Fairness In a Non-Stationary Environment From an Optimal Control Perspective

Zhuotong Chen [1] **Qianxiao Li** [2] **Zheng Zhang** [1]

Abstract

The performance of state-of-the-art machine learning models is observed to degrade in scenarios involving under-represented demographic populations during training. This issue has been extensively studied within a supervised learning framework where data distribution remains unchanged. Nonetheless, real-world use cases often encounter distribution shifts induced by the models in deployment. For example, performance bias against minority users can affect customer retention rates, thereby skewing available data from active users due to the absence of minority user input. This feedback effect further exacerbates the discrepancy across various demographic groups in subsequent time steps. To mitigate this problem, we introduce asymptotic fairness, a criterion that aims at preserving sustained model performance across all demographic populations. In addition, we construct a surrogate retention system, based on existing literature on evolutionary population dynamics, to approximate the dynamics of distribution shifts on active user counts. This system allows the aim of achieving asymptotic fairness to be formulated as an optimal control problem. To evaluate the effectiveness of the proposed method, we design a generic simulation environment that simulates the population dynamics of the feedback effect between user retention and model performance. When we deploy the models to this simulation environment, by considering long-term planning, the optimal control solution outperforms existing baseline methods, demonstrating superior performance.

[1]Department of Electrical & Computer Engineering, University of California, Santa Barbara, CA 93106 [2]Department of Mathematics, National University of Singapore, Singapore & Institute of High Performance Computing, A*STAR, Singapore. Correspondence to: Zhuotong Chen <ztchen@ucsb.edu>.

1. Introduction

In dynamically changing environments, data distributions can evolve over time, giving rise to the phenomenon known as concept drift. Concept drift entails changes in the conditional distribution of the target variable given the input features, while the distribution of the input features themselves might remain fixed (Gama et al., 2014; Schlimmer & Granger, 1986; Widmer & Kubat, 1996). This work considers a distinct case of concept drift, where the performance of a machine learning system can impact the number of active users, while the feature distribution of these users remains fixed. A relevant scenario involves users interacting with personal devices such as Google Home or Amazon Alexa, providing input features like voice commands. Device performance feedback, be it positive (accurate voice identification) or negative (misidentification), can respectively increase user engagement or decrease user retention. This issue becomes more pronounced when populations from different demographic groups are considered (Harwell). A system demonstrating performance bias against minority demographics can result in diminished retention rates for these users. Subsequently, the available training data collected from active users might be insufficient to represent the distribution of minority users, Consequently, this can further amplify the representation disparity issue when the model is fine-tuned based on this training set (Hashimoto et al., 2018).

In this study, we aim to develop machine learning models that engage users from all demographic groups, under the aforementioned population dynamics. This task can be viewed as a trajectory planning problem, wherein the evolution of user engagement is optimized through control design. With the given population dynamics, trajectory planning can be considered as an optimal control problem, solvable by existing methods, such as dynamic programming (Bellman, 1952). However, the generation and evaluation of an optimal control solution pose three challenges. Firstly, the notion of fairness in a non-stationary environment has not been formally defined. Existing fairness definitions aim to measure performance disparities between different demographic groups at a singular time point, such as equal opportunity (Hardt et al., 2016), and demographic parity (Feldman et al., 2015). These definitions fail to adequately encapsulate the goals in a non-stationary environment. Second, solving the optimal control problem requires knowledge of the underlying population dynamics, which are typically complex and inaccessible. Lastly, evaluation of the performance of an optimal control solution (e.g., one generated from some estimated dynamic system) necessitates its deployment on population dynamics with real-world users, an often costly and unattainable requirement. In response to these challenges, our contributions are threefold:

1. We introduce the concept of asymptotic fairness to define the consistent performance across all demographic groups over an extended time period.
2. We propose a surrogate retention system built upon the existing literature on evolutionary population dynamics, from which we formulate the objective of achieving asymptotic fairness as an optimal control problem. To address this control problem, we consider Pontryagin's maximum principle which allows us to solve for the optimal control solution efficiently.
3. Through empirical evaluation, we highlight the advantages of taking into account the underlying dynamics in model design. Our results consistently outperform existing baseline methods, thereby validating the superiority of our approach in terms of performance.

Furthermore, we design a simulator that simulates the non-stationary environment of the user's willingness to retain or churn from a deployed model. This simulator allows for testing the evolutionary fairness property of machine learning models in synthetic population dynamics.

2. Related Works

We review existing literature about fairness in non-stationary settings, and further discuss the intersection of machine learning and optimal control, emphasizing its relevance and applicability to our work.

Fairness problems in the non-stationary setting. Recent studies have brought attention to the potential pitfalls of imposing static fairness constraints (Hardt et al., 2016; Feldman et al., 2015). These constraints can yield unfavorable long-term effects, as demonstrated in the work by (Liu et al., 2018; Zhang et al., 2020). The interplay between algorithmic decisions and individuals' reactions plays a pivotal role in shaping these long-term effects (Zhang et al., 2020). For instance, model decisions can cause changes in the underlying data distribution, subsequently affecting the model's performance in future time steps. (Zhang et al., 2019) recently presented a comprehensive study of the interaction between user retention rates and model decisions in dynamic environments. The most common approach to address this problem is via successive one-step methods, which prioritize fairness for minority demographic groups (Hashimoto et al., 2018). In view of these insights, our research emphasizes the importance of considering the inherent dynamics within a non-stationary environment.

The connection between deep learning and optimal control. Recent works have highlighted the connection between dynamical systems and deep neural networks (E, 2017; Haber & Ruthotto, 2017). This perspective offers valuable theoretical insights for understanding deep learning through an optimal control lens (Liu & Theodorou, 2019). The pioneering work that bridged and extended the classical back-propagation algorithm with optimal control theory was introduced by (Li et al., 2018; Li & Hao, 2018), establishing a direct relationship between the Pontryagon's maximum principle (Kirk, 1970) and gradient-based training. Building upon this foundation, (E et al.) established the mathematical basis for the optimal control viewpoint in deep learning. Moreover, optimal control methods have been applied to tackle challenging issues in deep learning. For instance, (Liu et al., 2020a;b) proposed efficient high-order optimizers using differential dynamic programming, and (Chen et al., 2021; 2022) explored closed-loop controllers to improve robustness against adversarial attacks. These advances highlight the efficacy of optimal control approaches in resolving key challenges of deep learning.

3. Fairness in Non-Stationary Environment

In this section, we discuss the problem setup for fairness in a non-stationary environment, where user retention or churn is conditioned on the model's performance on the data they provide. This configuration leads to a condition for the machine learning models, which we term as asymptotic fairness. This condition requires the models to sustain their performance across all demographic groups over an extended period.

3.1. Problem Description for Fairness in a Non-Stationary Environment

In a non-stationary environment, we consider a predictive model with time-varying model parameters, denoted as $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$. We consider K distinct demographic groups, each comprising N^i users that include both participative and non-participative users of the predictive model. Let us denote Λ_t^i as the number of active users within the i^{th} demographic group at time t. To facilitate our analysis, we normalize Λ_t^i by defining a population density $\lambda_t^i = \frac{\Lambda_t^i}{N^i}$, which falls within the range of 0 to 1. The growth (resp. decay) of λ_t^i indicates that more users from the i^{th} demographic group participate in (resp. leave) the system. We denote $\mathcal{A}_t^i$ as a N-tuple to indicate the active users from the i^{th} demographic group at time step t (e.g. $\mathcal{A}_t^i = (1, 0, 1)$

means that the 1st and 3rd users from the ith group are active at time step t). We formulate the population dynamics of the user's willingness to participate as a Markov decision process and denote it as **population retention system**,

$$\mathcal{A}^i_{t+1} \sim \mathcal{P}(\cdot|\mathcal{A}^i_t, \boldsymbol{\theta}_t, \{\mathbf{z}^{i,n}\}^N_{n=1}, \Phi(\cdot)), \tag{1}$$

where the state space is a N-tuple that indicates the indices of active users, $\mathcal{P}(\cdot)$ denotes the transition probability, $\boldsymbol{\theta}_t$ is the model parameter at time step t, $\mathbf{z}^{i,n}$ represents the features of the n^{th} user from the i^{th} demographic group, this feature vector contains both input data (e.g. voice command) and characteristics (e.g. the probability of a user churn when encountered with a correct prediction, and the probability of a user retaining when receiving a wrong prediction) of the user, $\Phi(\cdot)$ is a loss (or reward) function acting on each individual user. In addition, a certain number of inactive users become active in the system at each time step, this number is conditioned on the model performance of currently active users. In a nutshell, if a user is active at time step t, the activeness of this user at the next time step is conditioned on the model performance of that specific user. Moreover, data features of only active users at each time step are considered an observable state that can be leveraged for generating a model. Once a sequence of models $\{\boldsymbol{\theta}_t\}^T_{t=1}$ is generated, we evaluate its performance using the population retention system defined in Eq. (1).

3.2. The Definition for Asymptotic Fairness

The population retention system, defined in Eq. (1), simulates variation in the active user population density, resulting from the individual model performance of each user. In this context, the deployment of a model with an adequately minimized population risk encourages increased user participation within the system. This encourages more active users to subsequently contribute training data for model refinement, culminating in an enhanced model, which in turn can further decrease population risk in the subsequent time step. This positive feedback loop, given enough time, results in the population risks for all demographic groups converging towards 0 for a large number of total users N^i, while the number densities approach 1. This motivates us to define the concept of asymptotic fairness as follows:

Definition 3.1. Asymptotic fairness A sequence of models satisfies asymptotic fairness if the dynamics it drives satisfy the following condition:

$$\lambda^i_t \to 1, \ \text{ as } t \to \infty \ \ \forall i \in [1, 2, ..., K],$$
$$\text{s.t. } \mathcal{A}^i_{t+1} \sim \mathcal{P}(\cdot|\mathcal{A}^i_t, \boldsymbol{\theta}_t, \{\mathbf{z}^{i,n}\}^N_{n=1}, \Phi(\cdot)),$$
$$\text{where } \ \lambda^i_t = \frac{\sum^{N^i}_{n=1}[\mathcal{A}^i_t]_n}{N^i}.$$

Furthermore, the satisfaction of asymptotic fairness by a sequence of models is implicitly linked to the initial population densities. In scenarios where all demographic groups initially have high population densities, the likelihood of achieving asymptotic fairness increases. Conversely, scenarios with highly imbalanced representations of demographic groups pose significant challenges in meeting this condition. Therefore, the representation of demographic groups plays a critical role in the successful implementation of models adhering to the condition of asymptotic fairness.

Remark 3.2. The mode of convergence as per Definition 3.1 is application-dependent, thus, it varies across different contexts. In this work, we introduce a deterministic surrogate system as defined in Section 4.1. Given the deterministic nature of this surrogate system, we abstain from specifying the mode of convergence pertaining to random variables. The absence of inherent stochasticity in the surrogate system obviates the need for a specific convergence criterion that would otherwise be necessary for the presence of random variables.

Remark 3.3. The concept of asymptotic fairness provides a more precise interpretation of fairness in a non-stationary environment. Prior research has considered disparity amplification (Hashimoto et al., 2018) to assess the representation disparity across all demographic groups at each individual time step. However, the definition of asymptotic fairness diverges from this approach as it emphasizes long-term behavior. To illustrate, consider an extreme scenario where the population densities of all demographic groups concurrently decay to zero. Although this is an undesirable situation, it would nonetheless satisfy the condition of disparity amplification, yet not meet the criterion of asymptotic fairness. Thus, the distinction underscores the importance of considering long-term behavior in fairness definitions, a perspective that asymptotic fairness uniquely encapsulates.

4. An Optimal Control Solution for Asymptotic Fairness

According to Def. 3.1, our goal is to maximize the population densities across all demographic groups within the context of the population retention system, as defined by Eq. (1). Due to the inaccessibility of the underlying dynamics of the population retention system, our initial step involves the construction of a surrogate system to estimate these dynamics. Subsequently, we formulate the condition of asymptotic fairness as an optimal control problem and provide an efficient solver based on Pontryagin's maximum principle (PMP) (Pontryagin, 1987).

4.1. Surrogate Retention System for the Evolutionary Population Dynamics

Our design of the surrogate retention system is rooted in the existing body of literature on evolutionary population dynamics (Cushing, 2019). This system features a low-dimensional state representation, adding to its computational efficiency. Moreover, the system is deterministic and differentiable, which allows for solving the optimal control solution efficiently. Thus, the surrogate retention system not only provides a meaningful connection to evolutionary dynamics but also offers practical advantages in terms of computational efficiency.

Evolutionary population dynamics describes the dynamics of user participation. Difference equations are commonly employed to describe discrete-time dynamics where temporal variations in vital rates are influenced by population density dependencies. Activities of an individual, such as reproduction and survival, may experience fluctuations, thereby contributing to the evolutionary dynamics of population density. Such explicit temporal dependencies can be encapsulated through the optimization of the coefficients of a difference equation over time (Vincent & Brown, 2005). To account for such evolutionary mechanisms, a difference equation population model can be developed (Cushing, 2019). In a simplified context, the population's growth and decay are attributed to births and deaths, respectively. Individuals present at time step $t+1$ are either newcomers within the time frame or survivors from time step t. We model those dynamics as the following discrete dynamic system:

$$\lambda_{t+1}^i = (1-\lambda_t^i)\beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)) + \lambda_t^i \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)), \quad (2)$$

where $\kappa^i(\cdot)$ computes a value that reflects the response of the i$^{\text{th}}$ population on some external controls $\boldsymbol{\theta}_t$ (e.g. medical treatment, resource allocation), $\beta(\cdot)$ and $\sigma(\cdot)$ compute the ratios of newborns and survived population during a time interval respectively. We consider the range of both birth and survival rate functions to be $[0,1]$, in which case, the population densities take the range of $[0,1]$. Moreover, we denote $\boldsymbol{\lambda}_t = [\lambda_t^1, \lambda_t^2, ..., \lambda_t^K]^T$ as a K-dimensional vector. Subsequently, a K-dimensional discrete dynamic system describing the simplified evolutionary population dynamics can be constructed as follows:

$$\boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t), \quad (3)$$

where the i$^{\text{th}}$ element of $\mathcal{T}(\cdot)$ is defined in Eq. (2), and the evolutionary dynamics between different demographic groups are only coupled via the model $\boldsymbol{\theta}_t$.

In cases where user retention or churn rates are influenced by population dynamics, the function $\kappa^i(\cdot)$ is used to measure the model's performance on the currently active population. Here, the birth rate $\beta(\cdot)$ and the survival rate $\sigma(\cdot)$ denote the proportions of incoming and retained users at each respective time step. More precisely, the number of incoming users is dependent on the current model performance (e.g. a high-performing model on a demographic group attracts new users from this group), and the amount of retaining users depends on how the model performance on currently active users. For performance evaluation of the model, we maintain a small holdout set of accessible users, all of whose user features are observable. From this set, data from active users can be sampled at every time step, based on the population density λ_t^i. Moreover, when model performance is evaluated via a reward (or alternately, a loss) function, we hypothesize that the birth and survival rates correspond proportionally (or inversely) to the model performance $\kappa^i(\cdot)$. This assumption ensures that an improved model performance leads to an increase in population density. We denote this difference equation as the **surrogate retention system**.

Evaluation of model parameters through distributionally robust optimization. The surrogate retention system, as defined by Eq. (3), leads to a low-dimensional state representation, consisting only of the population densities across all demographic groups. However, its simulation requires random sampling of training data from the holdout set, based on the present population density. This is inconsistent with the population retention system, where the training data are provided from active users. To resolve this discrepancy, we consider the formulation of distributionally robust optimization, which considers the λ_t^i proportion of users who received optimal model performance.

To begin with, let $d_{\mathcal{X}^2}(\mathcal{M}||\mathcal{Q}) = \int(\frac{d\mathcal{M}}{d\mathcal{Q}} - 1)^2 d\mathcal{Q})$ denote the $\mathcal{X}^2$-divergence between two probability distributions $\mathcal{M}$ and $\mathcal{Q}$, $\mathcal{B}(\mathcal{M}, r) = \{\mathcal{Q} : d_{\mathcal{X}^2}(\mathcal{M}||\mathcal{Q}) \leq r\}$ denote the chi-squared ball around a probability distribution $\mathcal{M}$ of radius r. Let $\mathcal{M}^i$ be the feature distribution of users from the i$^{\text{th}}$ demographic group, we consider the performance measure $\kappa^i(\cdot)$ as the worst-case distributional loss over all r-radius balls around $\mathcal{M}^i$ defined as follows,

$$\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) = \sup_{\mathcal{Q}\in\mathcal{B}(\mathcal{M}^i, r_t^i)} \mathbb{E}_{\mathbf{z}\sim\mathcal{Q}} \Phi(\boldsymbol{\theta}_t, \mathbf{z}),$$
$$r_t^i = (1/\lambda_t^i - 1)^2. \quad (4)$$

Clearly, as the number density λ_t^i approaches 1, r_t^i decays to 0, and $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ is equivalent to population risk. For small λ_t^i, the radius $r_t^i \to \infty$ and this leads to a large loss value. In general, computing the worst-case distributional loss over a set of distributions is a challenging task. Fortunately, the maximization problem in Eq. (4) can be reformulated into its dual form (Duchi et al., 2019). More specifically, if $\Phi(\cdot)$ is upper semi-continuous for any $\boldsymbol{\theta}$, then for $r_t^i \geq 0$ and any $\boldsymbol{\theta}$, the following holds true:

$$\sup_{\mathcal{Q}\in\mathcal{B}(\mathcal{M}^i, r_t^i)} \mathbb{E}_{\mathbf{z}\sim\mathcal{Q}} \Phi(\boldsymbol{\theta}_t, \mathbf{z})$$
$$= \inf_{\eta\in\mathbb{R}} \Big(C(\lambda_t^i) \cdot \big(\mathbb{E}_{\mathcal{M}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{z}) - \eta]_+^2\big]\big)^{\frac{1}{2}} + \eta \Big),$$
$$\text{where } C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}}, \quad (5)$$

where $[x]_+ = x$ if $x \geq 0$ and 0 otherwise. At time step t, given $\boldsymbol{\theta}_t$ and λ_t^i, the worst-case distributional loss is computed by averaging the sample losses that are higher than the optimal $\eta^*(\lambda_t^i, \boldsymbol{\theta}_t)$, where $\eta^*(\lambda_t^i, \boldsymbol{\theta}_t)$ attains the infimum.

Rather than calculating the worst-case distributional loss, we consider $\Phi(\cdot)$ as a reward function. In this setting, the worst-case distributional loss corresponds to the computation of the maximal distributional reward, as defined within the chi-square ball of radius λ_t^i., and the active users are the ones who received a reward equal to or greater than $\eta^*(\lambda_t^i, \boldsymbol{\theta}_t)$. This computation can be performed efficiently using Eq. (5).

Overall, the surrogate retention system for the i$^{\text{th}}$ demographic population is defined as follows,

$$\begin{gathered}\lambda_{t+1}^i = (1-\lambda_t^i)\beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)) + \lambda_t^i \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)),\\ \text{where } \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) = \sup_{\mathcal{Q}\in\mathcal{B}(\mathcal{M}^i, r_t^i)} \mathbb{E}_{\mathbf{z}\sim\mathcal{Q}}\Phi(\boldsymbol{\theta}_t, \mathbf{z}),\\ r_t^i = (1/\lambda_t^i - 1)^2,\end{gathered}$$

where the worst-case distributional loss $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ can be efficiently computed via Eq. (5).

Here we describe two remarks about the proposed surrogate retention system.

Remark 4.1. The surrogate retention system, when implemented with the worst-case distributional loss formulation, yields a low-dimensional state representation, with the state consisting of the population densities of all demographics. Furthermore, given a model $\boldsymbol{\theta}_t$ and population density $\boldsymbol{\lambda}_t$, the system represents a deterministic difference equation where active users are generated as per Eq.(5). The system is differentiable with respect to the state $\boldsymbol{\lambda}_t$, allowing for efficient computation of the optimal model parameters, as detailed in Sec.4.2.

Remark 4.2. Model-based reinforcement learning (RL) techniques (Sutton, 1991; 1990) offer notable advantages in terms of data efficiency and interpretability. More specifically, model-based RL methods can learn from fewer interactions because they can simulate experiences using their model of the environment, addressing a key challenge faced by model-free RL methods such as policy gradient-based approaches (Sutton et al., 1999). Moreover, model-based approaches often provide more interpretability than model-free methods, since the model can offer insights into the agent's understanding of the environment dynamics. Despite these benefits, scaling model-based RL to high-dimensional problems introduces several challenges, due to the complexity and variability of these types of environments (Schrittwieser et al., 2020). In this study, we propose a novel approach that considers population density as a statistical average, effectively capturing underlying evolutions. By adopting this perspective, we simplify the problem to a low-dimensional surrogate retention system. The simulations conducted in this work can serve as a valuable foundation for the development of model-free RL methods.

4.2. Optimal Control Formulation for Asymptotic Fairness

We denote $\Psi(\cdot, \mathbf{1})$ as the binary cross-entropy loss between population densities and a fixed vector of $\mathbf{1}$ at a specific time step and $\mathbf{1}$. In this case, minimizing $\Psi(\cdot, \mathbf{1})$ is equivalent to maximizing the population densities at the terminal time step. Consequently, the objective of realizing asymptotic fairness can be formulated as follows:

$$\min_{\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}} \Psi(\boldsymbol{\lambda}_T, \mathbf{1}) \ \text{ s.t.} \boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t), \ \text{ given } \boldsymbol{\lambda}_0, \quad (6)$$

where $\mathcal{T}(\cdot)$ is the surrogate retention system defined in Eq. (3). This is a special case of a class of general optimal control problems for discrete dynamical systems, in which we consider the control variables as the model parameters at all time steps. From this optimal control perspective, asymptotic fairness can be achieved by solving for a set of controls such that Eq. (6) is satisfied.

This is known as closed-loop control in which the optimal control $\boldsymbol{\theta}_t^*(\boldsymbol{\lambda}_t)$ has explicit dependence on the state. The optimal control solution can be solved via dynamical programming principle (Bellman, 1952), which involves solving a Hamilton Jacobi Bellman partial differential equation. For an efficient model generation, PMP (Pontryagin, 1987) converts dynamical programming into two difference equations and a maximization condition. Instead of computing the closed-loop control $\boldsymbol{\theta}_t^*(\boldsymbol{\lambda}_t)$, the PMP provides a necessary condition for the optimality with fixed control. To begin with, we define the Hamiltonian as

$$H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t) := \mathbf{p}_{t+1}^T \cdot \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) - \mathcal{L}(\boldsymbol{\theta}_t, \boldsymbol{\lambda}_t),$$

where $\mathcal{L}(\boldsymbol{\theta}_t, \lambda_t^i)$ is a running loss at time t. We consider all running losses as 0 since asymptotic fairness is defined at the terminal state. The PMP consists of a two-point boundary value problem,

$$\boldsymbol{\lambda}_{t+1}^* = \nabla_p H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t^*), \quad \boldsymbol{\lambda}_0 \text{ given}, \quad (7)$$

$$\mathbf{p}_t^* = \nabla_\lambda H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_{t+1}^*, \boldsymbol{\theta}_t^*), \quad \mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_T}, \quad (8)$$

plus a maximum condition of the Hamiltonian.

$$H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t^*) \geq H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t), \ \forall \ \boldsymbol{\theta}_t \text{ and } t. \quad (9)$$

We consider the method of successive approximation (Chen et al., 2022) to solve for the control solution. Given a initial condition $\boldsymbol{\lambda}_0$, Eq.(7) corresponds to the surrogate retention system. We then set a terminal condition, $\mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_T}$. During backpropagation of the adjoint state, the current state $\boldsymbol{\lambda}_t$ and adjoint state $\mathbf{p}_{t+1}$ remain fixed, allowing for

Algorithm 1 Method of Successive Approximation.

```
input $\boldsymbol{\lambda}_0$, learning rate lr, maxItr, InnerItr
output models $\{\boldsymbol{\theta}_t\}_{t=0}^{T}$
  for $m = 1$ to maxItr do
    for $t = 0$ to T-1 do
      // Forward propagation (Eq. (7)).
      $\boldsymbol{\lambda}^*_{t+1} = \nabla_p H(t, \boldsymbol{\lambda}^{i,*}_t, \mathbf{p}^*_t, \boldsymbol{\theta}^*_t)$,
    end for
    // Set terminal condition.
    $\mathbf{p}^*_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_T}$,
    for $t = T-1$ to 0 do
      for $\tau = 0$ to InnerItr do
        // Compute Hamiltonian with $\mathbf{p}_{t+1}$ and $\boldsymbol{\lambda}_t$.
        $H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t) = \mathbf{p}^T_{t+1} \cdot \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t)$,
        // Max Hamiltonian (Eq. (9)).
        $\boldsymbol{\theta}_t = \boldsymbol{\theta}_t + \mathrm{lr} \cdot \nabla_{\boldsymbol{\lambda}} H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t)$,
      end for
      // Backward propagation (Eq. (8)).
      $\mathbf{p}^*_t = \nabla_{\boldsymbol{\lambda}} H(t, \boldsymbol{\lambda}^{i,*}_t, \mathbf{p}^*_{t+1}, \boldsymbol{\theta}^*_t)$,
    end for
  end for
```

the maximization of the t^{th} Hamiltonian for $InnerItr$ iterations. This strategy permits multiple updates to the model $\boldsymbol{\theta}_t$ within a single forward-backward propagation. Once a locally optimal model is achieved, we continue the backpropagation of the adjoint state $\mathbf{p}_{t+1}$ to $\mathbf{p}_t$ and optimize the model $\boldsymbol{\theta}_{t-1}$. In this configuration, executing the Hamiltonian dynamics n times can be decomposed into $maxItr$ complete iterations and $InnerItr$ local updates. The algorithm is depicted in Alg.1, with further implementation details provided in Appendix B.

5. Numerical Experiments

In this section, we describe two simulation environments that function as the population retention system, as defined in Eq. (5.1). Subsequently, we empirically validate the proposed optimal control solution utilizing a synthetic dataset in Section 5.2, and two realistic datasets frequently employed in fairness studies, as discussed in Section 5.3.

5.1. A Generic Platform for Fairness in Non-Stationary Environment

In this work, we construct two distinct population retention systems. The first configuration is based on a Markov decision process, where the model's prediction accuracy at each time step determines user participation in the next time step. Additionally, the feature vector $\mathbf{z}^{i,n}$ describes the probabilities of user churn following a correct prediction and user retention subsequent to an incorrect prediction. This creates a more realistic simulation, as the decision to retain or churn is typically not solely dependent on the accuracy of the model's prediction. The second configuration describes user retention as conditioned on a consequence of the historical accuracy of model predictions for a specific user. For instance, a user might churn if the model has produced a set number of incorrect predictions (e.g., 3 accumulated wrong predictions) for that specific user in the past. In both configurations, we assume that the number of new users joining at each time step is positively correlated with the performance of the model. For instance, if the model demonstrates high accuracy across all active users at time t, it attracts a larger number of new users at $t+1$. This enables an increase in population density as the performance of the model improves. For each simulation run, a set of active users is randomly sampled according to a given $\boldsymbol{\lambda}_0$ as initialization. In this case, simulation results in the evaluation phase are not expected to align with the training simulations, necessitating the models to exhibit robust generalization capabilities when different users are considered active during the initialization of the simulation. We denote $\mathcal{P}_1$ and $\mathcal{P}_2$ as the population retention systems in the Markovian and non-Markovian settings respectively.

For evaluation, we discuss a variety of existing baseline methods to compare against the proposed optimal control solution. We define $\mathcal{S}(\boldsymbol{\theta}_t, \mathbf{z}^{i,n}) = \Phi(\boldsymbol{\theta}_t, \mathbf{z}^{i,n})$ if $[\mathcal{A}^i_t]_n = 1$, and 0 otherwise.

- Empirical risk minimization (**ERM**) optimizes an average loss of all observable data,

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \frac{1}{\sum_{i=1}^{K}\sum_{n=1}^{N}[\mathcal{A}^i_t]_n} \sum_{i=1}^{K}\sum_{n=1}^{N} \mathcal{S}(\boldsymbol{\theta}_t, \mathbf{z}^{i,n}).$$

- Minimax optimization (**Minimax**) optimizes the worst-case loss among all groups (Diana et al., 2021),

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \max_{i=[1,\dots,K]} \frac{1}{\sum_{n=1}^{N}[\mathcal{A}^i_t]_n} \sum_{n=1}^{N} \mathcal{S}(\boldsymbol{\theta}_t, \mathbf{z}^{i,n}).$$

- Distributionally robust optimization (**DRO**) sets a lower bound on the population density of active users and considers a proportion of data that produces the worst-case loss (Hashimoto et al., 2018).

- A greedy control (**Greedy**) is equivalent to one-step planning of the optimal control method,

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \Psi(\boldsymbol{\lambda}_{t+1}, \mathbf{1})$$
$$\text{s.t.}\boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t), \quad \text{given } \boldsymbol{\lambda}_t.$$

The aforementioned baseline methods do not take into consideration the underlying dynamics, each of those achieves the specific optimum defined at each time step.

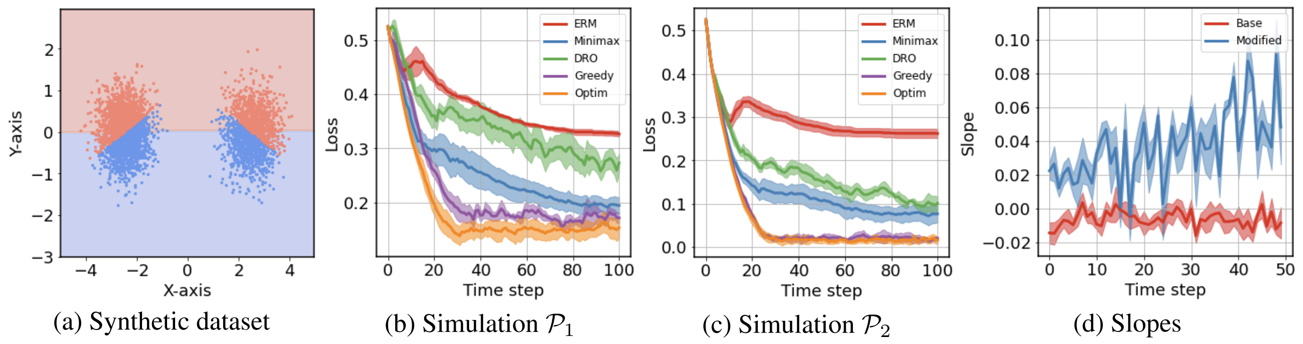


Figure 1. (a) illustrates the synthetic dataset with a fair model decision boundary. (b) and (c) plot the binary cross-entropy losses of population densities resulting from ERM, Minimax, DRO, Greedy, and the proposed Optim methods in simulation environments $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively. (d) plots model slopes resulting from the Optim method in the base and modified environments.

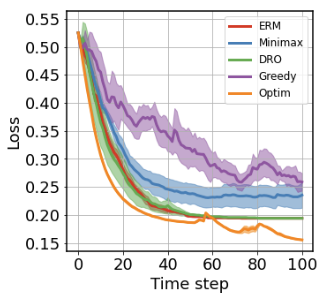
5.2. Modeling with Synthetic Dataset

In this study, we employ both population retention systems, $\mathcal{P}_1$ and $\mathcal{P}_2$, with a synthetic binary classification dataset. As depicted in Fig. 1 (a), the synthetic dataset is composed of two Gaussian blobs, each centered at disparate locations, to formulate the feature distributions of two demographic groups. The blobs located on the left and right are denoted as the majority and minority demographic groups, respectively, with respective initial population densities of 0.7 and 0.3. All experiments are reiterated with five random seeds, and the binary cross-entropy loss $\Psi(\boldsymbol{\lambda}_t, \mathbf{1})$ is used to quantify the quality of each state $\boldsymbol{\lambda}_t$. For instance, as $\boldsymbol{\lambda}_t$ increases towards $\mathbf{1}$, this loss diminishes to zero.

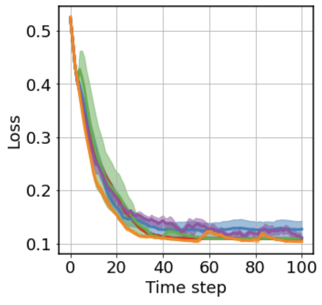
In the proposed optimal control methodology, we express both the survival rate and birth rate functions of the surrogate retention system as a weighted sum of polynomials. Subsequently, we impose constraints on the weighting parameters to ensure a monotonically increasing behavior in both functions, thereby aligning with the assumption underpinning the proposed surrogate retention system. During each simulation run, the optimal control method learns a surrogate retention system by optimizing the weight parameters of both the birth and survival functions. It then generates a sequence of models utilizing the method of successive approximation, as detailed in Algorithm 1.

Figure 1 (b) and (c) demonstrate the results simulated from $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. As observed, in both simulation environments, ERM results in comparably high losses at all time steps due to the low population densities in the minority group. For instance, at $t = 100$, ERM results in $\lambda_T^1 = 1$ and $\lambda_T^2 = 0.53$ with a loss of 0.32. For DRO, we utilize the initial minority population density to compute the DRO loss. This approach yields conservative models at all time steps, leading to low densities accompanied by high losses. Minimax is capable of mitigating representation disparity by minimizing the worst-case loss among both groups, thereby resulting in a more balanced terminal state with $\lambda_T^1 = 0.9$ and $\lambda_T^2 = 0.76$. It is noteworthy that the aforementioned methods do not consider the underlying dynamics. Conversely, the optimal and greedy control methods implement T-step and 1-step planning respectively, leveraging the estimated underlying dynamics. When these dynamics significantly impact user retention, the T-step planning demonstrates superior performance compared to the single-step planning, as depicted in Fig.1 (b) under simulation environment $\mathcal{P}_1$. However, simulation environment $\mathcal{P}_2$ stipulates user churn when 3 wrong predictions have accrued, this leads to a system with smoother population dynamics, where T-step planning performs comparably to the single-step planning method, as shown in Fig.1 (c).

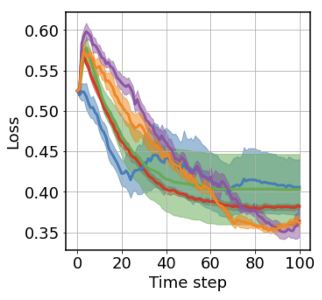
The advantage of trajectory planning that acknowledges the underlying dynamics is explored herein. The idea is to interpret the evolution of the population retention system at certain time steps and observe corresponding adjustments made in the optimal control solution. The optimal control method makes performance tradeoffs from the majority group to balance the population densities of both groups at the terminal step. We manually introduce a substantial quantity of users into the minority demographic group at $t = 50$. Consequently, it is expected that the optimal control method would make fewer tradeoffs and adjust its decision boundaries accordingly at earlier time steps (e.g., $t < 50$). Referring to Fig.1 (a), we consider a linear classifier where a positive (resp. negative) slope indicates a model favoring the majority demographic (resp. minority) group. Figure1 (d) illustrates the slopes of the model decision boundary at $t \in [0, 50]$. As observed, the introduction of additional users to the minority group at $t = 50$ enables the optimal control solution to make less performance tradeoff against the majority group due to the increased population density at a later time step. This adjustment cannot be accomplished with greedy control-based methods.



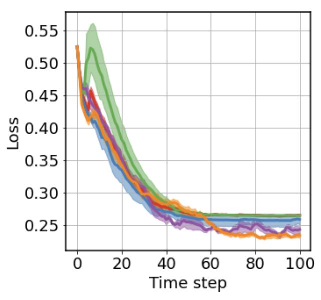
(a) Adult in $\mathcal{P}_1$



(b) Adult in $\mathcal{P}_2$



(c) COMPAS in $\mathcal{P}_1$



(d) COMPAS in $\mathcal{P}_2$

Figure 2. (a) and (b) depict the trajectories of losses $\Psi(\boldsymbol{\lambda}_t, \mathbf{1})$ resulting from simulation environments $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively, utilizing the Adult dataset. Similarly, the outcomes from $\mathcal{P}_1$ and $\mathcal{P}_2$ using the COMPAS dataset are illustrated in figures (c) and (d) respectively.

5.3. Modeling with Adult Income and COMPAS Recidivism Racial Bias.

We explore two real-world datasets: Adult Income (Adult) (Kohavi et al., 1996) and COMPAS (Barenstein, 2019). The Adult dataset describes an individual's annual income, influenced by factors such as gender, race, age, and education level, while COMPAS is a widely-used commercial tool that assists judges and parole officers in assessing the likelihood of a criminal defendant reoffending. In both datasets, gender attributes are used to differentiate demographic groups. The population retention system, as defined in Eq. (1), is applied to these static datasets to simulate population dynamics. A total of $N = 3000$ samples are randomly selected from each demographic group, setting initial population densities at $\lambda_0^1 = 0.7$ and $\lambda_0^2 = 0.5$ for the majority and minority groups respectively. Fig. 2 (a) and (b) illustrate the trajectories of the binary cross-entropy loss in the environments $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. The optimal control solution (Optim) surpasses other baselines, achieving terminal states with $\lambda_T^1 = 0.93$ and $\lambda_T^2 = 0.78$ in $\mathcal{P}_1$, and $\lambda_T^1 = 1$ and $\lambda_T^2 = 0.81$ in $\mathcal{P}_2$. Fig. 2 (c) and (d) present the results obtained from the COMPAS dataset. In both instances, due to the relatively smooth dynamics inherent in the COMPAS dataset, single-step planning performs on par with the optimal control solution.

6. Limitations and Future Works

In this section, we provide a discussion of the limitations of the proposed framework, thereby highlighting potential areas for improvement and development.

Constraints on intermediate population densities: In the context of fairness, asymptotic behavior plays a crucial role in defining long-term population densities. However, exclusive dependence on asymptotic fairness might fall short in managing representation disparities during intermediate time steps, potentially resulting in unsatisfying user experiences. To mitigate this limitation, our future work aims to incorporate an additional running loss into the framework.

Surrogate retention system: In real-world scenarios, user churn behavior is influenced by a multitude of factors, including financial background and health conditions. These factors are not static but vary over time, which requires the adoption of a dynamic modeling approach. Existing literature in the field of evolutionary population dynamics often employs a canonical equation, known as Darwinian dynamics, to describe the evolution of individuals' traits (Cushing, 2019). However, to achieve a more accurate representation of population dynamics, it is essential to incorporate individual characteristics into the modeling framework.

Computational efficiency: The current framework addresses the control problem using the PMP, which involves solving an optimization problem. However, this approach becomes inefficient when the algorithm is deployed in real-world services that require frequent updates. To overcome this limitation, we have developed a closed-form solution in Appendix A for achieving asymptotic fairness through linear approximation. However, our preliminary findings indicate that this approach leads to underperforming models. Consequently, our next objective is to derive an accurate closed-form solution that maintains the desired fairness objectives. By refining the closed-form solution, we aim to develop a more efficient and effective algorithm that can be seamlessly integrated into services requiring frequent updates, while still preserving the desired level of performance.

7. Conclusion

As the implementation of machine learning systems in real-world applications continues to proliferate, the examination of long-term fairness issues becomes increasingly critical. This stems from the fact that the outcomes produced by these systems have profound implications, making the topic of fairness a major concern. In the present study, we have devised an analytical framework aimed at exploring fairness in dynamic settings. Such settings are characterized by the fact that the level of user engagement is directly influenced by the performance of the model in use. This perspective aligns with real-world scenarios where users might decide to continue or discontinue the use of a system based on its perceived fairness or bias.

To gauge fairness in this fluctuating landscape, we introduced the concept of asymptotic fairness. Asymptotic fairness, in this context, allows for the appraisal of fairness over time, taking into account the evolving relationship between user participation and model performance. The concept was then incorporated into an optimal control problem formulation, an approach that enables us to deal with dynamic systems efficiently and predict the best control actions over time. As part of our methodology, we developed a surrogate retention system, serving as a proxy for the real-world environment. This system plays a vital role in estimating how users' engagement might change over time based on the performance of the model, facilitating a clearer understanding of the dynamics at play. Our proposed optimal control solution, built upon this framework, has been validated through a series of simulations, demonstrating its effectiveness in managing fairness in dynamic settings.

Looking ahead, our future research will delve into the design and implementation of advanced surrogate retention systems. These systems will be engineered to handle complex environments, possibly interacting with real-world human users. Such complexity will serve to enhance the reliability and validity of our investigations. Furthermore, we aspire to gain deeper theoretical insights into the functioning principle of our proposed optimal control solution. A focal point of this exploration will be the delineation of our approach from methods reliant on greedy control strategies. Specifically, we aim to elucidate the nuances of how the underlying dynamics of the system influence the derivation of optimal control solutions, particularly in relation to fairness within non-stationary environments.

Acknowledgement Zhuotong Chen and Zheng Zhang are supported by NSF Grant # 2107321 and DOE Grant # DE-SC0021323. Qianxiao Li is supported by the National Research Foundation, Singapore, under the NRF fellowship (project No. NRF-NRFF13-2021-0005).

References

Barenstein, M. Propublica's compas data revisited. *arXiv preprint arXiv:1906.04711*, 2019.

Bellman, R. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America*, 38(8):716, 1952.

Chen, Z., Li, Q., and Zhang, Z. Towards robust neural networks via close-loop control. In *International Conference on Learning Representations*, 2021.

Chen, Z., Li, Q., and Zhang, Z. Self-healing robust neural networks via closed-loop control. *Journal of Machine Learning Research*, 23(319):1–54, 2022. URL `http://jmlr.org/papers/v23/22-0529.html`.

Cushing, J. Difference equations as models of evolutionary population dynamics. *Journal of Biological Dynamics*, 13(1):103–127, 2019.

Diana, E., Gill, W., Kearns, M., Kenthapadi, K., and Roth, A. Minimax group fairness: Algorithms and experiments. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 66–76, 2021.

Duchi, J. C., Hashimoto, T., and Namkoong, H. Distributionally robust losses against mixture covariate shifts. *Under review*, 2, 2019.

E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

E, W., Han, J., and Li, Q. A mean-field optimal control formulation of deep learning. 6(1):10. ISSN 2522-0144.

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 259–268, 2015.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

Haber, E. and Ruthotto, L. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017.

Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

Harwell, D. Amazon's alexa and google home show accent bias, with chinese and spanish hardest to understand. 2018.

Hashimoto, T., Srivastava, M., Namkoong, H., and Liang, P. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pp. 1929–1938. PMLR, 2018.

Kirk, D. E. *Optimal control theory: an introduction*. Springer, 1970.

Kohavi, R. et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pp. 202–207, 1996.

Li, Q. and Hao, S. An optimal control approach to deep learning and applications to discrete-weight neural networks. In *International Conference on Machine Learning*, pp. 2985–2994. PMLR, 2018.

Li, Q., Chen, L., Tai, C., and E, W. Maximum principle based algorithms for deep learning. *Journal of Machine Learning Research*, 18(165):1–29, 2018. URL `http://jmlr.org/papers/v18/17-653.html`.

Liu, G.-H. and Theodorou, E. A. Deep learning theory review: An optimal control and dynamical systems perspective. *arXiv preprint arXiv:1908.10920*, 2019.

Liu, G.-H., Chen, T., and Theodorou, E. A. Differential dynamic programming neural optimizer. *arXiv preprint arXiv:2002.08809*, 2020a.

Liu, G.-H., Chen, T., and Theodorou, E. A. A differential game theoretic neural optimizer for training residual networks. *arXiv preprint arXiv:2007.08880*, 2020b.

Liu, L. T., Dean, S., Rolf, E., Simchowitz, M., and Hardt, M. Delayed impact of fair machine learning. In *International Conference on Machine Learning*, pp. 3150–3158. PMLR, 2018.

Pontryagin, L. S. *Mathematical theory of optimal processes*. CRC press, 1987.

Schlimmer, J. C. and Granger, R. H. Incremental learning from noisy data. *Machine learning*, 1(3):317–354, 1986.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pp. 216–224. Elsevier, 1990.

Sutton, R. S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Vincent, T. L. and Brown, J. S. *Evolutionary game theory, natural selection, and Darwinian dynamics*. Cambridge University Press, 2005.

Widmer, G. and Kubat, M. Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23 (1):69–101, 1996.

Zhang, X., Khaliligarekani, M., Tekin, C., et al. Group retention when using machine learning in sequential decision making: the interplay between user dynamics and fairness. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhang, X., Tu, R., Liu, Y., Liu, M., Kjellstrom, H., Zhang, K., and Zhang, C. How do fair decisions fare in long-term qualification? *Advances in Neural Information Processing Systems*, 33:18457–18469, 2020.

A. Closed-Form Solution of the Optimal Control in Linear Case

In this section, we derive a closed-form solution for the proposed optimal control framework within a simplified linear scenario. Let us revisit the surrogate retention system:

$$\boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) = \begin{bmatrix} \beta(\kappa^1(\lambda_t^1, \boldsymbol{\theta}_t)) \\ \beta(\kappa^2(\lambda_t^2, \boldsymbol{\theta}_t)) \\ \vdots \\ \beta(\kappa^K(\lambda_t^K, \boldsymbol{\theta}_t)) \end{bmatrix} \odot (\mathbf{1} - \boldsymbol{\lambda}_t) + \begin{bmatrix} \sigma(\kappa^1(\lambda_t^1, \boldsymbol{\theta}_t)) \\ \sigma(\kappa^2(\lambda_t^2, \boldsymbol{\theta}_t)) \\ \vdots \\ \sigma(\kappa^K(\lambda_t^K, \boldsymbol{\theta}_t)) \end{bmatrix} \odot \boldsymbol{\lambda}_t.$$

To begin with, we linearize the above nonlinear surrogate retention system.

Lemma A.1. *The surrogate retention system, defined in Eq.* (3)*, can be linearized at (*$\boldsymbol{\lambda}_t$*,* $\boldsymbol{\theta}_t$*) in the following expression,*

$$\boldsymbol{\lambda}_{t+1} = \mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t, \tag{10}$$

where $\mathbf{A} \in \mathbb{R}^{K,K}$*,* $\mathbf{B} \in \mathbb{R}^{K,d}$*, the control* $\boldsymbol{\theta}_t$ *is shaped into a* d*-dimensional vector. The elements of* $\mathbf{A}$ *and* $\mathbf{B}$ *can be represented as the follows,*

$$\begin{cases} \mathbf{A}_{i,j} = (1-\lambda_t^j)\frac{\partial\beta(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial\lambda_t^j} + \lambda_t^i\frac{\partial\sigma(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial\lambda_t^j} + \beta(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t)) - \sigma(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t)) & \textit{if } i=j \\ \mathbf{A}_{i,j} = 0, & \textit{otherwise} \end{cases}$$

$$\mathbf{B}_{i,j} = (1-\lambda_t^i)\frac{\partial\beta(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial[\boldsymbol{\theta}_t]_j} + \lambda_t\frac{\partial\sigma(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial[\boldsymbol{\theta}_t]_j},$$

where $[\boldsymbol{\theta}_t]_k$ *indicates the* $\mathrm{k^{th}}$ *element of* $\boldsymbol{\theta}_t$*.*

Given the linearized approximation of the surrogate retention model, we proceed to formulate a closed-form solution for the optimal control policy. Our goal, as prescribed by Definition 3.1, is to ensure asymptotic fairness, which we interpret as maximizing the terminal-time population densities across all demographic groups. This objective is best expressed as an optimal control problem in the Mayer form, wherein the objective function is defined at the terminal time step only. For analytical tractability, our first step involves transforming this Mayer form objective function into its equivalent Lagrangian form, thereby allowing the objective function to be defined at each temporal instance.

$$\max\|\boldsymbol{\lambda}_T\|_2^2 \equiv \max\|\boldsymbol{\lambda}_0\|_2^2 + \sum_{t=0}^{T-1}\|\boldsymbol{\lambda}_{t+1}\|_2^2 - \|\boldsymbol{\lambda}_t\|_2^2. \tag{11}$$

The following lemma presents the closed-form solution of the optimal control in the linear case.

Lemma A.2. *For the sum of the one-step costs over a finite horizon*

$$J(\boldsymbol{\lambda}_0, \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}) = \|\boldsymbol{\lambda}_0\|_2^2 + \sum_{t=0}^{T-1}\|\boldsymbol{\lambda}_{t+1}\|_2^2 - \|\boldsymbol{\lambda}_t\|_2^2, \text{ s.t. } Eq. (10),$$

the optimal cost-to-go function, parametrized as $V(\boldsymbol{\lambda}_t) = \boldsymbol{\lambda}_t^T\mathbf{P}_t\boldsymbol{\lambda}_t$*, is the solution of the following Riccati equation*

$$\begin{aligned} \mathbf{P}_t = {} & \frac{1}{2}(\mathbf{A}^T\mathbf{A} - \mathbf{I}) + \mathbf{A}^T\mathbf{P}_{t+1}\mathbf{A} \\ & - \frac{1}{2}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})^T(\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A}), \end{aligned} \tag{12}$$

with the optimal control solution

$$\boldsymbol{\theta}_t^* = -(\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})\boldsymbol{\lambda}_t, \tag{13}$$

where $\mathbf{A}$ *and* $\mathbf{B}$ *are defined in Lemma A.1.*

Proof. Given the optimal control problem defined in Laguarange form in Eq. (11), its associated Bellman's equation can be formulated as follows,

$$V(\boldsymbol{\lambda}_t) = \frac{1}{2}\|\boldsymbol{\lambda}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}_t\|_2^2 + V(\boldsymbol{\lambda}_{t+1}),$$

where $V(\cdot)$ is the optimal cost-to-go function. We parametrize $V(\cdot)$ as $\boldsymbol{\lambda}_t^T \mathbf{P}_t \boldsymbol{\lambda}_t$,

$$\begin{aligned} V(\boldsymbol{\lambda}_t) &= \frac{1}{2}\|\boldsymbol{\lambda}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}_t\|_2^2 + \boldsymbol{\lambda}_{t+1}^T \mathbf{P}_{t+1}\boldsymbol{\lambda}_{t+1}, \\ &= \frac{1}{2}\|\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}_t\|_2^2 + (\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t)^T \mathbf{P}_{t+1}(\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t). \end{aligned}$$

By taking the derivative of $V(\boldsymbol{\lambda}_t)$ w.r.t. $\boldsymbol{\theta}_t$,

$$\frac{\partial V(\boldsymbol{\lambda}_t)}{\partial \boldsymbol{\theta}_t} = (\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})\boldsymbol{\theta}_t + (\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})\boldsymbol{\lambda}_t.$$

Setting the above to $\mathbf{0}$ results in the optimal control $\boldsymbol{\theta}_t^*$,

$$\boldsymbol{\theta}_t^* = -(\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})\boldsymbol{\lambda}_t.$$

Recall the Bellman's equation, and $V(\boldsymbol{\lambda}_t) = \boldsymbol{\lambda}_t^T \mathbf{P}_t \boldsymbol{\lambda}_t$,

$$\begin{aligned} \boldsymbol{\lambda}_t^T \mathbf{P}_t \boldsymbol{\lambda}_t &= \min_{\boldsymbol{\theta}} \frac{1}{2}\|\boldsymbol{\lambda}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}_t\|_2^2 + \boldsymbol{\lambda}_{t+1}^T \mathbf{P}_{t+1}\boldsymbol{\lambda}_{t+1}, \\ &= \min_{\boldsymbol{\theta}} \frac{1}{2}\|\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}_t\|_2^2 + (\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t)^T \mathbf{P}_{t+1}(\mathbf{A}\boldsymbol{\lambda}_t + \mathbf{B}\boldsymbol{\theta}_t)^T, \\ &= \boldsymbol{\lambda}_t^T(\frac{1}{2}\mathbf{A}^T\mathbf{A} + \mathbf{A}^T\mathbf{P}_{t+1}\mathbf{A} - \frac{1}{2}\mathbf{I})\boldsymbol{\lambda}_t + \boldsymbol{\lambda}_t^T(\mathbf{A}^T\mathbf{B} + 2\mathbf{A}^T\mathbf{P}_{t+1}\mathbf{B})\boldsymbol{\theta}_t^*, \\ &\qquad + (\boldsymbol{\theta}_t^*)^T(\frac{1}{2}\mathbf{B}^T\mathbf{B} + \mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})\boldsymbol{\theta}_t^*. \end{aligned}$$

For the last term in the above, recall the optimal control solution $\boldsymbol{\theta}_t^*$ in Eq. 13),

$$\begin{aligned} &(\boldsymbol{\theta}_t^*)^T(\frac{1}{2}\mathbf{B}^T\mathbf{B} + \mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})\boldsymbol{\theta}_t^* \\ &= -\boldsymbol{\lambda}_t^T(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})^T(\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}(\frac{1}{2}\mathbf{B}^T\mathbf{B} + \mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})\boldsymbol{\theta}_t, \\ &= -\frac{1}{2}\boldsymbol{\lambda}_t^T(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})^T\boldsymbol{\theta}_t^*. \end{aligned}$$

Therefore,

$$\boldsymbol{\lambda}_t^T \mathbf{P} \boldsymbol{\lambda}_t = \boldsymbol{\lambda}_t^T(\frac{1}{2}\mathbf{A}^T\mathbf{A} + \mathbf{A}^T\mathbf{P}_{t+1}\mathbf{A} - \frac{1}{2}\mathbf{I})\boldsymbol{\lambda}_t - \frac{1}{2}\boldsymbol{\lambda}_t^T(\mathbf{A}^T\mathbf{B} + 2\mathbf{A}^T\mathbf{P}_{t+1}\mathbf{B})^T\boldsymbol{\theta}_t^*.$$

Recall the optimal control solution $\boldsymbol{\theta}_t^*$ in Eq. 13),

$$\begin{aligned} \mathbf{P}_t = &\frac{1}{2}(\mathbf{A}^T\mathbf{A} - \mathbf{I}) + \mathbf{A}^T\mathbf{P}_{t+1}\mathbf{A} \\ &- \frac{1}{2}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A})^T(\mathbf{B}^T\mathbf{B} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{A} + 2\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A}), \end{aligned}$$

□

B. Extensive Details On the Pontryagon's Maximum Principle

In this section, we provide details on solving Pontryagon's maximum principle (PMP) via the method of successive approximation. To begin with, we recall the definition of the Hamiltonian,

$$H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t) := \mathbf{p}_{t+1}^T \cdot \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) - \mathcal{L}(\boldsymbol{\theta}_t, \boldsymbol{\lambda}_t),$$

where $\mathcal{T}(\cdot)$ represents the surrogate retention system, defined in Eq. (3), $\mathcal{L}(\boldsymbol{\theta}_t, \lambda_t^i)$ is a running loss at time t. We consider all running losses as 0 since asymptotic fairness is defined at the terminal state. The PMP consists of a two-point boundary value problem

$$\boldsymbol{\lambda}_{t+1}^* = \nabla_p H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t^*), \quad \boldsymbol{\lambda}_0 \text{ given}, \tag{14}$$

$$\mathbf{p}_t^* = \nabla_\lambda H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_{t+1}^*, \boldsymbol{\theta}_t^*), \quad \mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_T}, \tag{15}$$

and a maximization condition on the Hamiltonian,

$$H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t^*) \geq H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t), \ \ \forall \, \boldsymbol{\theta}_t \text{ and } t. \tag{16}$$

Alg. 1 presents the method of successive approximation that solves the PMP iteratively. Given a initial condition $\boldsymbol{\lambda}_0$, the method of successive approximation simulates the surrogate retention system via Eq. (14). Notice that

$$\boldsymbol{\lambda}_{t+1}^* = \nabla_p H(t, \boldsymbol{\lambda}_t^{i,*}, \mathbf{p}_t^*, \boldsymbol{\theta}_t^*) = \mathcal{T}(\boldsymbol{\lambda}_t^*, \boldsymbol{\theta}_t),$$

which is equivalent to the forward propagation of the surrogate retention system.

Once we reach the terminal state $\boldsymbol{\lambda}_T$, the adjoint system defined in Eq. (15) is a difference equation that propagates the derivative of the terminal loss w.r.t. state $\boldsymbol{\lambda}_t$ at every time step t. In this work, we consider a terminal loss as the binary cross-entropy loss between the terminal state $\boldsymbol{\lambda}_T$ and a vector $\mathbf{1}$, in which case, minimizing this loss is equivalent to maximizing the population densities of all demographic groups at the terminal time step. Let $\Psi(\boldsymbol{\lambda}_T, \mathbf{1})$ denote the loss measured at the terminal time step, the adjoint state at each time step can be represented as

$$\frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_t} = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_T} \cdot \frac{\partial \boldsymbol{\lambda}_T}{\partial \boldsymbol{\lambda}_{T-1}} \cdots \frac{\partial \boldsymbol{\lambda}_{t+2}}{\partial \boldsymbol{\lambda}_{t+1}} \cdot \frac{\partial \boldsymbol{\lambda}_{t+1}}{\partial \boldsymbol{\lambda}_t}.$$

Recall the second boundary value problem defined in Eq. (15),

$$\mathbf{p}_t^* = \mathbf{p}_{t+1}^T \cdot \frac{\partial \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t)}{\partial \boldsymbol{\lambda}_t},$$

we replace the adjoint state $\mathbf{p}_t$ with the derivative $\frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_t}$,

$$\frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_t} = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_{t+1}}^T \cdot \frac{\partial \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t)}{\partial \boldsymbol{\lambda}_t} = \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_{t+1}}^T \cdot \frac{\partial \boldsymbol{\lambda}_{t+1}}{\partial \boldsymbol{\lambda}_t}.$$

Therefore, the adjoint system computes the derivative of the terminal loss w.r.t. the state at every time step, $\frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_t}$, for all t. We derive the derivative $\frac{\partial \boldsymbol{\lambda}_{t+1}}{\partial \boldsymbol{\lambda}_t}$ in Sec. B.1.

Once we obtain the state $\boldsymbol{\lambda}_t$ and adjoint state $\frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_t}$, we maximize the Hamiltonian w.r.t. the model parameters $\boldsymbol{\theta}_t$,

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta}} H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}),$$

this can be solved via any optimization method,

$$\boldsymbol{\theta}_t^{\text{next}} = \boldsymbol{\theta}_t^{\text{current}} + \text{lr} \cdot \frac{\partial \Psi(\boldsymbol{\lambda}_T, \mathbf{1})}{\partial \boldsymbol{\lambda}_{t+1}} \cdot \frac{\partial \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t^{\text{current}})}{\partial \boldsymbol{\theta}_t},$$

where $\boldsymbol{\theta}_t^{\text{current}}$ and $\boldsymbol{\theta}_t^{\text{next}}$ are the model parameters at time step t resulting from current and next gradient updates respectively.

B.1. Derivation for the Derivative Between Two Consecutive States

Here we derive the derivative $\frac{\partial \boldsymbol{\lambda}_{t+1}^i}{\partial \boldsymbol{\lambda}_t^i}$. Recall the proposed surrogate retention system,

$$\boldsymbol{\lambda}_{t+1} = \mathcal{T}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) = \begin{bmatrix} \beta(\kappa^1(\lambda_t^1, \boldsymbol{\theta}_t)) \\ \beta(\kappa^2(\lambda_t^2, \boldsymbol{\theta}_t)) \\ \vdots \\ \beta(\kappa^K(\lambda_t^K, \boldsymbol{\theta}_t)) \end{bmatrix} \odot (\mathbf{1} - \boldsymbol{\lambda}_t) + \begin{bmatrix} \sigma(\kappa^1(\lambda_t^1, \boldsymbol{\theta}_t)) \\ \sigma(\kappa^2(\lambda_t^2, \boldsymbol{\theta}_t)) \\ \vdots \\ \sigma(\kappa^K(\lambda_t^K, \boldsymbol{\theta}_t)) \end{bmatrix} \odot \boldsymbol{\lambda}_t,$$

where the performance measure $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ is defined as follows,

$$\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) = \sup_{\mathcal{Q} \in \mathcal{B}(\mathcal{M}^i, r_t^i)} \mathbb{E}_{\mathbf{z} \sim \mathcal{Q}} \Phi(\boldsymbol{\theta}_t, \mathbf{z}) = \inf_{\eta \in \mathbb{R}} \Big(C(\lambda_t^i) \cdot \left(\mathbb{E}_{\mathcal{M}^i} \left[[\Phi(\boldsymbol{\theta}_t, \mathbf{z}) - \eta]_+^2 \right] \right)^{\frac{1}{2}} + \eta \Big),$$
$$\text{where } \ C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}}.$$

Since the evolutions of all population densities are decoupled, for simplicity, we consider a one-dimensional difference equation that describes the evolution of the i$^{\text{th}}$ population density,

$$\lambda_{t+1}^i = \beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))(1 - \lambda_t^i) + \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))\lambda_t^i.$$

The derivative $\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i}$ can be computed as follows,

$$\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i} = (1 - \lambda_t^i) \frac{\partial \beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \lambda_t^i} + \lambda_t^i \frac{\partial \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \lambda_t^i} + \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)) - \beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)).$$

Recall the definition of $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$, let us define the optimum $\eta^*(\lambda_t^i, \boldsymbol{\theta}_t)$ as follows,

$$\eta^*(\lambda_t^i, \boldsymbol{\theta}_t) = \arg \inf_{\eta \in \mathbb{R}} \Big(C(\lambda_t^i) \cdot \left(\mathbb{E}_{\mathcal{M}^i} \left[[\Phi(\boldsymbol{\theta}_t, \mathbf{z}) - \eta]_+^2 \right] \right)^{\frac{1}{2}} + \eta \Big).$$

In practice, we estimate the population risk with N samples randomly drawn from the data distribution $\mathcal{M}^i$, we define the following function,

$$T(\lambda_t^i, \boldsymbol{\theta}_t, \mathbf{z}^{i,n}) := \begin{cases} (\Phi(\boldsymbol{\theta}_t, \mathbf{z}_i) - \eta^*(\lambda_t^i, \boldsymbol{\theta}_t))^2 & \text{if } \Phi(\boldsymbol{\theta}_t, \mathbf{z}_i) > \eta^*(\lambda_t^i, \boldsymbol{\theta}_t) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

The derivative of $T(\lambda_t^i, \boldsymbol{\theta}_t, \mathbf{z}^{i,n})$ w.r.t. λ_t^i can be computed as follows,

$$\frac{\partial T(\lambda_t^i, \boldsymbol{\theta}_t, \mathbf{z}^{i,n})}{\partial \lambda_t^i} = \begin{cases} -2\Big(\Phi(\boldsymbol{\theta}_t, \mathbf{z}_i) - \eta^*(\lambda_t^i, \boldsymbol{\theta}_t)\Big) \cdot \frac{\partial \eta^*(\lambda_t^i, \boldsymbol{\theta}_t)}{\lambda_t^i} & \text{if } \Phi(\boldsymbol{\theta}_t, \mathbf{z}_i) > \eta^*(\lambda_t^i, \boldsymbol{\theta}_t) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

The derivative terms $\frac{\partial \beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \lambda_t^i}$ and $\frac{\partial \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \lambda_t^i}$ can be decomposed as $\frac{\partial \beta(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}$ and $\frac{\partial \sigma(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t))}{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot$

$\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}$ respectively. The common term $\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}$ can be derived as follows,

$$\begin{aligned}
\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} &= \frac{\partial\Big(C(\lambda_t^i)\cdot\big(\mathbb{E}_{\mathbf{z}\sim\mathcal{M}^i}\big[[\Phi(\boldsymbol{\theta}_t,\mathbf{z})-\eta^*(\lambda_t^i,\boldsymbol{\theta}_t)]_+^2\big]\big)^{\frac{1}{2}}+\eta^*(\lambda_t^i,\boldsymbol{\theta}_t)\Big)}{\partial \lambda_t^i},\\
&= \frac{\partial\Big(C(\lambda_t^i)\cdot\big(\frac{1}{N}\sum_{n=1}^N T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}^{i,n}]\big)^{\frac{1}{2}}+\eta^*(\lambda_t^i,\boldsymbol{\theta}_t)\Big)}{\partial \lambda_t^i},\\
&= \frac{\partial \eta^*(\lambda_t^i,\boldsymbol{\theta}_t)}{\partial \lambda_t^i}+\frac{\partial\Big(C(\lambda_t^i)\cdot\big(\frac{1}{N}\sum_{n=1}^N T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}^{i,n}]\big)^{\frac{1}{2}}\Big)}{\partial \lambda_t^i},\\
&= \frac{\partial \eta^*(\lambda_t^i,\boldsymbol{\theta}_t)}{\partial \lambda_t^i}+\frac{\partial C(\lambda_t^i)}{\lambda_t^i}\cdot\big(\frac{1}{N}\sum_{n=1}^N T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}^{i,n})]\big)^{\frac{1}{2}}\\
&\quad + C(\lambda_t^i)\cdot\Big(\frac{1}{2}\frac{1}{N}\sum_{n=1}^N T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}_i)\Big)^{-\frac{1}{2}}\cdot\frac{1}{N}\sum_{n=1}^N\frac{\partial T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}^{i,n})}{\partial\lambda_t^i},
\end{aligned} \tag{19}$$

where $T(\lambda_t^i, \boldsymbol{\theta}_t, \mathbf{z}_i)$ and $\frac{\partial T(\lambda_t^i,\boldsymbol{\theta}_t,\mathbf{z}^{i,n})}{\partial\lambda_t^i}$ are defined in Eq. (17) and Eq. (18) respectively.

Therefore,

$$\begin{aligned}
&\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i}\\
&= \frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}\cdot\Big((1-\lambda_t^i)\frac{\partial\beta(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t)}+\lambda_t^i\frac{\partial\sigma(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))}{\partial\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t)}\Big)+\sigma(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t))-\beta(\kappa^i(\lambda_t^i,\boldsymbol{\theta}_t)),
\end{aligned}$$

where $\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}$ is defined in Eq. (19).

B.2. Implementation of the Optimal Control Method

```
# PMP implementation
def pmp(self, num_density_init, lr, max_iterations):
    # set up optimizer
    optimizer = torch.optim.SGD(self.agent.params.parameters(), lr=lr)
    # set up loss function
    cirterion = torch.nn.BCELoss()
    # reference label
    ref = torch.ones_like(num_density_init)
    for step in range(max_iterations):
        optimizer.zero_grad()
        # initialize population densities
        x = num_density_init
        for t_idx in range(self.horizon_len):
            # skip if any population density is too small
            if any(x < 0.1):
                continue
            # forward propagation
            x = self.surrogate_model.forward(x, t_idx, self.agent)
        # compute terminal loss
        loss = cirterion(x, ref)
        # backpropagate adjoint state
        loss.backward()
        # update model parameters
        optimizer.step()
        self.lr_update(optimizer, lr, step, max_iterations)
```

```
# Surrogate retention system implementation
def forward(self, x, t, agent):
    # performance measure as bce loss
    criterion = torch.nn.BCELoss(reduction='none')
    # get all users from the holdout set
    user_state = self.user_data.return_all_users()
    # model outputs with current models
    actions = agent.step(user_state, t)
    performance = []
    for index_demo, data in user_state.items():
        # get true label
        label = data["label"]
        # compute rewards
        R = -criterion(actions[index_demo], label)
        # compute worst-case distributional loss
        eta = self.grid_search_etaStar(x[index_demo], Reward_all)
        C = self.compute_c(x[index_demo])
        Reward = C * (torch.nn.ReLU()(R - eta)**2).mean().sqrt() + eta
        # collect reward
        performance.append(Reward)
    performance = torch.stack(performance)
    # compute survival rate based on performance
    survival_rate = self.survival_rate(performance)
    # compute birth rate based on performance
    birth_rate = self.birth_rate(performance)
    # next density from survival users
    x_survival = survival_rate * x
    # next density from new coming users
    x_newborn = birth_rate * x
    x_next = x_survival + x_newborn
    return x_next
```

C. Details On the Population Retention System

Algorithm 2 illustrates the detailed implementation of the population retention system. In the following, we explain each important step in detail.

- Data construction: Given a dataset $\mathcal{D}$, we categorize $\mathcal{D}$ based on the sensitive attribute (e.g., in this work, we consider the sex attribute), which produces $\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^K$ for K demographic groups. Within each dataset, we randomly select N^i samples $\{\mathbf{z}^{i,n}\}_{n=1}^{N^i}$ to construct the population. The feature $\mathbf{z}^{i,n}$ contains user input (e.g. voice recommend), and a probability $p_1^{i,n}$ for user churn when received with a correct model prediction, a probability $p_2^{i,n}$ for user retention when received with a wrong model prediction.
- State initialization: Based on the given initial population densities $\boldsymbol{\lambda}_0$, we randomly select $\lambda_0^i \cdot N^i$ users as active, which produces $\mathcal{A}_0^i$. For instance, when $N^1 = 4$ and $\lambda_0^1 = 0.5$, a possible $\mathcal{A}_0^1 = (1, 0, 1, 0)$, which indicates the 1st and 3rd users are active initially.
- Return active users: Based on the N-tuple $\mathcal{A}_t^i$, we collect features contributed by all active users $\mathbf{z}^{i,n}$ if $[\mathcal{A}_t^i]_n = 1$.
- Model prediction: The tth model makes a prediction on a user feature $\mathbf{z}^{i,n}$. The outcome is a binary value in which 1 indicates a correct prediction and 0 corresponds to a wrong prediction.
- Environment update: A currently active user remains active at the next time step if both model prediction is correct and the Flag is true,

$$[\mathcal{A}_{t+1}^i]_n = \begin{cases} 1 & \text{if Flag} = 1 \text{ and } \boldsymbol{\theta}_t(\mathbf{z}^{i,n}) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Algorithm 2 Design of Population Retention System for Evaluation.

```
Input: Dataset $\mathcal{D}$, (Data construction)
a sequence of models $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$,
initial population densities $\boldsymbol{\lambda}_0$.
number of episodes: max episodes,
number of iterations: horizon,
Output: Population densities at all time step $\{\boldsymbol{\lambda}_t\}_{t=0}^{T}$.
for episode = 1 to max episodes do
  // Set up initial user state based on initial population densities.
  Set up $\mathcal{A}_0^i, \forall i = [1, 2, ..., K]$ (State initialization).
  for t = 1 to horizon do
    // Return the features of currently active users.
    for i = 1 to K do
      Collect $\{\mathbf{z}^{i,n}\}$ if $[\mathcal{A}_t^i]_n = 1$, (Return active users)
    end for
    // Model prediction based on the provided features.
    for i = 1 to K do
      Model predict $\{\boldsymbol{\theta}_t(\mathbf{z}^{i,n})\}$ if $[\mathcal{A}_t^i]_n = 1$, (Model prediction)
    end for
    // Environment update for active users based on model predictions.
    for i = 1 to K do
      $[\mathcal{A}_{t+1}^i]_n = 1$ if $\boldsymbol{\theta}_t(\mathbf{z}^{i,n}) = $ True, and Flag = True, (Environment update)
    end for
  end for
end for
```

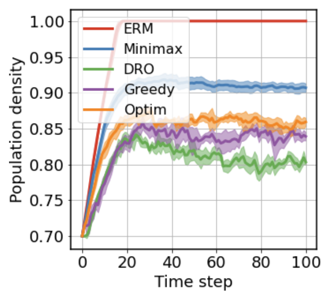
where the Flag is computed based on $p_1^{i,n}$ and $p_2^{i,n}$, given a uniform random number $p \in [0, 1]$,

$$\text{Flag} = \begin{cases} \text{True} & \text{if } \boldsymbol{\theta}_t(\mathbf{z}^{i,n}) = 1 \text{ and } p \leq p_1^{i,n} \text{ or } \boldsymbol{\theta}_t(\mathbf{z}^{i,n}) = 0 \text{ and } p \geq p_2^{i,n} \\ 0, & \text{otherwise} \end{cases}$$

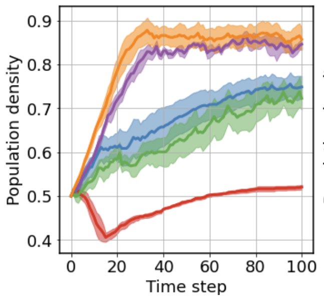
The population retention system returns a trajectory of population densities. Fig. 3 shows the outputs simulated from both environments $\mathcal{P}_1$ and $\mathcal{P}_2$ from 5 repeated simulations with different random seeds. This simulation results match with the experimental results shown in Fig. 1 where population densities $\boldsymbol{\lambda}_t$ is converted to a loss $\Psi(\boldsymbol{\lambda}_t, \mathbf{1})$, and $\Psi(\boldsymbol{\lambda}_t, \mathbf{1})$ computes the binary cross-entropy loss between $\boldsymbol{\lambda}_t$ and $\mathbf{1}$.

The following illustrates a Python implementation of the environment. If model training is required, the t^{th} model is optimized from the features of currently active users at time step t. The proposed optimal control method generates all models based on a surrogate retention system, which does not need access to the population retention system during model generation. It optimizes all models before the evaluation begins (denoted as agent.pmp()).

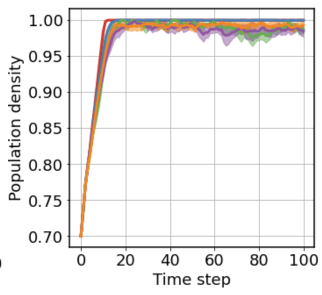
```python
# Implementation of the population retention system
for episode in range(max_episodes):
    user_data.initialize_state(num_density_init)
    if agent.agent_name == "optim-control" and training == True:
        agent.pmp()
    for t in range(horizon_len):
        state = user_data.return_active_users()
        if training == True:
            agent.update(state, episode, t)
        actions = agent.step(state, t)
        state_updated, step_info = environment.step(state, actions, t)
        user_data.update(state_updated)
```



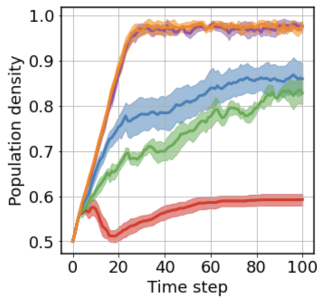
(a) Majority in $\mathcal{P}_1$



(b) Minority in $\mathcal{P}_1$



(c) Majority in $\mathcal{P}_2$



(d) Minority in $\mathcal{P}_2$

Figure 3. Under simulation environment $\mathcal{P}_1$ with the synthetic dataset, (a) and (b) show the population densities of majority and minority demographic groups, respectively. Under simulation environment $\mathcal{P}_2$ with the synthetic dataset, (c) and (d) show the population densities of majority and minority demographic groups, respectively.

```python
# Implementation of population retention system under MDP.
def step(self, state: dict, actions: dict, t: int):
    state_updated = dict()
    for index_demo, data in state.items():
        # get model prediction
        model_output = actions[index_demo]
        # get true label
        label = data["label"]
        # get prediction
        performance = model_output == label
        # probability of a user churn when received a correct prediction
        p_1 = data["p_1"]
        # probability of a user retention when received a wrong prediction
        p_2 = data["p_2"]
        # number of users in population
        num_total = data["total_number_of_population"]
        # indices of currently active users
        indices_active = data["indices_of_active_users"]
        # number of users that are currently active
        num_of_active_users = len(indices_active)
        # a uniform number in range [0, 1]
        rand_uniform = torch.rand_like(p_1)
        # indicator of user churn when received correct predictions
        indices_correct_prediction_but_leave = rand_uniform < p_2
        # indicator of user retention when received wrong predictions
        indices_wrong_prediction_but_stay = rand_uniform < p_2
        # indices of new users
        new_user_indices = []
        for user_index in range(num_of_active_users):
            if performance[user_index] == False
            and indices_wrong_prediction_but_stay[user_index] == False:
                continue
            if performance[user_index] == True
            and indices_correct_prediction_but_leave[user_index] == True:
                continue
            # a survived user
            new_user_indices.append(indices_active[user_index])
        # number of survived users
```

```
num_user_survived = len(new_user_indices)
# indices of currently inactive users
indices_inactive
= [ii for ii in range(num_total) if ii not in new_user_indices]
# randomly shuffle the indices of inactive users
random.shuffle(indices_inactive)
# compute survival rate
survival_rate = len(new_user_indices) / num_of_active_users
# compute birth rate
birth_rate = self.new_users_ratio(survival_rate)
# number of incoming users
num_new_users = int(num_of_active_users * birth_rate)
# indices of active users at the next time step
new_users = new_user_indices + indices_inactive[:num_new_users]
# compute population density at the next time step
density = len(new_users) / num_total
# state update
state_updated[index_demo] = {"density": density,
                             "ids_active": new_users,
                             }
```