\partial \hat{J}_{100}(\theta)}{\partial \hat{J}_i(\theta)}$ for all values of $i \in [1, 99]$ using different basis functions to encode the time index. Notice that many weights are negative when using the identity or Fourier bases.

$\partial \hat{J}_{100}(\theta)/\partial \hat{J}_i(\theta)$ for PDIS gradients of each episode i, when the performance for 100^{th} episode is forecasted using data from the past 99 episodes. For the specific setting when Ψ is an OLS estimator, these weights are independent of Y in (6) and their pattern remains constant for any given sequence of MDPs.

Importantly, note the occurrence of negative weights in Figure 3 when the identity basis or Fourier basis is used, suggesting that the optimization procedure should move towards a policy that had *lower* performance in some of the past episodes. While this negative weighting seems unusual at first glance, it has an intriguing interpretation. We discuss this in the following paragraphs.

To better understand these negative weights, consider a qualitative comparison when weights from different methods in Figure 3 are used along with the performance estimates of policies π_1 and π_2 in Figure 1. Despite having lower estimates of return everywhere, π_2's rising trend suggests that it might have higher performance in the future, that is, $J_{k+1}(\pi_2) > J_{k+1}(\pi_1)$. Existing online learning methods like FTL, maximize performance on all the past data uniformly (green curve in Figure 3). Similarly, the exponential weights (red curve in Figure 3) are representative of approaches that only optimize using data from recent episodes and discard previous data. Either of these methods that use only non-negative weights can never capture the trend to forecast $J_{k+1}(\pi_2) > J_{k+1}(\pi_1)$. However, the weights obtained when using the identity basis would facilitate *minimization* of performances in the distant past and maximization of performance in the recent past. Intuitively, this means that it moves towards a policy whose performance is on a linear rise, as it expects that policy to have better performance in the future.

While weights from the identity basis are useful for forecasting whether $J_{k+1}(\pi_2) > J_{k+1}(\pi_1)$, it cannot be expected that the trend will always be linear as in Figure 1. To be more flexible and allow for any smooth trend, we opt to use the Fourier basis in our experiments. Observe the alternating sign of weights in Figure 3 when using the Fourier basis. This suggests that the optimization procedure will take into account the *sequential differences* in performances over the past, thereby favoring the policy that has shown the most performance increments in the past. This also avoids restricting the performance trend of a policy to be linear.

7. Mitigating Variance

While model-free algorithms for finding a good policy are scalable to large problems, they tend to suffer from high-variance (Greensmith et al., 2004). In particular, the use of importance sampling estimators can increase the variance further (Guo et al., 2017). In our setup, high variance in estimates of past performances $\hat{J}_{1:k}(\pi)$ of π can hinder capturing π's performance trend, thereby making the forecasts less reliable.

Notice that a major source of variance is the availability of only a *single* trajectory sample per MDP $\mathcal{M}_i$, for all $i \in \mathbb{N}$. If this trajectory H_i, generated using β_i is likely when using β_i, but has near-zero probability when using π then the estimated $\hat{J}_i(\pi)$ is also nearly zero. While $\hat{J}_i(\pi)$ is an unbiased estimate of $J_i(\pi)$, information provided by this single H_i is of little use to evaluate $J_i(\pi)$. Subsequently, discarding this from time-series analysis, rather than setting it to be 0, can make the time series forecast more robust against outliers. In comparison, if trajectory H_i is unlikely when using β_i but likely when using π, then not only is H_i very useful for estimating $J_i(\pi)$ but it also has a lower chance of occurring in the future, so this trajectory must be emphasized when making a forecast. Such a process of (de-)emphasizing estimates of past returns using the collected data itself can introduce bias, but this bias might be beneficial in this few-sample regime.

To capture this idea formally, we build upon the insights of Hachiya et al. (2012) and Mahmood et al. (2014), who draw an equivalence between weighted least-squares (WLS) estimation and the weighted importance sampling (WIS) (Precup, 2000) estimator. Particularly, let $G_i := \sum_{t=0}^{T} \gamma^t R_i^t$ be the discounted return of the i^{th} trajectory observed from a stationary MDP, and $\rho_i^\dagger := \rho_i(0, T)$ be the importance ratio of the entire trajectory. The WIS estimator, $\hat{J}^\dagger(\pi)$, of the performance of π in a stationary MDP can then be obtained as,

$$\hat{J}^\dagger(\pi) := \operatorname*{argmin}_{c \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \rho_i^\dagger (G_i - c)^2 = \frac{\sum_{i=1}^{n} \rho_i^\dagger G_i}{\sum_{i=1}^{n} \rho_i^\dagger}.$$

To mitigate variance in our setup, we propose extending WIS. In the non-stationary setting, to perform WIS while capturing the trend in performance over time, we use a

modified forecasting function $\Psi^\dagger$, which is a weighted least-squares regression model with a $d-$dimensional basis function ϕ, and parameters $w^\dagger \in \mathbb{R}^{d\times 1}$,

$$w^\dagger := \underset{c\in\mathbb{R}^{d\times 1}}{\text{argmin}} \frac{1}{n}\sum_{i=1}^{n} \rho_i^\dagger (G_i - c^\top \phi(i))^2. \tag{7}$$

Let $\Lambda \in \mathbb{R}^{k\times k}$ be a diagonal weight matrix such that $\Lambda_{ii} = \rho_i^\dagger$, let $\Phi \in \mathbb{R}^{k\times d}$ be the basis matrix, and let the following be input and output variables,

$$X := [1, 2, ..., k]^\top \qquad \in \mathbb{R}^{k\times 1},$$
$$Y := [G_1, G_2, ..., G_k]^\top \qquad \in \mathbb{R}^{k\times 1}.$$

The solution to the weighted least squares problem in (7) is then given by $w^\dagger = (\Phi^\top \Lambda \Phi)^{-1}\Phi^\top \Lambda Y$ and the forecast of the future performance can be obtained using,

$$\hat{J}_{k+1}^\dagger(\pi) := \phi(k+1)w^\dagger = \phi(k+1)(\Phi^\top \Lambda \Phi)^{-1}\Phi^\top \Lambda Y.$$

$\hat{J}_{k+1}^\dagger(\pi)$ has several desired properties. It incorporates a notion of how relevant each observed trajectory is towards forecasting, while also capturing the desired trend in performance. The forecasts are less sensitive to the importance sampling variances and the entire process is still fully differentiable.

8. Strictly Generalizing the Stationary Setting

As the agent is unaware of how the environment is changing, a natural question to ask is what if the agent wrongly assumed a stationary environment was non-stationary? What is the quality of of the agent's performance forecasts? What is the impact of the negative weights on past evaluations of a policy's performance? Here we answer these questions.

Before stating the formal results, we introduce some necessary notation and two assumptions. Let $J(\pi)$ be the performance of policy π for a stationary MDP. Let $\hat{J}_{k+\delta}(\pi)$ and $\hat{J}_{k+\delta}^\dagger(\pi)$ be the non-stationary importance sampling (NIS) and non-stationary weighted importance sampling (NWIS) estimators of performance δ episodes in future. Further, let the basis function ϕ used for encoding the time index in both Ψ and $\Psi^\dagger$ be such that it satisfies the following conditions: (a) $\phi(\cdot)$ always contains 1 to incorporate a bias/intercept coefficient in least-squares regression (for example, $\phi(\cdot) = [\phi_1(\cdot), ..., \phi_{d-1}(\cdot), 1]$, where $\phi_i(\cdot)$ are arbitrary functions). (b) Φ has full column rank such that $(\Phi^\top\Phi)^{-1}$ exists. Both these properties are trivially satisfied by most basis functions.

With this notation and these assumptions, we then have the following results indicating that NIS is unbiased and consistent like ordinary importance sampling and NWIS is biased and consistent like weighted importance sampling.

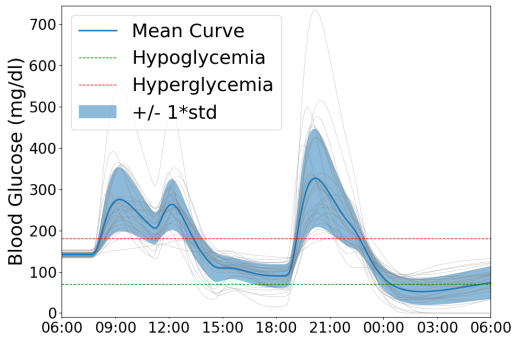

Figure 4. Blood-glucose level of an *in-silico* patient for 24 hours (one episode). Humps in the graph occur at times when a meal is consumed by the patient.

Theorem 1 (Unbiased NIS). *For all $\delta \geq 1$, $\hat{J}_{k+\delta}(\pi)$ is an unbiased estimator of $J(\pi)$, that is $\mathbb{E}[\hat{J}_{k+\delta}(\pi)] = J(\pi)$.*

Theorem 2 (Biased NWIS). *For all $\delta \geq 1$, $\hat{J}_{k+\delta}^\dagger(\pi)$ may be a biased estimator of $J(\pi)$, that is, $\mathbb{E}[\hat{J}_{k+\delta}^\dagger(\pi)] \neq J(\pi)$ always.*

Theorem 3 (Consistent NIS). *For all $\delta \geq 1$, $\hat{J}_{k+\delta}(\pi)$ is a consistent estimator of $J(\pi)$, that is as $N \to \infty$, $\hat{J}_{N+\delta}(\pi) \xrightarrow{a.s.} J(\pi)$.*

Theorem 4 (Consistent NWIS). *For all $\delta \geq 1$, $\hat{J}_{k+\delta}^\dagger(\pi)$ is a consistent estimator of $J(\pi)$, that is as $N \to \infty$, $\hat{J}_{N+\delta}^\dagger(\pi) \xrightarrow{a.s.} J(\pi)$.*

Proof. See Appendix A for all of these proofs. □

Since NWIS is biased and consistent like the WIS estimator, we expect it to have similar variance reduction properties that can potentially make the optimization process more efficient in a non-stationary MDP.

9. Empirical Analysis

This section presents empirical evaluations using several environments inspired by real-world applications that exhibit non-stationarity. In the following paragraphs, we briefly discuss each environment; a more detailed description is available in Appendix D.

Non-stationary Diabetes Treatment:

This environment is based on an open-source implementation (Xie, 2019) of the FDA approved Type-1 Diabetes Mellitus simulator (T1DMS) (Man et al., 2014) for treatment of Type-1 Diabetes. Each episode consists of a day in an *in-silico* patient's life. Consumption of a meal increases the blood-glucose level in the body and if the blood-glucose level becomes too high, then the patient suffers from hyperglycemia and if the level becomes too low, then the patient

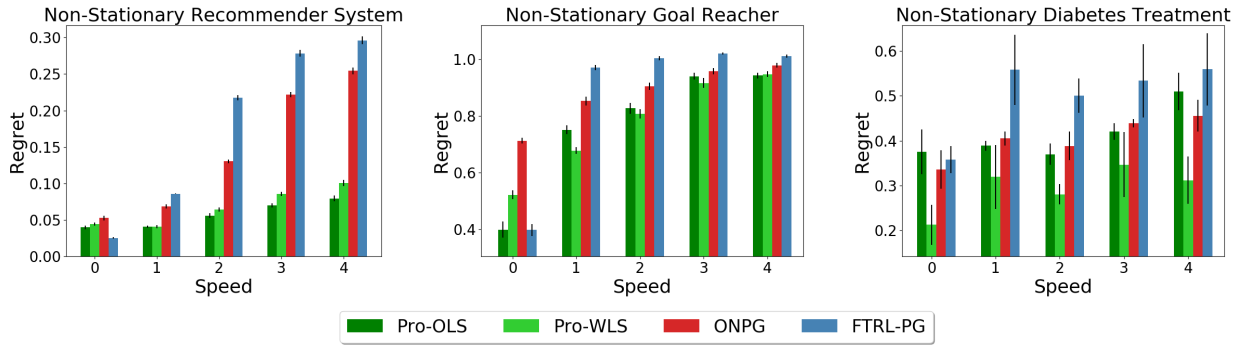

Figure 5. Best performances of all the algorithms obtained by conducting a hyper-parameter sweep over 2000 hyper-parameter combinations per algorithm, per environment. For each hyper-parameter setting, 30 trials were executed for the recommender system and the goal reacher environments, and 10 trials for the diabetes treatment environment. Error bars correspond to the standard error. The x-axis represents how fast the environment is changing and the y-axis represents regret (lower is better). Individual learning curves for each speed, for each domain, is available in Appendix E.

suffers from hypoglycemia. The goal is to control the blood-glucose level of a patient by regulating the insulin dosage to minimize the risk associated with both hyper and hypoglycemia.

However, the insulin sensitivity of a patient's internal body organs vary over time, inducing non-stationarity that should be accounted for. In the T1DMS simulator, we induce this non-stationarity by oscillating the body parameters (e.g., insulin sensitivity, rate of glucose absorption, etc.) between two known configurations available in the simulator.

Non-stationary Recommender System: In this environment a recommender engine interacts with a user whose interests in different items fluctuate over time. In particular, the rewards associated with each item vary in seasonal cycles. The goal is to maximize revenue by recommending an item that the user is most interested in at any time.

Non-stationary Goal Reacher: This is a 2D environment with four (left, right, up, and down) actions and a continuous state set representing the Cartesian coordinates. The goal is to make the agent reach a moving goal position.

For all of the above environments, we regulate the *speed* of non-stationarity to characterize an algorithms' ability to adapt. Higher speed corresponds to a greater amount of non-stationarity; A speed of zero indicates that the environment is stationary.

We consider the following algorithms for comparison:

Prognosticator: Two variants of our algorithm, **Pro-OLS** and **Pro-WLS**, which use OLS and WLS estimators for Ψ.

ONPG: Similar to the adaptation technique presented by Al-Shedivat et al. (2017), this baseline performs purely online optimization by fine-tuning the existing policy using only the trajectory being observed online.

FTRL-PG: Similar to the adaptation technique presented by Finn et al. (2019), this baseline performs Follow-the-(regularized)-leader optimization by maximizing performance over both the current and all the past trajectories.

9.1. Results

In the non-stationary recommender system, as the exact value of J_k^* is available from the simulator, we can compute the true value of regret. However, for the non-stationary goal reacher and diabetes treatment environment, as J_k^* is not known for any k, we use a surrogate measure for regret. That is, let $\tilde{J}_k^*$ be the maximum return obtained in episode k by any algorithm, then we use $(\sum_{k=1}^{N}(\tilde{J}_k^* - J_k(\pi)))/(\sum_{k=1}^{N}\tilde{J}_k^*)$ as the surrogate regret for a policy π.

In the non-stationary recommender system, all the methods perform nearly the same when the environment is stationary. FTRL-PG has a slight edge over ONPG when the environment is stationary as all the past data is directly indicative of the future MDP. It is interesting to note that while FTRL-PG works the best for the stationary setting in the recommender system and the goal reacher task, it is not the best in the diabetes treatment task as it can suffer from high variance. We discuss the impact of variance in later paragraphs.

With the increase in the speed of non-stationarity, performance of both the baselines deteriorate quickly. Of the two, ONPG is better able to mitigate performance lag as it discards all the past data. In contrast, both the proposed methods, Pro-OLS and Pro-WLS, can leverage all the past data to better capture the impact of non-stationarity and thus are consistently more robust to the changes in the environment.

In the non-stationary goal reacher environment, a similar trend as above is observed. While considering all the past

data equally is useful for FTRL-PG in the stationary setting, it creates drastic performance lag as the speed of the non-stationarity increases. Between Pro-OLS and Pro-WLS, in the stationary setting, once the agent nearly solves the task all subsequent trajectories come from nearly the same distribution and thus the variance resulting from importance sampling ratio is not severe. In such a case, where the variance is low, Pro-WLS has less advantage over Pro-OLS and additionally suffers from being biased. However, as the non-stationarity increases, the optimal policy keeps changing and there is a higher discrepancy between distributions of past and current trajectories. This makes the lower variance property of Pro-WLS particularly useful. Having the ability to better capture the underlying trend, both Pro-OLS and Pro-WLS consistently perform better than the baselines when there is non-stationarity.

The non-stationary diabetes treatment environment is particularly challenging as it has a continuous action set. This makes importance sampling based estimators subject to much higher variance. Consequently, Pro-OLS is not able to reliably capture the impact of non-stationarity and performs similar to ONPG. In comparison, the combination of both high variance and performance lag makes FTRL-PG perform poorly across all the speeds. The most advantageous algorithm in this environment is Pro-WLS. As it is designed to better tackle variance stemming from importance sampling, Pro-WLS is able to efficiently use the past data to capture the underlying trend and performs well across all the speeds of non-stationarity.

10. Conclusion

We presented a policy gradient-based algorithm that combines counter-factual reasoning with curve-fitting to proactively search for a good policy for future MDPs. Irrespective of the environment being stationary or non-stationary, the proposed method can leverage all the past data, and in non-stationary settings it can pro-actively optimize for future performance as well. Therefore, our method provides a single solution for mitigating performance lag and being data-efficient.

While the proposed algorithm has several desired properties, many open questions remain. In our experiments, we noticed that the proposed algorithm is particularly sensitive to the value of the entropy regularizer λ. Keeping λ too high prevents the policy from adapting quickly. Keeping λ too low lets the policy overfit to the forecast and become close to deterministic, thereby increasing the variance for subsequent importance sampling estimates of policy performance. While we resorted to hyper-parameter search, leveraging methods that adapt λ automatically might be fruitful (Haarnoja et al., 2018).

Our framework highlights new research directions for studying bias-variance trade-offs in the non-stationary setting. While tackling the problem from the point of view of a univariate time-series is advantageous as the model-bias of the environment can be reduced, this can result in higher variance in the forecasted performance. Developing lower variance off-policy performance estimators is also an active research direction which directly complements our algorithm. In particular, often a partial model of the environment is available and using it through doubly-robust estimators (Jiang & Li, 2015; Thomas & Brunskill, 2016) is an interesting future direction.

Further, there are other forecasting functions, like kernel regression, Gaussian Processes, ARIMA, etc., and some break-point detection algorithms that can potentially be used to incorporate more domain knowledge in the forecasting function Ψ, or make Ψ robust to jumps and auto-correlations in the time series.

11. Acknowledgement

Part of the work was done when the first author was an intern at Adobe Research, San Jose. The research was later supported by generous gifts from Adobe Research. We are thankful to Ian Gemp, Scott M. Jordan, and Chris Nota for insightful discussions and for providing valuable feedback.

Research reported in this paper was also sponsored in part by the CCDC Army Research Laboratory under Cooperative Agreement W911NF-17-2-0196 (ARL IoBT CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Abbasi, Y., Bartlett, P. L., Kanade, V., Seldin, Y., and Szepesvári, C. Online learning in Markov decision processes with adversarially chosen transition probability distributions. In *Advances in Neural Information Processing Systems*, pp. 2508–2516, 2013.

Abdallah, S. and Kaisers, M. Addressing environment non-stationarity by repeating q-learning updates. *The Journal of Machine Learning Research*, 2016.

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

Basso, E. W. and Engel, P. M. Reinforcement learning in non-stationary continuous time and space scenarios. In *Artificial Intelligence National Meeting*, volume 7, pp. 1–8. Citeseer, 2009.

Bastani, M. Model-free intelligent diabetes management using machine learning. *M.S. Thesis, University of Alberta*, 2014.

Besbes, O., Gur, Y., and Zeevi, A. Stochastic multi-armed-bandit problem with non-stationary rewards. In *Advances in Neural Information Processing Systems*, pp. 199–207, 2014.

Bishop, C. M. *Pattern recognition and machine learning*. springer, 2006.

Bowling, M. Convergence and no-regret in multiagent learning. In *Advances in Neural Information Processing Systems*, pp. 209–216, 2005.

Cheevaprawatdomrong, T., Schochetman, I. E., Smith, R. L., and Garcia, A. Solution and forecast horizons for infinite-horizon nonhomogeneous Markov decision processes. *Mathematics of Operations Research*, 32(1):51–72, 2007.

Cheung, W. C., Simchi-Levi, D., and Zhu, R. Reinforcement learning under drift. *arXiv preprint arXiv:1906.02922*, 2019.

Choi, S. P., Yeung, D.-Y., and Zhang, N. L. An environment model for nonstationary reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 987–993, 2000.

Conitzer, V. and Sandholm, T. Awesome: A general multi-agent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43, 2007.

Cuzick, J. A strong law for weighted sums of i.i.d. random variables. *Journal of Theoretical Probability*, 8(3):625–641, 1995.

Even-Dar, E., Kakade, S. M., and Mansour, Y. Experts in a Markov decision process. In *Advances in Neural Information Processing Systems*, pp. 401–408, 2005.

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. *arXiv preprint arXiv:1902.08438*, 2019.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

Foster, D. J., Li, Z., Lykouris, T., Sridharan, K., and Tardos, E. Learning in games: Robustness of fast convergence. In *Advances in Neural Information Processing Systems*, pp. 4734–4742, 2016.

Gajane, P., Ortner, R., and Auer, P. A sliding-window algorithm for Markov decision processes with arbitrarily changing rewards and transitions. *arXiv preprint arXiv:1805.10066*, 2018.

Garcia, A. and Smith, R. L. Solving nonstationary infinite horizon dynamic optimization problems. *Journal of Mathematical Analysis and Applications*, 244(2):304–317, 2000.

Ghate, A. and Smith, R. L. A linear programming approach to nonstationary infinite-horizon Markov decision processes. *Operations Research*, 61(2):413–425, 2013.

Greene, W. H. *Econometric analysis*. Pearson Education India, 2003.

Greensmith, E., Bartlett, P. L., and Baxter, J. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov): 1471–1530, 2004.

Guo, Z., Thomas, P. S., and Brunskill, E. Using options and covariance testing for long horizon off-policy policy evaluation. In *Advances in Neural Information Processing Systems*, pp. 2492–2501, 2017.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Hachiya, H., Sugiyama, M., and Ueda, N. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, 80:93–101, 2012.

Hopp, W. J., Bean, J. C., and Smith, R. L. A new optimality criterion for nonhomogeneous Markov decision processes. *Operations Research*, 35(6):875–883, 1987.

Jacobsen, A., Schlegel, M., Linke, C., Degris, T., White, A., and White, M. Meta-descent for online, continual prediction. In *AAAI Conference on Artificial Intelligence*, 2019.

Jagerman, R., Markov, I., and de Rijke, M. When people change their mind: Off-policy evaluation in non-stationary recommendation environments. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, Melbourne, VIC, Australia, February 11-15, 2019*, 2019.

Jiang, N. and Li, L. Doubly robust off-policy value evaluation for reinforcement learning. *arXiv preprint arXiv:1511.03722*, 2015.

Jong, N. K. and Stone, P. Bayesian models of nonstationary Markov decision processes. *Planning and Learning in A Priori Unknown or Dynamic Domains*, pp. 132, 2005.

Kearney, A., Veeriah, V., Travnik, J. B., Sutton, R. S., and Pilarski, P. M. TIDBD: Adapting temporal-difference step-sizes through stochastic meta-descent. *arXiv preprint arXiv:1804.03334*, 2018.

Lecarpentier, E. and Rachelson, E. Non-stationary Markov decision processes a worst-case approach using model-based reinforcement learning. *arXiv preprint arXiv:1904.10090*, 2019.

Levine, N., Crammer, K., and Mannor, S. Rotting bandits. In *Advances in Neural Information Processing Systems*, pp. 3074–3083, 2017.

Li, C. and de Rijke, M. Cascading non-stationary bandits: Online learning to rank in the non-stationary cascade model. *arXiv preprint arXiv:1905.12370*, 2019.

Li, Y., Zhong, A., Qu, G., and Li, N. Online Markov decision processes with time-varying transition probabilities and rewards. In *Real-world Sequential Decision Making workshop at ICML 2019*, 2019.

Lu, K., Mordatch, I., and Abbeel, P. Adaptive online planning for continual lifelong learning. *arXiv preprint arXiv:1912.01188*, 2019.

Mahmood, A. R., van Hasselt, H. P., and Sutton, R. S. Weighted importance sampling for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems*, pp. 3014–3022, 2014.

Mahmud, M. and Ramamoorthy, S. Learning in non-stationary mdps as transfer learning. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 1259–1260. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Man, C. D., Micheletto, F., Lv, D., Breton, M., Kovatchev, B., and Cobelli, C. The UVA/PADOVA type 1 diabetes simulator: New features. *Journal of Diabetes Science and Technology*, 8(1):26–34, 2014.

Mealing, R. and Shapiro, J. L. Opponent modelling by sequence prediction and lookahead in two-player games. In *International Conference on Artificial Intelligence and Soft Computing*, pp. 385–396. Springer, 2013.

Mohri, M. and Yang, S. Accelerating online convex optimization via adaptive prediction. In *Artificial Intelligence and Statistics*, pp. 848–856, 2016.

Moore, B. L., Pyeatt, L. D., Kulkarni, V., Panousis, P., Padrez, K., and Doufas, A. G. Reinforcement learning for closed-loop propofol anesthesia: A study in human volunteers. *The Journal of Machine Learning Research*, 15(1):655–696, 2014.

Moulines, E. On upper-confidence bound policies for non-stationary bandit problems. *arXiv preprint arXiv:0805.3415*, 2008.

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018a.

Nagabandi, A., Finn, C., and Levine, S. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018b.

Ornik, M. and Topcu, U. Learning and planning for time-varying mdps using maximum likelihood estimation. *arXiv preprint arXiv:1911.12976*, 2019.

Padakandla, S. A survey of reinforcement learning algorithms for dynamically varying environments. *arXiv preprint arXiv:2005.10619*, 2020.

Padakandla, S., J., P. K., and Bhatnagar, S. Reinforcement learning in non-stationary environments. *CoRR*, abs/1905.03970, 2019.

Precup, D. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, 2000.

Rakhlin, A. and Sridharan, K. Online learning with predictable sequences. *arXiv preprint arXiv:1208.3728*, 2013.

Ring, M. B. *Continual learning in reinforcement environments*. PhD thesis, University of Texas at Austin, Texas 78712, 1994.

Rosenbaum, P. R. and Rubin, D. B. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Schmidhuber, J. A general method for incremental self-improvement and multi-agent learning. In *Evolutionary Computation: Theory and Applications*, pp. 81–123. World Scientific, 1999.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Seznec, J., Locatelli, A., Carpentier, A., Lazaric, A., and Valko, M. Rotting bandits are no harder than stochastic ones. *arXiv preprint arXiv:1811.11043*, 2018.

Shalev-Shwartz, S. et al. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

Singh, S., Kearns, M., and Mansour, Y. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pp. 541–548. Morgan Kaufmann Publishers Inc., 2000.

Sinha, S. and Ghate, A. Policy iteration for robust nonstationary Markov decision processes. *Optimization Letters*, 10(8):1613–1628, 2016.

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

Thomas, P. and Brunskill, E. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pp. 2139–2148, 2016.

Thomas, P. S. *Safe reinforcement learning*. PhD thesis, University of Massachusetts Libraries, 2015.

Thomas, P. S., Theocharous, G., Ghavamzadeh, M., Durugkar, I., and Brunskill, E. Predictive off-policy policy evaluation for nonstationary decision problems, with applications to digital marketing. In *Twenty-Ninth Innovative Applications of Artificial Intelligence Conference*, 2017.

Thrun, S. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Wagener, N., Cheng, C.-A., Sacks, J., and Boots, B. An online learning approach to model predictive control. *arXiv preprint arXiv:1902.08967*, 2019.

Wang, J.-K., Li, X., and Li, P. Optimistic adaptive acceleration for optimization. *arXiv preprint arXiv:1903.01435*, 2019a.

Wang, L., Zhou, H., Li, B., Varshney, L. R., and Zhao, Z. Be aware of non-stationarity: Nearly optimal algorithms for piecewise-stationary cascading bandits. *arXiv preprint arXiv:1909.05886*, 2019b.

Xie, A., Harrison, J., and Finn, C. Deep reinforcement learning amidst lifelong non-stationarity. *arXiv preprint arXiv:2006.10701*, 2020.

Xie, J. *Simglucose v0.2.1 (2018)*, 2019. URL https://github.com/jxx123/simglucose.

Yang, S. and Mohri, M. Optimistic bandit convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2297–2305, 2016.

Yu, J. Y. and Mannor, S. Online learning in Markov decision processes with arbitrarily changing rewards and transitions. In *2009 International Conference on Game Theory for Networks*, pp. 314–322. IEEE, 2009.

Zhang, C. and Lesser, V. Multi-agent learning with policy prediction. In *Twenty-fourth AAAI conference on artificial intelligence*, 2010.