University of California
Santa Barbara

Self-Healing Robust and Fair Neural Networks via Optimal Control

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Zhuotong Chen

Committee in charge:

Professor Zheng Zhang, Chair
Professor Upamanyu Madhow
Professor Ramtin Pedarsani
Professor Qianxiao Li

September 2024

The Dissertation of Zhuotong Chen is approved.

Professor Upamanyu Madhow

Professor Ramtin Pedarsani

Professor Qianxiao Li

Professor Zheng Zhang, Committee Chair

July 2024

Self-Healing Robust and Fair Neural Networks via Optimal Control

by

Zhuotong Chen

Acknowledgements

I extend my deepest gratitude to my advisor, Prof. Zheng Zhang, for the unwavering support and guidance throughout my Ph.D. journey. Beginning as a visiting researcher, it was your encouragement and mentorship that inspired me to pursue my research as a Ph.D. student. During the lowest points of my life, especially amid the pandemic, your support was instrumental in helping me navigate through the challenges.

I would like to express my deepest gratitude to my mentor, Prof. Qianxiao Li, for his unwavering support and guidance throughout my Ph.D. journey. His mentorship has been invaluable in shaping my ability to think critically, formulate, and solve research problems. I am also grateful to my committee members, Prof. Upamanyu Madhow and Prof. Ramtin Pedarsani, for their support and insightful feedback.

A heartfelt thank you to my friends, who enriched my time in Santa Barbara and made it truly wonderful. Your companionship and support have been a source of strength and joy throughout my Ph.D. journey. The laughter, discussions, and shared experiences have not only made this journey enjoyable but have also created lasting bonds that I will treasure forever.

To my family, thank you for your unconditional love and support. None of this would have been possible without you. Your belief in me has been my foundation, giving me the courage and confidence to pursue my dreams.

Curriculum Vitæ

Zhuotong Chen

Education

2024	Ph.D. in Communications, Controls, and Signal Processing (Expected), University of California, Santa Barbara.
2021	M.S. in Communications, Controls, and Signal Processing, University of California, Santa Barbara.
2018	B.S. in Electrical and Computer Engineering, University of Manitoba.

Publications and Articles

1. **Chen, Z.**, Wang, Z., Yang, Y., Li, Q., Zhang, Z. (2024). PID control-based self-healing to improve the robustness of large language models. *Transactions on Machine Learning Research.*
2. **Chen, Z.**; Li, Q., Zhang, Z. Asymptotically Fair Participation in Machine Learning Models: an Optimal Control Perspective. Under submission. arXiv preprint arXiv:2311.10223.
3. **Chen, Z.**, Ma, Y., Kveton, B. and Deoras, A. (2023) "Active Learning with Crowd Sourcing Improves Information Retrieval", *Interactive Learning with Implicit Human Feedback workshop at ICML.*
4. **Chen, Z.**, Li, Q. and Zhang, Z. (2023). "Fairness in a Non-Stationary Environment from an Optimal Control Perspective", *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems.*
5. **Chen, Z.**, Li, Q. and Zhang, Z. (2022). "Self-Healing Robust Neural Networks via Closed-Loop Control", *Journal of Machine Learning Research (JMLR), 23(319), 1-54.*
6. **Chen, Z.**, Li, Q. and Zhang, Z. (2021). "Towards Robust Neural Networks via Close-loop Control", *International Conference on Learning Representations (ICLR).*
7. Wicker, M., Laurenti, L., Patane, A., **Chen, Z.**, Zhang, Z. and Kwiatkowska, M. (2021, March). Bayesian inference with certifiable adversarial robustness. *In International Conference on Artificial Intelligence and Statistics (pp. 2431-2439). PMLR.*
8. **Chen, Z.**, Zheng, S. and Okhmatovski, V. I. (2019). "Tensor train accelerated solution of volume integral equation for 2-D scattering problems and magneto-quasi-static characterization of multiconductor transmission lines", *IEEE Transactions on Microwave Theory and Techniques, 67(6), 2181-2196.*
9. **Chen, Z.**, Gomez, L. J., Zheng, S., Yucel, A. C., Zhang, Z. and Okhmatovski, V. I. (2019), "Sparsity-aware precorrected tensor train algorithm for fast solution of 2-D scattering problems and current flow modeling on unstructured meshes". *IEEE Transactions on Microwave Theory and Techniques, 67(12), 4833-4847.*
10. **Chen, Z.**, Gholami, R., Mojolagbe, J. B. and Okhmatovski, V. (2018), "Formulation of surface-volume-surface-EFIE for solution of three-dimensional scattering problems on composite dielectric objects", *IEEE Antennas and Wireless Propagation Letters, 17(6), 1043-1047.*

Internship Experience

Amazon Web Services *Applied Scientist Intern* June - September 2023

- Title: Fine-tuning Large Language Models to Design Recommender Systems.
- I use reinforcement learning techniques, such as policy gradient, and proximal policy optimization, to fine-tune large language models to maximize the probability of user clicks for recommender system design.

Amazon Web Services *Applied Scientist Intern* August - December 2022

- Title: Active Learning with Crowd Sourcing Improves Information Retrieval
- I developed a system named Crowd-Coachable Retriever, which conducts active learning for information retrieval using only publicly available human resources and reproducible training procedures.

Oxford University *Research assistant* June - September 2019

- Title: Bayesian Inference with Certifiable Adversarial Robustness
- I considered adversarial training of deep neural networks through the lens of Bayesian learning and presented a principled framework for adversarial training of Bayesian Neural Networks with certifiable guarantees.

Abstract

Self-Healing Robust and Fair Neural Networks via Optimal Control

by

Zhuotong Chen

This dissertation investigates the challenges of improving the robustness and fairness of deep neural networks through the lens of optimal control theory. Deep neural networks, despite their extensive application across various engineering fields, are vulnerable to imperceptible perturbations and often exhibit biased performance towards underrepresented demographic populations. The first part of this dissertation presents a novel self-healing framework designed to improve the robustness of deep neural networks against unforeseen perturbations. This framework is realized by a novel closed-loop control approach grounded in optimal control theory, which adaptively generates control signals to identify and correct potential errors in the state trajectory of perturbed input data during inference. The second part of this dissertation introduces a PID control framework that generalizes the closed-loop method with additional integral and derivative controllers. We derive an analytical solution for fast online inference, making our control framework applicable to large-scale models. The third part of this dissertation addresses the fairness issue in machine learning within dynamic environments, where undesired model biases against minority users could lead to significant user churn, thereby diminishing the training data for model tuning in subsequent time steps. This negative feedback loop can further exacerbate demographic disparity. To address this, we introduce the concept of asymptotic fairness to maintain consistent model performance across all demographic groups and propose an optimal control solution to achieve this goal.

Contents

Chapter 1

Motivation

1.1 Trustworthy Machine Learning

As machine learning technology gets applied to actual products and solutions, new challenges have emerged. Models unexpectedly fail to generalize to small changes in the distribution; some models are found to utilize sensitive features that could treat certain demographic user groups unfairly; models tend to be confident on novel data they have never seen, or models cannot communicate the rationale behind their decisions effectively with the end users like medical staff to maximize the human-machine synergies. Collectively, we face a trustworthiness issue with the current machine learning technology. A summary of machine learning trustworthy issues is demonstrated in Figure 1.1. Below we discuss these issues and their importance.

Robustness. Robustness in machine learning is a measure of a model's ability to maintain performance when exposed to variations, noise, and adversarial attacks. More specifically, when imperceptible perturbations are applied to input data, the performance of state-of-the-art machine learning models drops to surprisingly low levels, often falling significantly below their expected accuracy rates [1, 2]. Machine learning robustness is crucial for ensuring that models perform reliably across diverse and unforeseen conditions. In real-world applications, robust models are essential for maintaining high accuracy and safety standards. For instance, in healthcare, robust machine learning models are vital for accurate diagnoses and treatment recommendations across different medical institutions, thus ensuring patient safety. Similarly,

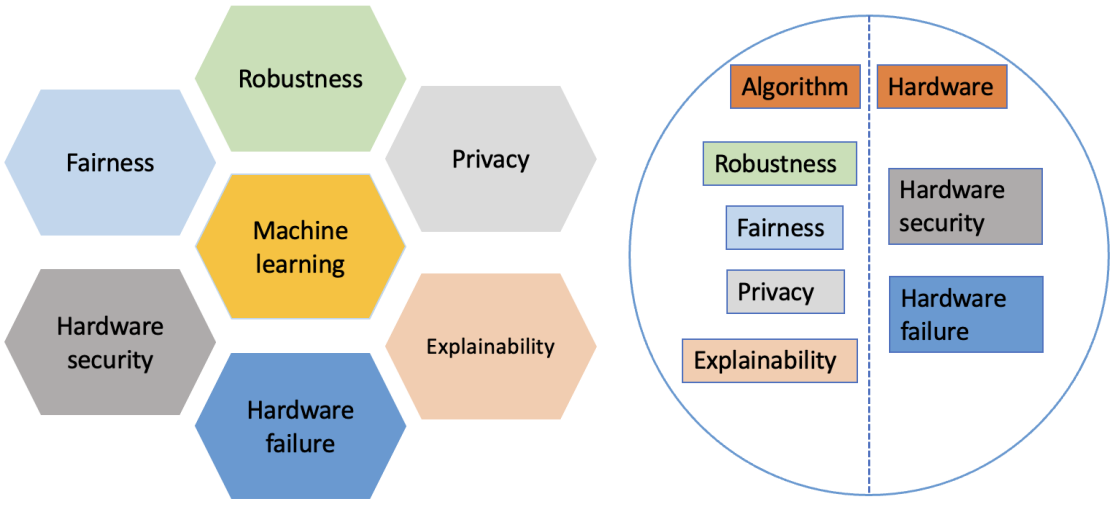


Figure 1.1: (a): Various trustworthy issues in machine learning. (b): Algorithm-level and hardware-level trustworthy issues in machine learning.

autonomous vehicles rely on robust models to navigate safely in varied and unpredictable environments. The robustness issue is crucial as it significantly impacts both market viability and governmental policies [3]. Governments worldwide are recognizing these risks; initiatives such as the European Union's AI Act aim to establish stringent regulations ensuring AI reliability and safety [4].

Fairness. The machine learning fairness problem is essential to prevent biases from influencing decisions in machine learning. For instance, in hiring processes, if not designed with fairness in mind, these models might favor certain demographics over others, perpetuating inequality [5]. Similarly, in predictive policing, biased models could disproportionately target specific communities, amplifying social disparities. Machine learning fairness issues significantly impact the market by affecting consumer trust, brand reputation, and exposing companies to legal and financial risks, while also offering opportunities for ethical differentiation and investment attraction [6]. To address these challenges, governments are implementing regulatory frameworks, developing guidelines and standards, funding research, fostering public-private partnerships, raising public awareness, and establishing ethics committees and oversight bodies.

Privacy. Machine learning privacy issues primarily arise from the collection, use, and sharing of vast amounts of personal data required to train and refine models [7]. The privacy issues are critical due to the sensitivity of the data involved, regulatory compliance requirements, and the impact on user trust and adoption. The market is increasingly valuing privacy, with financial repercussions for data breaches and a growing demand for privacy-enhancing technologies [8]. Government initiatives worldwide, through legislation like GDPR and CCPA, funding for privacy-preserving research, and public education campaigns, are driving the adoption of privacy measures [9].

Explainability. Explainability in machine learning refers to the ability to interpret the decisions made by machine learning models, ensuring transparency, accountability, and trust in their outputs [10]. In marketing, explainability helps businesses understand and justify the decisions made by ML models, enhancing transparency and fostering consumer trust. For government initiatives, explainability is crucial for accountability in applications such as criminal justice, healthcare, and public policy [11].

Hardware security. Hardware security in machine learning is essential to protect physical components from attacks, such as fault injections. This issue is critical because vulnerabilities in hardware can undermine the performance of entire machine learning system, despite algorithms being implemented. The marketing impact of hardware security breaches can result in substantial financial losses and damaged reputations. Companies that emphasize secure hardware gain a competitive edge. Government initiatives, including the EU Cybersecurity Act, highlight the importance of secure hardware in protecting national security and critical infrastructure. These initiatives provide funding and regulatory frameworks to enhance hardware security, reflecting its crucial role in maintaining consumer trust and complying with regulatory standards [12].

Hardware failure. Hardware failure in machine learning is a significant issue that impacts the reliability of models. Key issues include processing variations in hardware like ASICs, which

arise from manufacturing inconsistencies, leading to discrepancies in computation accuracy. Another issue is the mismatch in model parameters due to the precision limitations of deployment hardware, which can degrade model performance. Hardware aging further exacerbates these problems. In photonic components, thermal sensitivity issues can cause significant deviations in computations due to the delicate nature of photonic elements. Addressing these hardware failures is crucial for ensuring the trustworthiness of machine learning systems, especially in critical applications where reliability and robustness are paramount [13].

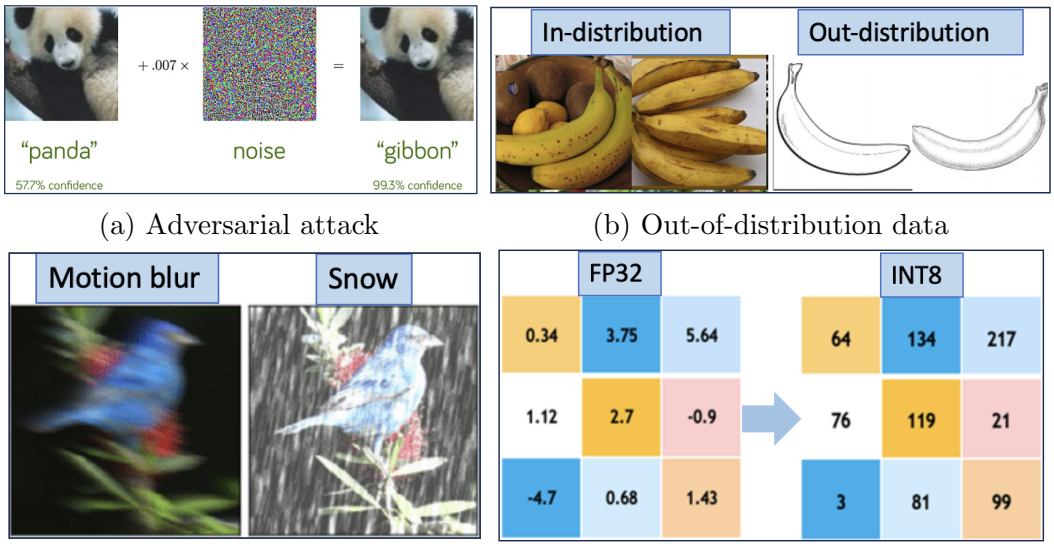
In the next Sections, we discuss more details about robustness and fairness issues, which are the focus on this dissertation.

1.2 Robustness of Deep Neural Networks

Due to the increasing data and computing power, deep neural networks have achieved state-of-the-art performance in many applications such as computer vision and natural language processing. However, many deep neural networks are vulnerable to various malicious perturbations due to their black-box nature: a small (even imperceptible) perturbation of input data may lead to completely wrong predictions [14, 15]. This has been a major concern in some safety-critical applications such as autonomous driving [16] and medical image analysis [17]. In the following, we discuss various sources that cause robustness issues and solutions for improving model robustness.

1.2.1 Various Sources Causing Robustness Issues

Robustness issues in machine learning learning arise from various sources. These issues can be categorized into data-related, model-related, and hardware deployment robustness issues. Data-related robustness issues include adversarial attack [1], out-of-distribution data [18], data corruption [18]. Model-related robustness issue refers to perturbations in model parameters [19]. Hardware deployment related-robustness are critical, especially when implementing models on specialized hardware like ASIC chips [20]. In the following, we discus each of these three



(a) Adversarial attack (b) Out-of-distribution data

(c) Data corruption (d) Quantization

Figure 1.2: (a): Adversarial attack [1] causing a panda image to be misclassified as a gibbon. (b): Out-of-distribution data illustrated by a sketch of a banana [21]. (c): Data corruption exemplified by the application of motion blur and snow effects. (d): Quantization from FP32 to INT8.

robustness issue categories in details.

Data-related robustness issues.

- **Adversarial attack [22, 23].** Let F denote a deep neural network, $\mathbf{x}$ be a natural input data and y be its label. An adversarial attack searches for a perturbed counterpart $\tilde{\mathbf{x}}$ such that the well-trained deep neural network misclassifies $\tilde{\mathbf{x}}$ as a wrong label instead of y .

$$y \neq \arg\max F(\tilde{\mathbf{x}}, \boldsymbol{\theta}), \quad s.t. \quad \|\tilde{\mathbf{x}} - \mathbf{x}\| \leq \epsilon,$$

  Where $\|\cdot\|$ denotes a norm (e.g., $\|\cdot\|_2$ measures the Euclidean distance between the original input $\mathbf{x}$ and the perturbed input $\tilde{\mathbf{x}}$) and ϵ is the perturbation budget, a deep neural network is considered ϵ-robust with respect to $\tilde{\mathbf{x}}$ if no valid perturbation $\tilde{\mathbf{x}}$ can be found. An example of adversarial attack is shown in Figure 1.2 (a) where a panda image is misclassified as a gibbon when small noise is applied.

- **Out-of-distribution data [18].** Out-of-distribution robustness refers to the ability of a machine learning model to perform well on data that is different from the data it was trained on. This is crucial because real-world applications often encounter scenarios where the input data distribution shifts or changes from the training distribution [24]. Machine learning models are typically trained on a specific dataset that represents a particular distribution of data. This dataset might not capture all possible variations that could occur in the real world. When the model is deployed, it might encounter data that has different characteristics compared to the training data. This out-of-distribution data can arise due to various reasons, such as changes in the environment, different population characteristics, or unexpected events. An example of out-of-distribution data is shown in Figure 1.2 (b) with a sketch of a banana.

- **Data corruption [25].** Data corruption in machine learning refers to the introduction of errors, noise, or unintended alterations in the dataset, which can significantly impact the model performance and reliability. This corruption can occur during data collection, storage, or preprocessing stages. Data corruption undermines the ability of model to generalize and make accurate predictions, leading to potentially harmful decisions, especially in critical applications like healthcare, finance, and autonomous systems. An example of data corruption is shown in Figure 1.2 (c) where motion blue and snow condition are applied to an image of bird.

Model-related robustness issues.

- **Model parameter error [26].** Model parameter error in machine learning refers to the inaccuracies and uncertainties that arise from the limitations of numerical precision in computational processes. In deep learning, models often involve complex mathematical operations and large-scale data processing, which require high precision to ensure accurate results. However, due to hardware constraints and finite memory, these operations are typically performed with limited precision, leading to rounding errors and approximation

inaccuracies. These precision errors can accumulate and propagate through the layers of a neural network, potentially resulting in significant deviations from expected outputs [27]. Precision error undermines the robustness of models, particularly in critical applications where even small inaccuracies can lead to incorrect predictions or decisions. An example of precision error caused by quantization is shown in Figure 1.2 (d).

- **Parameter perturbation attacks [19].** Parameter perturbation in the robustness of deep learning involves intentionally altering the parameters (weights and biases) of a neural network. This technique helps evaluate how small or significant modifications in the model's parameters affect its performance, particularly when exposed to out-of-distribution data or adversarial attacks.

Hardware failure and uncertainty.

- **Process Variation in ASIC Deployment [28].** Application-Specific Integrated Circuits (ASICs) are custom chips designed for specific tasks, including machine learning inference. During the manufacturing process, slight variations can occur due to differences in temperature, pressure, and material quality, among other factors. These process variations can lead to discrepancies in the physical properties of the transistors and other components on the chip, which in turn can cause deviations in the expected model parameter values.

- **Hardware Aging [29].** Over time, hardware components degrade due to various factors such as material fatigue. This degradation can affect the performance and reliability of the machine learning models deployed on these devices. Aging can cause changes in the electrical characteristics of the components, leading to drift in the parameter values and reduced accuracy of computations.

- **Sensitivity with respect to external factors [30].** This is a common issue in analog AI hardware such as photonic AI accelerators. Photonic AI hardware offers significant

advantages in terms of speed and energy efficiency for machine learning tasks. However, these devices are highly sensitive to external conditions such as temperature fluctuations, mechanical vibrations, and manufacturing imperfections. This sensitivity can introduce noise into the parameters of the ML model, leading to inaccuracies.

1.2.2 Robustness Improvement in Deep Learning

Numerous methods have been proposed to improve the robustness of machine learning models, which can be categorized into two types: training-based defenses that focus on improving the model itself, and data-based defenses that leverage the underlying data information.

Training-based defense. A leading example of training-based defense is adversarial training [23], which employs robust optimization techniques and has proven effective against all forms of adversarial attacks [31].

$$\min_{\theta} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\|\tilde{\mathbf{x}}-\mathbf{x}\|\leq\epsilon} \mathcal{L}(F(\tilde{\mathbf{x}};\boldsymbol{\theta}),y) \right],$$

where $\boldsymbol{\theta}$ are model parameters, $\mathcal{D}$ is the data distribution, and $\mathcal{L}$ is the loss function (e.g., cross-entropy loss) that measure the discrepancy between model prediction and truth label. Nonetheless, training-based defenses presents several key challenges. (1) training-based defenses may incur significant computational costs in large-scale deployments [32]. The computational demand is substantial [33], as finding adversarial examples in the discrete domain of natural language processing often requires addressing a combinatorial optimization problem [34], which becomes exponentially more complex as the problem size increases. (2) Improving adversarial robustness might inadvertently impair the model's performance on regular, unperturbed datasets [35]. (3) Adversarial training is less effective against unforeseen adversarial perturbations [36].

Data-based defense. On the other hand, data-based defenses are introduced as alternatives to training-based defenses in part to alleviate some of the aforementioned issues [37, 38]. Data-based defenses aim to enhance model robustness by leveraging inherent data properties. These methods typically involve detecting or filtering out perturbed data, thus maintaining model performance. The idea is that since perturbed data are generated from clean data by adding imperceptible perturbations, it is possible to decontaminate them by searching for more probable data within a small distance of the original ones. However, the mechanisms underpinning these methods are not fully understood, they have not been able to achieve state-of-the-art performance [31].

1.3 Fairness of Deep Neural Networks

Ensuring the fairness of deep neural networks is critical to avoid reinforcing or exacerbating existing biases in society. Fairness problem is especially important in sensitive domains like education, finance, and healthcare, where decisions significantly impact individuals [39]. In education, fair algorithms ensure that all students, regardless of their background, have equal access to opportunities and are not discriminated against based on race, gender, or socioeconomic status. This promotes diversity and equity, fostering a more inclusive educational environment. In finance, fairness ensures equal access to financial services, preventing economic disparities and helping institutions comply with regulatory standards. Unfair financial algorithms can deny loans to minority groups, perpetuating cycles of poverty and limiting economic mobility. In healthcare, fairness ensures accurate and equitable diagnoses and treatments for all patients, reducing health disparities and fulfilling ethical responsibilities [40]. Biases in healthcare algorithms can lead to unequal treatment, worsening health outcomes for minority groups. Addressing challenges such as biased training data, lack of algorithmic transparency, and the need for fairness-aware algorithms is crucial [41].

1.3.1 Various Definitions of Fairness

The fairness property can be defined differently, which leads to different results. We introduce three popular fairness definitions, including demographic parity, equalized odds and individual fairness, and discuss their advantages and drawbacks.

Demographic parity. Demographic Parity [42] requires the decision-making process to be independent of a protected attribute such as race, gender, or age. This means that the probability of a positive outcome (e.g., receiving a loan or being hired) should be the same across different groups defined by the protected attribute. This ensures that members of different groups have equal chances of receiving the positive outcome. However, it is important to note that achieving demographic parity does not necessarily mean the classifier is fair in all contexts, as it might ignore differences in base rates (the true proportion of positive outcomes) between groups. For instance, if one group has a naturally higher incidence of the positive outcome, enforcing demographic parity might lead to either an unfair burden on the model or unequal treatment in practice.

Equalized odds. Equalized Odds [43] is a fairness criterion that ensures a classifier's performance is equally accurate for all groups defined by a protected attribute. Specifically, it requires that both the true positive rate and false positive rate are equal across these groups. This means that, for example, the probability of correctly identifying a positive instance (true positive rate) and the probability of incorrectly identifying a negative instance as positive (false positive rate) must be the same regardless of the group. Equalized Odds helps prevent biased decision-making that could disproportionately benefit or harm specific groups, but it can be challenging to achieve in practice, especially when base rates differ between groups.

Individual fairness. Individual Fairness [44] is a principle that emphasizes treating similar individuals similarly. This concept is rooted in the idea that fairness should not only be assessed at the group level but also at the individual level. This ensures that two individuals who are

Table 1.1: Various Fairness Definitions and Explanations

Definition	Explanation
Demographic Parity [42]	It requires the decision-making process to be independent of a protected attribute
Equalized Odds [43]	It is a fairness criterion that ensures a classifier's performance is equally accurate for all groups defined by a protected attribute
Individual Fairness [44]	This is a principle that emphasizes treating similar individuals similarly
Equal Opportunity [43]	The true positive rate should be the same across groups
Counterfactual [42]	Counterfactual fairness is concerned with ensuring that decisions would be the same in a counterfactual world where the individual belonged to a different demographic group.
Causal Fairness [45]	Causal fairness deals with the causal relationships between variables to ensure fairness.
Procedural Fairness [46]	Procedural fairness focuses on the fairness of the processes leading to outcomes rather than the outcomes themselves.

alike in relevant aspects (e.g., qualifications for a job or risk profile for a loan) receive similar treatment from the model. Individual Fairness requires a careful definition of the similarity metric, which can be context-specific and may involve subjective judgment.

Other fairness definitions. There are many other fairness definitions [41], such as Equal Opportunity [43], Counterfactual fairness [42], Causal Fairness [45], Procedural Fairness [46]. Table 1.1 summarizes all fairness definitions.

1.3.2 Existing Solutions of Machine Learning Fairness

Fair representation learning. Fair representation learning aims to create representations of data that remove or mitigate the influence of sensitive attributes, such as race or gender, to ensure that downstream models do not propagate bias [47]. One common approach is to use adversarial training [23], where a neural network is trained to produce representations that are useful for the primary task (e.g., classification) but uninformative about the sensitive attribute. This is achieved by including an adversary that tries to predict the sensitive attribute from

the representation, and the main model is trained to minimize the primary task loss while maximizing the adversary's loss. This method helps in reducing discriminatory outcomes in machine learning models by ensuring that the learned representations are fair and unbiased with respect to the sensitive attributes.

Fairness constrained training. Applying fairness constraint is a method that requires a model's predictions to be independent of sensitive attributes given the true outcome [43]. In other words, for a binary classification problem, the false positive rate and false negative rate should be equal across different groups defined by the sensitive attribute. This can be achieved through post-processing techniques, where the output probabilities of a classifier are adjusted to ensure equalized odds. Alternatively, constraints can be incorporated into the training process to directly optimize for equalized odds. This method aims to ensure that individuals from different demographic groups are treated equally by the model, reducing disparities in prediction errors.

1.4 Thesis Contributions

1.4.1 Self-Healing Methods to Improve AI Robustness

Research gap for existing robustness study Predominantly, existing robustness improvement methods are rooted in the foundation of adversarial training to overcome the effects of specific adversarial perturbations. This is achieved by adjusting either the entire set of model parameters or a significant portion thereof [48]. Despite its effectiveness, this approach raises three critical concerns. Firstly, adjusting model parameters using adversarial examples requires substantial computational resources [33]. Secondly, there exists a potential trade-off where improved adversarial robustness may lead to compromised performance on standard, natural datasets [35]. Thirdly, and more problematically, adversarial training is less effective against unforeseen adversarial perturbations [36]. This limitation becomes particularly noticeable when deploying models in practice, where anticipating the potential adversarial attacks in advance is

nearly impossible.

Thesis contribution on improving model robustness. To address these challenges, we explore a self-healing process inspired by the mechanisms of a robust biological immune system. Figuratively speaking, self-healing properties can be attributed to systems or processes that inherently or intentionally correct any disturbances.

- We propose a self-healing process to autonomously rectify or alleviate undesirable issues. We examine this concept as a cost-effective strategy to enhance the robustness of deep neural networks against various perturbations. On many experiments, the proposed method can consistently improve the robustness of the pre-trained models against various perturbations.

- The self-healing robust neural network framework is computationally prohibitive for large-scale deep neural networks, such as large language models. We revisit the self-healing robust neural network framework and derive a fast closed-form solution. This advancement makes the self-healing framework applicable for use with large language models that have more than 1 billion parameters.

1.4.2 Optimal Control to Improve Long-Term Model Fairness

Research gap for existing fairness study. Existing methods on addressing machine learning fairness problem have focused on a static setting where the data distribution does not change over time. However, practical machine learning models are often deployed in a non-stationary environment [49]. Specifically, the distribution of training data evolves with time, and the deployed model needs to be adjusted accordingly. The adjusted model will influence users' participation and future training data in turn. Consider a scenario wherein a deployed model is continuously fine-tuned based on the majority's input. Minority users may find that their interactions with the machine learning model are unsatisfying. As a result, these minority users tend to not engage with the model. The deployed model, which is already under-representing

them, gets even less data from minority users, leading to even worse services to this group [50]. Existing works have applied static fairness constraint at each time step to ensure fair treatment across all demographic groups [51]. However, this can inadvertently introduce undesirable long-term effect [49]. Alternatively, reinforcement learning methods have been applied to optimize fairness metric over a long-term horizon [52, 53]. Due to the complex non-stationary environment, these methods require significant computations for exploring the optimal model and often lead to suboptimal solution.

Thesis contribution on improving model fairness. With considerations of the interplay between a machine learning model and users' participation, we propose a model-based optimal control method to engage users from all demographic groups over a long time horizon. The optimal control formulation accounts for long-term planning, therefore it can achieve asymptotic fairness in a non-stationary setting. The proposed optimal control method initially biases the machine learning model towards the minority group to attract more users, and control the long-term behavior of the machine learning model.

Chapter 2

Background

This chapter provides background information on ordinary differential equations and optimal control theory, which are essential building blocks for this dissertation.

2.1 Background on Ordinary Differential Equations

This section introduces some basics of ordinary differential equations. An ordinary differential equation (ODE) is the equation

$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t), \quad \mathbf{x}_0 \in \mathbb{R}^d, \tag{2.1}$$

where $\dot{\mathbf{x}}_t$ denotes the time derivative, $F : \mathbb{R}^d \to \mathbb{R}^d$ is a function or vector field and $\mathbf{x}_0$ is the initial condition. This is called a time-homogeneous ODE since the vector field on the right does not depend explicitly on time t. On the other hand, a time-inhomogeneous ODE is given by

$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t, t), \quad \mathbf{x}_0 \in \mathbb{R}^d. \tag{2.2}$$

Notice that Equation (2.2) includes Equation (2.1). A solution of an ODE on $[0, T]$ is a function $\mathbf{x} : [0, T] \to \mathbb{R}^d$ with $\mathbf{x} := \{\mathbf{x}_t : t \in [0, T]\}$ that satisfies Equation (2.2).

The definition of solution requires $\mathbf{x}$ to be differentiable on $[0, T]$. Equation (2.2) can be

written as

$$\mathbf{x}_t = \mathbf{x}_0 + \int_0^t F(\mathbf{x}_s, s)ds. \tag{2.3}$$

2.1.1 Numerical Solutions of ODEs

In general, ODEs do not admit explicit solutions and a solution needs to be computed numerically. This section presents the simplest possible method, the forward Euler method. In this method, a solution of the ODE in Equation (2.2) is constructed by discretizing time,

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \triangle t F(\hat{\mathbf{x}}_k, k), \;\; \hat{\mathbf{x}}_0 = \mathbf{x}_0, \tag{2.4}$$

which can be seen as a first-order Taylor expansion of the integral form of the ODE in Equation (2.3) for small $\triangle t$. The latter is called the step size. This approximation is expected to get better as the step size $\triangle t$ becomes small. The following theorem makes this precise.

Theorem 2.1.1 *Let F be Lipschitz in $\mathbf{x}$ uniformly in t and continuous in t. Let $\mathbf{x}$ be a solution of the ODE defined in Equation (2.2) with initial condition $\mathbf{x}_0$ and $\hat{\mathbf{x}}$ be the iterates defined in Equation (2.4), then for each $K > 0$, there exists a constant $C > 0$ such that*

$$\max_{k \leq K} \|\hat{\mathbf{x}}_k - \mathbf{x}_{(k\triangle t)}\| \leq C \triangle t.$$

2.1.2 Connection between ODE and Deep Neural Networks

The intersection of ODEs and deep neural networks has emerged as a significant area of research, offering new perspectives and methodologies for neural network design and analysis. Central to this development is the consideration of deep neural networks as discretizations of dynamical systems. This perspective was notably advanced by the introduction of Neural Ordinary Differential Equations [54]. Neural ODEs transform the concept of discrete network layers into a continuous process, interpreting data transformation as the evolution governed by differential equations. This continuous depth model allows for adaptive computation, enabling

the network to dynamically determine the number of function evaluations required for a given input, leading to potential improvements in efficiency and accuracy [55].

Additionally, viewing the training process of deep neural networks through the lens of ODEs provides valuable insights. The backpropagation algorithm used in training can be seen as solving an ODE where gradient flow governs the evolution of network parameters [56]. Moreover, the stability and robustness of deep neural networks can be analyzed through the properties of the underlying ODEs, providing insights into the effects of small perturbations on network outputs and enhancing adversarial robustness [57]. Integrating ODEs into deep learning offers a continuous and theoretically grounded framework, paving the way for the development of more flexible and efficient neural network architectures.

In the following, we discuss the connection between ODE and deep learning using three examples, namely residual neural network, recurrent neural network, and normalizing flow. Recall the definition of an ODE and its first-order Taylor expansion (as known as the Euler's method) as discussed in Eq. (2.4), these three types of neural networks can be considered as a democratization of an ODE.

- **ResNet as an ODE [54].** A Residual Network (ResNet) can be expressed as:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + f(\mathbf{x}_n, t).$$

  This formulation is strikingly similar to Euler's method. Here, $\mathbf{x}_n$ represents the input at layer n, and f represents the ResNet layer. In the continuous limit (as the number of layers increases and each layer's contribution becomes infinitesimally small), this can be seen as solving an ODE.

- **Recurrent Neural Networks as an ODE [58].** Recurrent Neural Networks (RNNs) process sequences by updating their hidden states at each time step. The hidden state

update can be written as:

$$\mathbf{h}_{t+1} = \sigma(\mathbf{W}_t\mathbf{h}_t + \mathbf{W}_t\mathbf{x}_{t+1} + \mathbf{b}),$$

where $\mathbf{h}_t$ is the hidden state at time t. This can be interpreted as a discrete approximation to the continuous ODE:

$$\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), \mathbf{x}_t).$$

Here, the function f defines the dynamics of the hidden state.

- **Normalizing flows as an ODE [54].** Normalizing flows are a class of generative models that provide a powerful way to model complex probability distributions. They achieve this by transforming a simple base distribution (e.g., Gaussian) into a more complex target distribution through a series of invertible and differentiable mappings. Normalizing flows can also be viewed through the lens of ODEs and deep learning. The idea is to interpret the transformations in a flow as the solution to an ODE. This leads to the concept of continuous normalizing flows (CNFs).

2.2 Background on Optimal Control Theory

The study of optimal control theory originates from the classical theory of the calculus of variations, beginning with the seminal work of Euler and Lagrange in the 1700s. These culminated in the so-called Lagrangian mechanics that reformulate Newtonian mechanics in terms of extremal principles. In a nutshell, the calculus of variations studies optimization over "curves", which one can picture as an infinite dimensional extension of traditional optimization problems. Optimal control theory is a nontrivial extension of the classical theory of calculus of variations in two main directions: to dynamical and non-smooth settings. This leads to two interrelated directions: the Pontryagin's maximum principle and the Hamilton-Jacobi-Bellman theory.

2.2.1 The Optimal Control Formulation

Consider the ordinary differential equation

$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t, \mathbf{u}_t, t), \quad t \in [t_0, t_1], \quad \mathbf{x}_{t_0} = \mathbf{x}_0. \tag{2.5}$$

Here $\mathbf{x}_t \in \mathbb{R}^d$ is the state, $\mathbf{u} : [t_0, t_1] \to \mathcal{U} \subset \mathbb{R}^m$ is the control, with $\mathcal{U}$ the control set. Let the following conditions on F hold:

- $F(\mathbf{x}, \mathbf{u}, t)$ is continuous in t and $\mathbf{u}$ for all $\mathbf{x}$.
- $F(\mathbf{x}, \mathbf{u}, t)$ is continuously differentiable in $\mathbf{x}$ for all t and $\mathbf{u}$.

These conditions are sufficient to ensure that the differential equation defined in Equation (2.5) is well-posed. Let the objective functionals have the following form

$$J(\mathbf{u}) = \int_{t_0}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t)dt + \Phi(\mathbf{x}_{t_1}, t_1), \tag{2.6}$$

where

- $\mathcal{L}$: $\mathbb{R}^d \times \mathcal{U} \times \mathbb{R} \to \mathbb{R}$ is called the running loss.
- Φ: $\mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ is called the terminal cost.

The Bolza problem of optimal control is stated as the following objective function,

$$\begin{aligned} \inf_{\mathbf{u}} J(\mathbf{u}) &= \int_{t_0}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t)dt + \Phi(\mathbf{x}_{t_1}, t_1), \\ \text{s.t. } \dot{\mathbf{x}}_t &= F(\mathbf{x}_t, \mathbf{u}_t, t), \quad t \in [t_0, t_1], \quad \mathbf{x}_{t_0} = \mathbf{x}_0. \end{aligned} \tag{2.7}$$

The case where $\Phi = 0$ (no terminal cost) is called a Lagrange problem, whereas the case with $\mathcal{L}$ (no running cost) is called a Mayer problem. In optimal control theory, the initial condition $\mathbf{x}_0$ and initial time t_0 are often considered to be fixed. However, the terminal time t_1 can either be fixed or it can vary. As with classical optimization problems, the primary object of

study is optimality conditions. One differentiates between necessary and sufficient conditions for optimality. The former asks what conditions must any local/global optimum satisfy, and the latter concerns a condition that is enough to guarantee optimality. The following sections will investigate each of these aspects.

2.2.2 Pontryagin's Maximum Principle

This section discusses Pontryagin's Maximum Principle (PMP) as a necessary condition for optimality in both continuous and discrete settings.

PMP in continuous setting: Consider the Bolza problem of optimal control defined in Equation (2.7),

$$\inf_{\mathbf{u}} J(\mathbf{u}) = \int_{t_0}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t)dt + \Phi(\mathbf{x}_{t_1}, t_1),$$
$$\text{s.t. } \dot{\mathbf{x}}_t = F(\mathbf{x}_t, \mathbf{u}_t, t), \;\; t \in [t_0, t_1], \;\; \mathbf{x}_{t_0} = \mathbf{x}_0.$$

To begin with, let the Hamiltonian $H : \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^d \times \mathcal{U} \to \mathbb{R}$ be defined as

$$H(t, \mathbf{x}_t, \mathbf{p}_t, \mathbf{u}_t) := \mathbf{p}^\top F(\mathbf{x}_t, \mathbf{u}_t, t) - \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t), \tag{2.8}$$

A control $\mathbf{u}$ is admissable if $\mathbf{u}_t \in \mathcal{U}$ for all $t \in [t_0, t_1]$.

Theorem 2.2.1 *Let $\mathbf{u}^*$ be a bounded, measurable and admissible control that optimizes the objective function defined in (2.7), and $\mathbf{x}^*$ be its corresponding state trajectory. Then, there exists an absolutely continuous process $\mathbf{p} = \{\mathbf{p}_t : t \in [t_0, t_1]\}$ such that*

$$\dot{\mathbf{x}}_t = \nabla_{\mathbf{p}} H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t^*), \quad \mathbf{x}_0^* = \mathbf{x}_0, \tag{2.9}$$

$$\dot{\mathbf{p}}_t = -\nabla_{\mathbf{x}} H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t^*), \quad \mathbf{p}_{t_1}^* = -\nabla_{\mathbf{x}} \Phi(\mathbf{x}_{t_1}^*), \tag{2.10}$$

$$H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t^*) \geq H(t, x_t^*, \mathbf{p}_t^*, \mathbf{u}_t), \quad \forall \mathbf{u}_t \in \mathcal{U} \text{ and a.e. } t \in [t_0, t_1]. \tag{2.11}$$

The equation (2.9) is called the state equation, and it is simply

$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t, \mathbf{u}_t, t),$$

and it describes the evolution of the state under optimal control. The equation (2.10) is called the co-state equation, with $\mathbf{p}^*$ being the co-state. The role of the co-state equation is to propagate back the optimality condition and it is the adjoint of the variational equation. The maximization condition in Equation (2.11) is the heart of the maximum principle. It says that an optimal control must globally maximize the Hamiltonian. One can regard this as a nontrivial generalization of the Euler-Lagrange equations to handle strong extrema, as well as a generalization of the KKT conditions to nonsmooth settings.

PMP in discrete setting: In a discrete setting, the objective function of the optimal control problem is

$$\begin{aligned} \inf_{\mathbf{u}} J(\mathbf{u}) &= \sum_{t=t_0}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t) + \Phi(\mathbf{x}_{t_1}, t_1), \\ \text{s.t. } \mathbf{x}_{t+1} &= F(\mathbf{x}_t, \mathbf{u}_t, t), \;\; t \in [t_0, t_1], \;\; \mathbf{x}_{t_0} = \mathbf{x}_0, \end{aligned} \tag{2.12}$$

with a discrete dynamic system

$$\mathbf{x}_{t+1} = F(\mathbf{x}_t, \mathbf{u}_t, t), \;\; t \in [t_0, t_1], \;\; \mathbf{x}_{t_0} = \mathbf{x}_0. \tag{2.13}$$

Theorem 2.2.2 *Let F and Φ be sufficiently smooth in $\mathbf{x}$. Assume further that for each t and $\mathbf{x} \in \mathbb{R}^d$, the sets $\{F(\mathbf{x}_t, \mathbf{u}_t, t) : \mathbf{u}_t \in \mathcal{U}\}$ and $\{\mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t) : \mathbf{u}_t \in \mathcal{U}\}$ are convex. Then, there*

exists co-state processes $\mathbf{p}^* = \{\mathbf{p}_t^* : t = [t_0, t_1]\}$*, such that following holds for* $t \in [t_0, t_1]$*:*

$$\mathbf{x}_{t+1} = \nabla_{\mathbf{p}} H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t^*), \quad \mathbf{x}_0^* = \mathbf{x}_0, \tag{2.14}$$

$$\mathbf{p}_t = -\nabla_{\mathbf{x}} H(t, \mathbf{x}_t^*, \mathbf{p}_{t+1}^*, \mathbf{u}_t^*), \quad \mathbf{p}_{t_1}^* = -\nabla_{\mathbf{x}} \Phi(\mathbf{x}_{t_1}^*), \tag{2.15}$$

$$H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t^*) \geq H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \mathbf{u}_t), \quad \forall \mathbf{u}_t \in \mathcal{U} \text{ and a.e. } t \in [t_0, t_1]. \tag{2.16}$$

2.2.3 Hamilton-Jacobi-Bellman Equations

As a key alternative to the maximum principle, the Hamilton-Jacobi-Bellman Equations establish necessary and sufficient conditions for optimality for optimal control problems. This presents another approach to optimal control theory that is important in its own right, as it depends on the very general idea of dynamic programming.

The dynamic programming principle: Recall the Bolza problem with a fixed end time,

$$\begin{aligned} \inf_{\mathbf{u}} J(\mathbf{u}) &= \int_{t_0}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t) dt + \Phi(\mathbf{x}_{t_1}), \\ \text{s.t. } \dot{\mathbf{x}}_t &= F(\mathbf{x}_t, \mathbf{u}_t, t), \ \ t \in [t_0, t_1], \ \ \mathbf{x}_{t_0} = \mathbf{x}_0. \end{aligned} \tag{2.17}$$

Following the idea of dynamic programming, this problem is embedded in a bigger class of problems,

$$\begin{aligned} V(s, \mathbf{z}) &:= \inf_{\mathbf{u}} \int_{s}^{t_1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t) dt + \Phi(\mathbf{x}_{t_1}), \\ \text{s.t. } \dot{\mathbf{x}}_t &= F(\mathbf{x}_t, \mathbf{u}_t, t), \ \ t \in [s, t_1], \ \ \mathbf{x}_s = \mathbf{z}. \end{aligned} \tag{2.18}$$

The function $V : [t_0, t_1] \times \mathbb{R}^d \to \mathbb{R}$ is called the value function, it is the minimum cost attainable starting from the initial condition $\mathbf{z}$ at time s. Observe that $V(t_0, \mathbf{x}_0)$ is the optimal cost of Equation (2.17).

The dynamic programming principle concerns the value function for the optimal control

problem.

Theorem 2.2.3 *For every $s, \tau \in [t_0, t_1]$, $s \leq \tau$, and $\mathbf{z} \in \mathbb{R}^d$, the following holds,*

$$V(s, \mathbf{z}) = \inf_{\mathbf{u}} \Big[\int_s^\tau \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t)dt + V(\tau, \mathbf{x}_\tau) \Big],$$
$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t, \mathbf{u}_t, t), \quad t \in [s, \tau], \quad \mathbf{x}_s = \mathbf{z}. \tag{2.19}$$

The meaning of the dynamic programming principle is that the optimization problem defining $V(s, \mathbf{z})$ can be split into two parts:

- Solving the optimization problem on $[\tau, t_1]$ with the usual running cost $\mathcal{L}$ and terminal cost Φ, but for all initial conditions $\mathbf{z}' \in \mathbb{R}^d$. This gives the value function $V(\tau, \cdot)$.
- Solving the optimization problem on $[s, \tau]$ with running loss $\mathcal{L}$ and terminal cost $V(\tau, \cdot)$ given by the step before.

Hamilton-Jacobi-Bellman Equations: This section derives the key result from the dynamic programming approach to optimal control problems, which establishes connections with partial differential equations, in particular the Hamilton-Jacobi equations. The basic motivation here is to derive an infinitesimal version of the dynamic programming principle. To this end, from the extensive use of Taylor expansions by assuming that $\tau = s + \triangle s$ with $\triangle s << 1$ in Equation (2.19), giving the infinitesimal dynamic programming principle

$$V(s, \mathbf{z}) = \inf_{\mathbf{u}} \Big[\int_s^{s+\triangle s} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, t)dt + V(s + \triangle s, \mathbf{x}_{s+\triangle s}) \Big],$$
$$\dot{\mathbf{x}}_t = F(\mathbf{x}_t, \mathbf{u}_t, t), \quad t \in [s, s + \triangle s], \quad \mathbf{x}_s = \mathbf{z}. \tag{2.20}$$

Applying Taylor's expansion on the ODE in Equation (2.20),

$$\mathbf{x}_{s+\triangle s} = \mathbf{z} + \int_s^{s+\triangle s} F(\mathbf{x}_t, \mathbf{u}_t, t)dt = \mathbf{z} + F(\mathbf{x}_t, \mathbf{u}_t, t)\triangle s + \mathcal{O}(\triangle s). \tag{2.21}$$

Furthermore, assuming that V is sufficiently regular,

$$V(s+\triangle s, \mathbf{x}_{(s+\triangle s)}) = V(s,\mathbf{z}) + \partial_s V(s,\mathbf{z})\triangle s + [\nabla_{\mathbf{z}} V(s,\mathbf{z})]^\top F(\mathbf{x}_t,\mathbf{u}_t,t)\triangle s + \mathcal{O}(\triangle s). \quad (2.22)$$

Similarly, the running cost can be expanded

$$\int_s^{s+\triangle s} \mathcal{L}(\mathbf{x}_t,\mathbf{u}_t,t)dt = \mathcal{L}(\mathbf{z},\mathbf{u}_t,t)\triangle s + \mathcal{O}(\triangle s). \quad (2.23)$$

Combining Equations (2.20), (2.22) and (2.23),

$$V(s,\mathbf{z}) = \inf_{\mathbf{u}}\{\mathcal{L}(\mathbf{z},\mathbf{u}_t,t)\triangle s + V(s,\mathbf{z}) + \partial_s V(s,\mathbf{z}) + [\nabla_{\mathbf{z}} V(s,\mathbf{z})]^\top F(\mathbf{x}_t,\mathbf{u}_t,t)\triangle s + \mathcal{O}(\triangle s)\}. \quad (2.24)$$

Canceling the term $V(s,\mathbf{z})$ on both sides and taking the limit $\triangle s \to 0$,

$$\partial_s V(s,\mathbf{z}) + \inf_{\mathbf{u}}\{\mathcal{L}(\mathbf{z},\mathbf{u}_t,t) + [\nabla_{\mathbf{z}} V(s,\mathbf{z})]^\top F(\mathbf{x}_t,\mathbf{u}_t,t)\} = 0. \quad (2.25)$$

This is known as the Hamilton-Jacobi-Bellman (HJB) equation for the value function. It remains to specify the boundary conditions. One can quickly observe that at time $s = t_1$, $V(t_1,\mathbf{z}) = \Phi(\mathbf{z})$.

Theorem 2.2.4 *Let $V : [t_0,t_1]\times\mathbb{R}^d \to \mathbb{R}$ be the value function defined by Equation* (2.20)*. Then V is the unique viscosity solution of the Hamilton-Jacobi-Bellman equation*

$$\partial_t V(t,\mathbf{x}) + \inf_{\mathbf{u}}\{\mathcal{L}(\mathbf{z},\mathbf{u}_t,t) + [\nabla_{\mathbf{z}} V(s,\mathbf{z}]^\top F(\mathbf{x}_t,\mathbf{u}_t,t) = 0\}, \quad (t,\mathbf{x}) \in [t_0,t_1]\times\mathbb{R}^d,$$
$$V(t_1,\mathbf{x}) = \Phi(\mathbf{x}).$$

2.2.4 Connection between Optimal Control and Deep Neural Networks

The connection between optimal control theory and deep neural networks has become an intriguing area of research, leading to innovative approaches for training and understanding

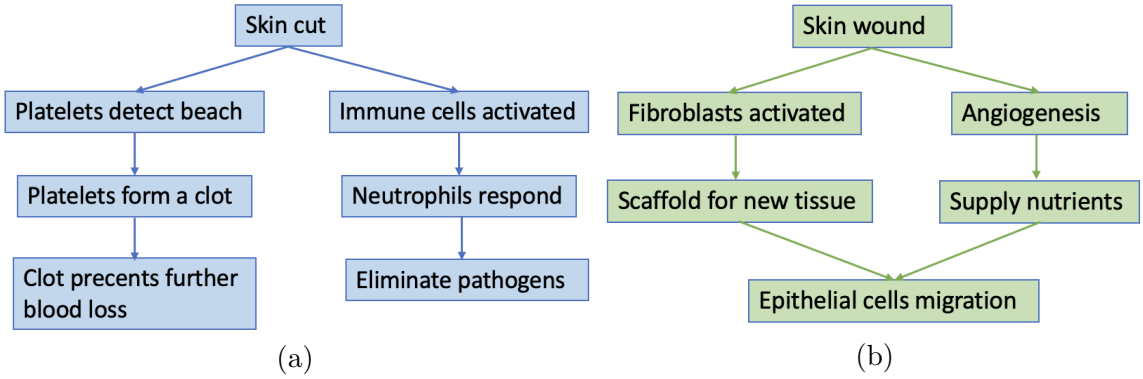


Figure 2.1: (a): Self-healing process in error detection (b): self-healing process in correction actions.

neural networks. One significant development in this area is the interpretation of the training of deep neural networks as a dynamic system governed by differential equations, where the objective is to find an optimal trajectory that minimizes a loss function. This approach aligns closely with the principles of optimal control, where the goal is to find a control policy that drives the system towards a desired state while minimizing a cost function. Techniques from optimal control, such as the Pontryagin's Maximum Principle, have been applied to derive efficient training algorithms for neural networks. These methods provide theoretical guarantees on convergence and can lead to more stable and robust training procedures [59].

2.3 Background on Self-Healing

2.3.1 Self-Healing from Biology

In this work, we define "self-healing" as the ability to autonomously detect and correct errors within a neural network. The idea of self-healing comes from how the human body heals itself and stays balanced. The human body is equipped with a sophisticated set of mechanisms that enable it to repair and regenerate tissues, ensuring survival and functionality. These systems work together through a mix of cells, molecules, and other factors.

Error detection: Error detection in the human body is a crucial aspect of its self-healing processes, relying on monitoring mechanisms and specific metrics to identify and quantify injuries or infections. For example, as shown in Figure 2.1 (a), when the skin is cut, the body immediately initiates a response. Platelets detect the breach and form a clot to prevent further blood loss, acting as the first line of defense. Simultaneously, immune cells such as neutrophils and macrophages are activated to identify and eliminate pathogens and debris at the injury site. This detection and response system ensures that the body can quickly address and mitigate damage. The body's ability to monitor for signs of injury or infection and to measure the severity of these events is vital for initiating the appropriate healing processes. This intricate system of detection and response enables the body to maintain homeostasis and promote recovery, ensuring overall health and functionality.

Corrective actions: Corrective actions in the human body's self-healing processes are vital for repairing and regenerating damaged tissues. Once an injury is detected, the body initiates a series of coordinated responses to address the damage. For example, as shown in Figure 2.1 (b), in the case of a skin wound, after the initial clot formation and immune response, the body moves into the proliferation phase. During this phase, fibroblasts are activated to produce collagen and other extracellular matrix components, creating a scaffold for new tissue. Concurrently, new blood vessels form through angiogenesis to supply nutrients and oxygen to the healing area. Epithelial cells migrate across the wound bed to re-establish the skin barrier. Throughout this process, growth factors and cytokines guide cell behavior, promoting cell division, migration, and differentiation. Finally, in the remodeling phase, the newly formed tissue is strengthened and restructured to restore its original function. This series of corrective actions ensures that the body can effectively repair itself, maintaining its integrity and health.

2.3.2 Self-Healing in Integral Circuit Design

The concept is extensively explored within the Integrated Circuit (IC) design field to amend faults stemming from nano-scale discrepancies during the fabrication of analog, mixed-signal,

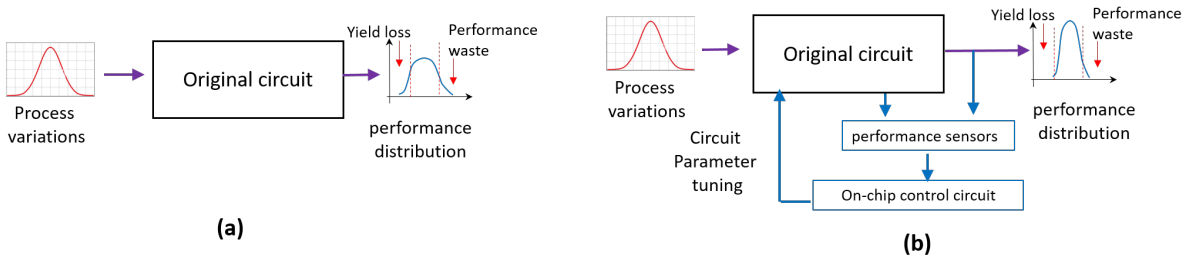


Figure 2.2: (a) Standard circuit design without self-healing. The result can have significant yield loss and performance waste; (b) Self-healing circuit with on-chip performance monitor and control, resulting higher yield and less performance waste.

and digital systems [60, 61, 62, 63, 64, 65, 66, 67]. Such fabrication challenges arise from the difficulty in precisely managing the geometric and material specifics, leading to performance degradation or outright failure of numerous circuit chips. To address these issues, two main strategies are employed: yield optimization and self-healing. Yield optimization [68, 69, 70, 71], akin to adversarial training, involves selecting ideal circuit parameters during the design stage to reduce failure risks based on a defined probability density function of process variations. Conversely, self-healing aims to correct circuit errors post-design, without prior knowledge of process variation distributions. Similarly, in developing reliable neural network models, predicting the nature of perturbations that might occur once deployed is challenging. As such, employing post-training adjustments presents a viable solution for addressing many potential errors that adversarial training alone cannot resolve.

Within the scope of self-healing implementations, the application of closed-loop control systems have demonstrated considerable effectiveness in chip design [60, 61]. This approach is illustrated in Figure 2.2, where Figure 2.2 (a) depicts a typical circuit that, despite undergoing yield optimization to enhance its success rate amidst uncertainties, may still encounter yield reduction or efficiency loss due to unforeseeable variations in the fabrication process. As depicted in Figure 2.2 (b), to mitigate these challenges, the integration of either global or local on-chip sensors is proposed to continuously assess key performance indicators. Additionally, the inclusion of a control circuit on the chip, which adjusts specific parameters such as bias currents, supply voltages, or variable capacitors, aims to rectify identified errors. Consequently, this

adjustment ensures that the circuit's output performance aligns more closely with the desired region, thereby improving yield and minimizing inefficiency.

2.3.3 Self-Healing in Other Engineering Domains

Besides integral circuits, the concept of self-healing can be applied to many other engineering domains, including but not limited to

- **Materials Science:** In materials science, the development of self-healing materials has revolutionized how we think about durability and longevity [72]. These materials, particularly polymers and composites, are engineered to autonomously repair damage such as cracks or fractures, extending their useful life and reducing the need for maintenance. The mechanisms of self-healing in these materials vary; some involve microcapsules filled with healing agents that break open when a crack forms, releasing the agent to fill and bond the crack. Others rely on intrinsic properties of the material itself, such as thermoplastic polymers that can re-melt and re-bond under heat.

- **Self-healing in aerospace engineering:** Self-healing polymers used in aerospace engineering are advanced materials designed to autonomously repair damage, such as cracks or abrasions, without the need for external intervention [73]. These polymers often incorporate microcapsules filled with healing agents or are made with reversible chemical bonds that can break and reform in response to stimuli like heat or light. When damage occurs, the healing agents are released or the bonds reconnect, effectively restoring the material's integrity. This self-repair capability significantly enhances the safety, reliability, and longevity of aerospace components, reducing maintenance costs and improving overall structural performance.

In this work, we seek to extend this concept to the development of a self-healing neural network, leveraging the principles outlined above.

Chapter 3

Self-Healing Robust Neural Networks via Closed-Loop Control

This chapter investigates the robustness issue of neural networks when imperceptible perturbations are applied to input data. We propose a novel concept inspired by the self-healing mechanisms of biological immune systems. In psychology, self-healing often refers to the recovery of a patient from a psychological disturbance guided by instinct only. In physiology, the most well-known self-healing mechanism is probably the human immune system: B cells and T cells can work together to identify and kill many external attackers (e.g., bacteria) to maintain the health of the human body [74]. This idea has been applied in semiconductor chip design, where self-healing integrated circuits can automatically detect and fix the errors caused by imperfect nano-scale fabrication, noise, or electromagnetic interference [60, 61, 62, 63, 64, 65, 66, 67]. In machine learning, a similar self-healing process could autonomously address and mitigate issues within the system. We realize this proposal via a closed-loop control method. Significantly differing from existing methods, this self-healing process does not depend on perturbation information; instead, it autonomously identifies and corrects errors within the neural network.

Contribution Summary. The specific contributions of this chapter are summarized below:

- **Development of a closed-loop control formulation for post-training self-healing with margin-based analysis:** We introduce a closed-loop control approach aimed at improving the robustness of neural networks in the post-training stage against a wide

array of unexpected perturbations. This self-healing approach consists of two main components: embedding functions at both input and hidden layers to detect the potential errors, and a control process to adjust the states to fix or mitigate these errors before making a prediction. We investigate the working principle of the proposed control loss function and reveal that it can modify the decision boundary and increase the margin of a classifier. Based on this, we introduce three types of margins that are of relevance, which involve an interplay of the ground truth classifier, the trained classifier, and the data manifold. Armed with this understanding, we build embedding functions that exploit both model information and data structure.

- **Fast numerical solver for the control objective function:** Addressing the computational demands of implementing the self-healing neural network through closed-loop control, we employ Pontryagin's Maximum Principle and the method of successive approximations. This numerical solver allows us to handle both deep and wide neural networks.

- **Comprehensive theoretical error analysis:** Our work includes a thorough error analysis of the proposed framework in its broadest scope, considering nonlinear dynamics alongside nonlinear embedding manifolds. The theoretical setup aligns with our algorithm implementation without simplification.

- **Empirical validation across multiple datasets:** We conduct extensive experiments on two standard and one challenging dataset. These experiments validate the effectiveness of our closed-loop control approach for self-healing in consistently improving the robustness of neural networks against diverse perturbations.

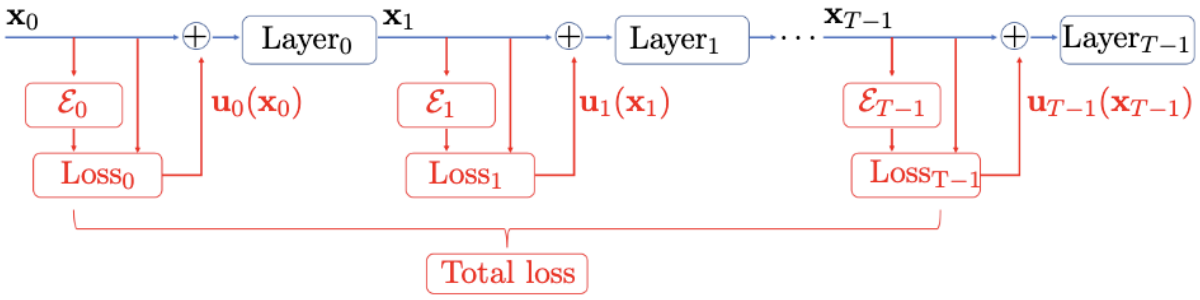


Figure 3.1: The structures of feed-forward neural network (blue) and the proposed closed-loop control method (red).

3.1 Self-Healing Robust Neural Network via Closed-Loop Control

We implement post-training self-healing via closed-loop control to achieve better robustness of neural networks. Figure 3.1 illustrates the proposed control approach. At every layer, an embedding function $\mathcal{E}_t(\cdot)$ is used to monitor the performance of a hidden state and generate a loss. The control signal $\mathbf{u}_t(\mathbf{x}_t)$ is computed by optimizing the total loss which is a summation of all running losses. The generated controls are then applied to adjust the states, such that possible errors can be eliminated or mitigated before they propagate to the output label. In the following, we abbreviate the feedback control signal to $\mathbf{u}_t$ that is generated based on the state $\mathbf{x}_t$.

Remark 3.1.1 *It's important to clarify that the neural network design shown in Figure 3.1 is not a form of open-loop control. In this framework,* $\mathbf{x}_0$ *represents the neural network's initial input conditions, while the excitation input signal is denoted by* $\mathbf{u}_t$ *(which is zero during standard forward propagation). The forward signal path is from* $\mathbf{u}_t$ *to internal states* $\mathbf{x}_t$ *and then to the output label* $\mathbf{y}$*. The path from* $\mathbf{x}_t$ *to the embedding function* $\mathcal{E}_t(\mathbf{x}_t)$ *and then to the control signal* $\mathbf{u}_t$ *forms a feedback and closes the whole loop.*

Due to the closed-loop structure, the forward propagation of the proposed self-healing neural network at layer t can be written as $\mathbf{x}_{t+1} = F_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t)$, in which the control $\mathbf{u}_t$ is added to the

state $\mathbf{x}_t$. Compared with standard neural networks, the proposed network needs to compute the control signals $\overline{\mathbf{u}} = \{\mathbf{u}_t\}_{t=0}^{T-1}$ during inference by solving an optimal control problem:

$$\min_{\overline{\mathbf{u}}} \mathbb{E}_{(\mathbf{x}_0,\mathbf{y})\sim\mathcal{D}} \left[J(\mathbf{x}_0, \mathbf{y}, \overline{\mathbf{u}})\right] := \min_{\overline{\mathbf{u}}} \mathbb{E}_{(\mathbf{x}_0,\mathbf{y})\sim\mathcal{D}} \ \Phi(\mathbf{x}_T, \mathbf{y}) + \sum_{t=0}^{T-1} \mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, \cdot),$$
$$\text{s.t. } \mathbf{x}_{t+1} = F_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t), \ t = 0, \cdots, T-1, \tag{3.1}$$

where Φ is the terminal loss, and $\mathcal{L}$ denotes a running loss that possibly depends on state $\mathbf{x}_t$, control $\mathbf{u}_t$ and some external functions.

To improve the robustness through self-healing closed-loop control, Section 3.2 discusses the principles for formulating the control objective function defined in Equation (3.1) to ensure that the resulting controls can effectively correct errors. Section 3.3 constructs an efficient solver to solve the control objective function defined in Equation (3.1). Section 3.4 establishes theoretical performance guarantees for the self-healing neural network.

3.2 Design of Self-Healing via Optimal Control

In Section 3.2.1, we introduce a control objective function aimed at enhancing the self-healing robustness of neural networks for classification tasks. Following this, Section 3.2.2 illustrates how this control objective function effectively increases the classification margin around the decision boundary.

3.2.1 Towards Better Robustness: Control Loss via Manifold Projection

In general, the control objective function defined in Equation (3.1) should have two components: a terminal loss and a running loss:

- In traditional optimal control, the terminal loss $\Phi(\mathbf{x}_T, \mathbf{y})$ can be a distance measurement between the terminal state of the underlying trajectory and some destination set given

beforehand. In supervised learning, this corresponds to controlling the underlying hidden states such that the terminal state $\mathbf{x}_T$ (or some transformation of it) matches the true label. This is impractical for general machine learning applications since the true label $\mathbf{y}$ is unknown during inference. Therefore, we ignore the terminal loss by setting it as zero.

- When considering a deep neural network as a discretization of a continuous dynamic system, the state trajectory (all input and hidden states) governed by this continuous transformation forms a high-dimensional structure embedded in the ambient state space. The set of state trajectories that leads to ideal model performance, in the discretized analogy, can be represented as a sequence of embedding manifolds $\{\mathcal{M}_t\}_{t=0}^{T-1}$. The embedding manifold is defined as $\mathcal{M}_t = f_t^{-1}(\mathbf{0})$ for a submersion[1] $f(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$, where we assume that all data samples lie in $\mathbb{R}^d$ and there exists a r-dimensional embedding manifold to encode all data. We can track a trajectory during neural network inference and enforce it onto the desired manifold $\mathcal{M}_t$ to improve model performance. This motivates us to design the running loss of Equation (3.1) as follows,

$$\mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, f_t(\cdot)) := \frac{1}{2} \cdot \|f_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2} \cdot \|\mathbf{u}_t\|_2^2. \tag{3.2}$$

  The submersion satisfies $\|f_t(\mathbf{x}_t)\|_2 = \|\mathcal{E}_t(\mathbf{x}_t) - \mathbf{x}_t\|_2$ and it measures the distance between a state $\mathbf{x}$ to the embedding manifold $\mathcal{M}_t$, $f_t(\mathbf{x}) = \mathbf{0}$ if $\mathbf{x} \in \mathcal{M}_t$. This can be understood based on the "manifold hypothesis" [75], which assumes that real-world high-dimensional data (represented as vectors in $\mathbb{R}^d$) generally lie in a low-dimensional manifold $\mathcal{M} \subset \mathbb{R}^d$. The first term in Equation (3.2) serves as a "performance monitor" in self-healing: it measures the discrepancy between the state variable $\mathbf{x}_t$ and the desired manifold $\mathcal{M}_t$. The regularization term with a hyper-parameter c prevents using large controls,

[1] a submersion is a differentiable map between differentiable manifolds whose differential is everywhere surjective

- The performance monitor can be realized by a manifold projection $\mathcal{E}_t(\cdot)$,

$$\mathcal{E}_t(\mathcal{M}_t, \mathbf{x}_t) := \arg\min_{\mathbf{z}\in\mathcal{M}_t} \frac{1}{2}\|\mathbf{x}_t - \mathbf{z}\|_2^2. \tag{3.3}$$

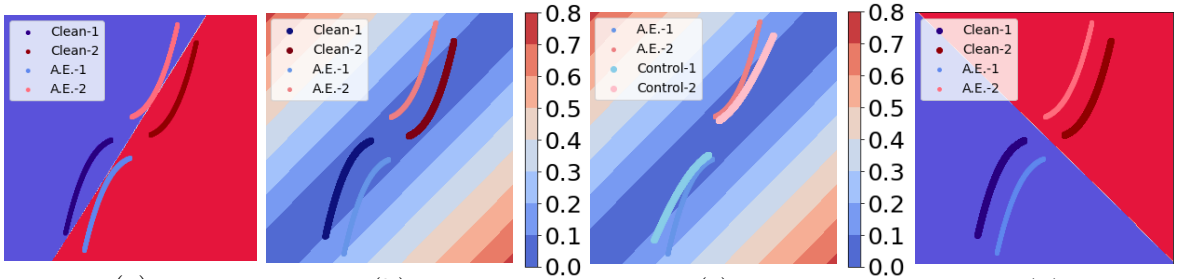
  The manifold projection can be considered as a constrained optimization. Given that $\mathcal{M}_t$ is a compact set, the solution of Equation (3.3) always exists. In practice, the manifold projection is realized as an auto-encoder due to its simplicity and generality. Specifically, an encoder embeds a state snapshot into a lower-dimensional space, and then a decoder reconstructs this embedded data back to the ambient state space. The auto-encoder can be obtained by minimizing the reconstruction loss on a given clean dataset,

$$\begin{aligned}\mathcal{E}_t^*(\mathcal{M}_t, \cdot) = \arg\min_{\mathcal{E}_t} \frac{1}{N}\sum_{i=1}^{N} \underbrace{\mathrm{CE}(\mathbf{x}_{i,T}, \mathbf{y}_i)}_{\text{model information}} + \underbrace{\|\mathcal{E}_t(\mathcal{M}_t, \mathbf{x}_{i,t}) - \mathbf{x}_{i,t}\|_2^2}_{\text{data information}}, \\ \text{s.t. } \mathbf{x}_{i,t+1} = F_t(\mathbf{x}_{i,t}, \mathbf{u}_{i,t}, \boldsymbol{\theta}_t),\ \mathbf{u}_{i,t} = \mathcal{E}_t(\mathbf{x}_{i,t}) - \mathbf{x}_{i,t},\end{aligned} \tag{3.4}$$

  where $\mathrm{CE}(\cdot,\cdot)$ denotes cross-entropy loss function, $\boldsymbol{\theta}_t$ is the model parameter at the t^{th} layer. The objective function Equation (3.4) defines an attack-agnostic setting, where only clean data and model information are accessible to the control system. Furthermore, we do not attempt to recover the underlying data manifold. Instead, we find a low-dimensional manifold that is defined by one having a submersion using the encoder-decoder function, and this estimated low-dimensional manifold approximately contains the true data manifold. If one is only concerned with approximating the true data manifold, Equation (3.4) can be modified to only optimize the data information [76].

Considering the zero terminal loss and non-zero running loss, the overall control objective function for self-healing can be designed as below,

$$\begin{aligned}\min_{\bar{\mathbf{u}}} \mathbb{E}_{(\mathbf{x}_0,\mathbf{y})\sim\mathcal{D}} \sum_{t=0}^{T-1} \|f_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2, \\ \text{s.t. } \mathbf{x}_{t+1} = F_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\theta}_t),\ t = 0, \cdots, T-1.\end{aligned} \tag{3.5}$$



(a) (b) (c) (d)

Figure 3.2: (a): Clean-1 and Clean-2 represent clean data of class 1 and 2, respectively. A.E.-1 and A.E.-2 are their adversarially perturbed counterparts. (b): the reconstruction loss field. (c): the controlled counterpart. (d): Manifold projection modifies decision boundary.

In neural network inference, the resulting control signals will help to attract the (possibly perturbed) trajectory towards the embedding manifolds.

3.2.2 A Margin-Based Analysis On the Running Loss

The discussion centers on the efficacy of the running loss as defined in Equation (3.2). To make the problem more manageable, we examine a specific instance of the control objective function shown in Equation (3.5). In this case, control is applied directly to the input (setting $T = 1$) and the applied control incurs no penalty ($c = 0$). This approach is applicable to any t-th layer of analysis, where $\mathbf{x}_{t-1}$ is treated as the input data. In this simplified setting, by choosing $\mathcal{M}$ as the embedding manifold in $\mathbb{R}^d$, the optimal control results in the solution of the constrained optimization in Equation (3.3).

Manifold projection enlarges decision boundary. For any given input $\tilde{\mathbf{x}}$, an optimal control process solves the constrained optimization defined in Equation (3.3) by reconstructing the nearest counterpart $\mathbf{x} \in \mathcal{M}$. This seemingly *adaptive* control process essentially forms some *deterministic* decision boundaries that enlarge the margin of a given classifier. In general, an accurate classifier can have a small "classifier margin" measured by an ℓ_p norm, i.e. the minimal perturbation in $\mathbb{R}^d$ required to change the model prediction label. This small margin can be easily exploited by adversarial attacks, such as PGD [23]. We illustrate these phenomena

with a numerical example. Figure 3.2 shows a binary classification problem in $\mathbb{R}^2$, where blue and red regions represent the classification predictions (their joint line represents the decision boundary of the underlying classifier). In Figure 3.2 (a), the given classifier has accuracies of 100% and 0% on clean data and against adversarial examples respectively. Figure 3.2 (b) shows the reconstruction loss field, computed by $\|(\mathbf{I}-\mathbf{P})\mathbf{x}\|_2^2, \forall \mathbf{x} \in \mathbb{R}^2$, where $\mathbf{P}$ is the ℓ_2 orthogonal projection onto the 1-d embedding subspace $\mathcal{M}$. The estimated embedding subspace $\mathcal{M}$ is represented as the reconstruction loss being less than 0.1. As expected, clean data samples are located in the low loss regions, and adversarial examples fall out of $\mathcal{M}$ and have larger reconstruction losses. In Figure 3.2 (c), our control process adjusts adversarially perturbed data samples towards the embedding subspace $\mathcal{M}$, and the classifier predicts those with 100% accuracy. Essentially, the manifold projection forces those adjacent out-of-manifold samples to have the same prediction as the clean data in the manifold, and the margin of the decision boundary has been increased as shown in Figure 3.2 (d).

In this simplified linear case, the embedding manifold $\mathcal{M}$ is the 1-D linear subspace highlighted as the darkest blue in Figure 3.2 (b) (c). Specifically, any data point in this subspace incurs zero reconstruction loss. Therefore, the constrained optimization problem in Equation (3.3) is the orthogonal projection onto a linear subspace $\mathcal{M}$, The manifold projection reduces the pre-image of a classifier $F(\cdot)$ from $\mathbb{R}^2 \mapsto \mathbb{R}^1$. Given a data point $\mathbf{x}$ sampled from this linear subspace, any out-of-manifold data $\tilde{\mathbf{x}}$ satisfies $\|\mathbf{P}\tilde{\mathbf{x}}-\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{x}}-\mathbf{x}\|_2^2$. Consequently, the margin of $F(\cdot)$ is enlarged.

A margin-based analysis on the manifold projection. Now we formally provide two definitions for margins related to classification problems. Specifically, we consider a classification dataset $\mathcal{D}$ belonging to the ground-truth manifold $\mathcal{M}^*$, $\mathcal{D} \subset \mathcal{M}^*$, this enables the formal definitions of different types of margins.

- **Manifold margin:** We define $\mathcal{R}_{\mathcal{M}}$ as the geodesics

$$\mathcal{R}_{\mathcal{M}}(\mathbf{a}, \mathbf{b}) := \inf_{\gamma \in \Gamma_{\mathcal{M}}(\mathbf{a},\mathbf{b}),} \int_0^1 \sqrt{\langle \gamma'(t), \gamma'(t) \rangle_{\gamma(t)}} dt,$$

  where $\gamma \in \Gamma_{\mathcal{M}}(\mathbf{a}, \mathbf{b})$ is a continuously differentiable curve $\gamma : [0, 1] \to \mathcal{M}$ such that $\gamma(0) = \mathbf{a}$ and $\gamma(1) = \mathbf{b}$. Here, $\langle , \rangle_p$ is the positive definite inner product on the tangent space $\mathcal{T}_p\mathcal{M}$ at any point p on the manifold $\mathcal{M}$. In other words, the distance $\mathcal{R}_{\mathcal{M}}(\mathbf{a}, \mathbf{b})$ between two points $\mathbf{a}$ and $\mathbf{b}$ of $\mathcal{M}$ is defined as the length of the shortest path connecting them. Given a manifold $\mathcal{M}$ and classifier $F(\cdot)$, the manifold margin $d_{\mathcal{M}}(F(\cdot))$ is defined as the shortest distance along $\mathcal{M}$ such that an instance of one class transforms to another.

$$d_{\mathcal{M}}(F(\cdot)) := \frac{1}{2} \inf_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}} \mathcal{R}_{\mathcal{M}}(\mathbf{x}_1, \mathbf{x}_2), \quad \text{s.t. } F(\mathbf{x}_1) \neq F(\mathbf{x}_2). \tag{3.6}$$

- **Euclidean margin:** In practice, data perturbations are any perturbations of a small Euclidean distance (or any equivalent norm). The classifier margin $d_e(F(\cdot))$ is the smallest magnitude of a perturbation in $\mathbb{R}^d$ that causes the change of output predictions.

$$d_e(F(\cdot)) := \inf_{\mathbf{x} \in \mathcal{D}} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \|\boldsymbol{\delta}\|_2, \quad \text{s.t. } F(\mathbf{x}) \neq F(\mathbf{x} + \boldsymbol{\delta}). \tag{3.7}$$

In addition, we introduce the ground-truth margin and manifold projection margin from the definitions of manifold and Euclidean margins, respectively.

- **Ground-truth margin:** For the ground-truth manifold $\mathcal{M}^*$ and ground-truth classifier $F^*(\cdot)$ (population risk minimizer), the ground-truth margin $d_{\mathcal{M}^*}(F^*(\cdot))$ [according to Equation (3.6)] is the largest classification margin.

- **Manifold projection margin:** The manifold projection Equation (3.3) modifies a classifier from $F(\cdot)$ to $F \circ \mathcal{E}(\mathcal{M}, \cdot)$. Therefore, its robustness depends on the "manifold

projection margin" [according to Equation (3.7)] as

$$d_e(F \circ \mathcal{E}(\cdot)) := \inf_{\mathbf{x} \in \mathcal{D}} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \|\boldsymbol{\delta}\|_2, \quad \text{s.t. } F(\mathcal{E}(\mathbf{x})) \neq F(\mathcal{E}(\mathbf{x} + \boldsymbol{\delta})).$$

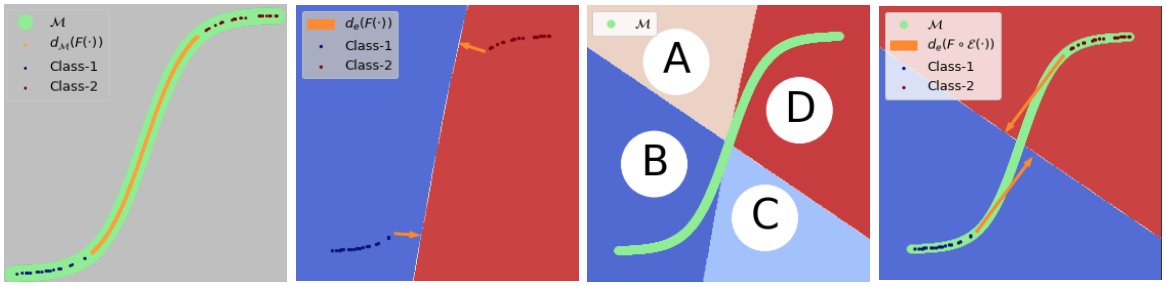
A manifold projection essentially constraints the data space $\mathbb{R}^d$ into a smaller subset according to the embedding manifold $\mathcal{M} \subset \mathbb{R}^d$.

In $\mathbb{R}^d$, a binary linear classifier forms a $(d-1)$-dimensional hyperplane that partitions $\mathbb{R}^d$ into two subsets. Let the range of $\mathbf{V} \in \mathbb{R}^{d \times (d-1)}$ be this hyperplane, $\hat{\mathbf{n}}$ as a d-dimensional normal vector such that $\mathbf{V}^T \hat{\mathbf{n}} = \mathbf{0}$. In general, a linear classifier with a random decision boundary can be defined as setting the normal vector $\hat{\mathbf{n}} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I})$. In this simplified linear setting, the following proposition provides a relationship between the Euclidean margin $d_e(F(\cdot))$ and the manifold margin $d_{\mathcal{M}}(F(\cdot))$.

Proposition 3.2.1 *Let $\mathcal{M} \subset \mathbb{R}^d$ be a r-dimensional ($r \leq d$) linear subspace that contains the ground-truth manifold $\mathcal{M}^*$, such that $\mathcal{M}^* \subset \mathcal{M}$, $F(\cdot)$ a linear classifier with a random decision boundary, then $\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}$.*

The detailed proof is shown in Section 3.7.1. The margin-based analysis explains the design choice of the running loss in Equation (3.2) that depends on an embedding manifold. Specifically, using an embedding manifold (a submersion function) to measure the running loss leads to an increased margin.

A demonstration of margin increase. Figure 3.3 (a) illustrates a binary classification dataset situated on a one-dimensional manifold, depicted as a green curve ($\mathcal{M}$). With a classifier denoted as $F(\cdot)$, the shortest distance by which an instance changes from one class to another is represented by the manifold margin $d_{\mathcal{M}}(F(\cdot))$, where the orange curve represents $2 \cdot d_{\mathcal{M}}(F(\cdot))$. As indicated in Figure 3.3 (b), this classifier yields a relatively small Euclidean margin. In Figure 3.3 (c), class-1 predictions are grouped into subsets A and B, while class-2 predictions fall into subsets C and D. The manifold projection, $\mathcal{E}(\cdot)$, maps subsets A and D onto the upper



(a): $2 \cdot d_{\mathcal{M}}(F(\cdot))$ (b): $d_e(F(\cdot))$ (c): $\mathcal{E}(\cdot)$ (d): $d_e(F \circ \mathcal{E}(\cdot))$

Figure 3.3: (a): a binary classification dataset embedded inside of a manifold $\mathcal{M}$. The manifold margin with a classifier F is shown. (b): the Euclidean margin with a classifier $F(\cdot)$. (c): the manifold projection and classifier form four partitions, regions A and B are projected onto the top portion of $\mathcal{M}$, and regions C and D are projected onto the lower portion of $\mathcal{M}$. (d): the manifold projection margin.

section, and subsets B and C onto the lower section of $\mathcal{M}$. This results in the classifier's decision boundary and the manifold projection dividing the $\mathbb{R}^2$ space into four distinct segments. For the combined classifier $F \circ \mathcal{E}(\cdot)$, any data points in regions A and D are classified as class-2, whereas those in regions B and C are classified as class-1. Consequently, Figure 3.3 (d) presents the decision boundary of $F \circ \mathcal{E}(\cdot)$, illustrating a substantial improvement in the manifold projection margin (highlighted in orange) compared to the Euclidean margin.

3.3 An Optimal Control Solver for Self-Healing

Section 3.3.1 discusses Pontryagin's Maximum Principle. Following this, Section 3.3.2 presents a more efficient approach aimed at reducing the computational cost incurred during control generation in the inference process.

3.3.1 Control Solver Based on Pontryagin's Maximum Principle

The proposed self-healing neural network can be achieved by solving the dynamical programming principle [77]. However, this has exponential complexity with respect to the state dimension. To overcome the computational challenge, we first describe a general solver for the optimal control problem in Equation (3.1) based on Pontryagin's Maximum Principle [78].

To begin with, we define the Hamiltonian $H(t, \mathbf{x}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t, \mathbf{u}_t)$ as

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

Pontryagin's maximum principle consists of a two-point boundary value problem,

$$\mathbf{x}_{t+1}^* = \nabla_p H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*), \qquad (\mathbf{x}_0^*, \mathbf{y}) \sim \mathcal{D}, \tag{3.8}$$

$$\mathbf{p}_t^* = \nabla_x H(t, \mathbf{x}_t^*, \mathbf{p}_{t+1}^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*), \qquad \mathbf{p}_T^* = \mathbf{0}, \tag{3.9}$$

plus a maximum condition of the Hamiltonian.

$$H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*) \geq H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t), \ \ \forall \mathbf{u} \in \mathbb{R}^{d'} \ \text{ and } \ \forall t \in \mathcal{T}. \tag{3.10}$$

To obtain a numerical solution, one can consider iterating through the forward dynamic Equation (3.8) to obtain all states $\{\mathbf{x}_t\}_{t=0}^{T-1}$, the backward dynamic Equation (3.9) to compute the adjoint states $\{\mathbf{p}_t\}_{t=0}^{T-1}$, and updating the Hamiltonian Equation (3.10) with current states and adjoint states via gradient ascent [79]. This iterative process is continued until convergence.

3.3.2 A Fast Implementation of the Closed-Loop Control

Now we discuss the computational overhead caused by the closed-loop control, and propose an accelerated numerical solver based on the unique condition of optimality in Pontryagin's Maximum Principle.

Computational Overhead in Inference. Upon implementing the closed-loop control module for inference, the traditional method of forward propagation is modified with iterations over Hamiltonian dynamics. For each input data, solving the optimal control problems necessitates iterating through both the forward dynamics, as per Equation (3.8), and the backward adjoint dynamics, according to Equation (3.9). Additionally, it involves the maximization of the Hamiltonian, as outlined in Equation (3.10), across all layers. This maximization process,

when repeated n times, leads to an approximate n-fold increase in time complexity compared to standard inference procedures. The computational overhead prevents deploying the closed-loop control module in real-world applications.

A Faster Pontryagin's Maximum Principle Solver. To tackle this challenge, we draw on the method of successive approximation [80], grounded in the optimal condition derived from Pontryagin's Maximum Principle. For any given input data, the state variables and adjoint states are determined by Equation (3.8) and Equation (3.9), respectively, corresponding to the current control set $\{\mathbf{u}_t\}_{t=0}^{T-1}$. To meet the optimal condition of the objective function as stated in Equation (3.1), it's essential to maximize all Hamiltonians as indicated in Equation (3.10). Rather than iterating all three Hamiltonian dynamics for a single control solution update, an alternative approach involves locally optimizing the t^{th} Hamiltonian for each $t \in [0, \cdots, T-1]$ using the current state $\mathbf{x}_t$ and the adjoint state $\mathbf{p}_{t+1}$. This approach allows for multiple updates of the control solution $\mathbf{u}_t$ within one full iteration cycle. After obtaining a locally optimal control $\mathbf{u}_t$ by maximizing $H(t, \mathbf{x}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t, \mathbf{u}_t)$ with respect to $\mathbf{u}_t$, the adjoint state $\mathbf{p}_{t+1}$ is backpropagated to $\mathbf{p}_t$ using the adjoint dynamic in Equation (3.9), followed by maximizing $H(t-1, \mathbf{x}_{t-1}, \mathbf{p}_t, \boldsymbol{\theta}_{t-1}, \mathbf{u}_{t-1})$. In this framework, simulating the Hamiltonian dynamics as per Equation (3.8), Equation (3.9), and Equation (3.10) n times is broken down into $maxItr$ full iterations and $InnerItr$ local updates. The number of $maxItr$ can be much lower than n, as the convergence process is expedited by the locally optimal control solutions derived from $InnerItr$ updates. Consequently, instead of repeating the complete Hamiltonian dynamics n times, this enhanced methodology involves iterating $maxItr$ full Hamiltonian dynamics along with $InnerItr$ local updates. The detailed implementation is presented in Algorithm 1.

3.4 Theoretical Analysis

In this section, we formally establish an error analysis for the closed-loop control framework. Let $\mathbf{x}_t$ be a "clean" state originated from an unperturbed data sample $\mathbf{x}_0$, and $\mathbf{x}_{\epsilon,t}$ be the

Algorithm 1 Method of Successive Approximation.

Input: Input $\mathbf{x}_0$ (possibly perturbed), a trained neural network $F(\cdot)$, embedding functions $\{\mathcal{E}_t(\cdot)\}_{t=0}^{T-1}$, control regularization c, learning rate lr, maxItr, InnerItr.
Output: $\mathbf{x}_T$.
Initialize controls $\{\mathbf{u}_t\}_{t=0}^{T-1}$ with the greedy solution.
for $i = 0$ to maxItr **do**
  The controlled initial condition:
  $\mathbf{x}_0^i = \mathbf{x}_0 + \mathbf{u}_0^i$.
  for $t = 0$ to $T-1$ **do**
    Controlled forward propagation Equation (3.8):
    $\mathbf{x}_{t+1}^i = F_t(\mathbf{x}_t^i, \mathbf{u}_t^i, \boldsymbol{\theta}_t)$
  end for
  The terminal condition of the adjoint state is set to 0:
  $\mathbf{p}_T^i = \mathbf{0}$
  for $t = T-1$ to 0 **do**
    for $\tau = 0$ to InnerItr **do**
      Compute Hamiltonian:
      $H(t, \mathbf{x}_t^i, \mathbf{p}_{t+1}^i, \boldsymbol{\theta}_t, \mathbf{u}_t^{i,\tau}) = \mathbf{p}_{t+1}^i \cdot F_t(\mathbf{x}_t^i, \mathbf{u}_t^{i,\tau}, \boldsymbol{\theta}_t) - \mathcal{L}(\mathbf{x}_t^i, \mathbf{u}_t^{i,\tau}, \mathcal{E}_t(\mathbf{x}_t^i))$
      Maximize Hamiltonian with respect to control $\mathbf{u}_t$:
      $\mathbf{u}_t^{i,\tau+1} = \mathbf{u}_t^{i,\tau} + \mathrm{lr} \cdot \nabla_{\mathbf{u}} H(t, \mathbf{x}_t^i, \mathbf{p}_{t+1}^i, \boldsymbol{\theta}_t, \mathbf{u}_t^{i,\tau})$
    end for
    Backward propagation Equation (3.9):
    $\mathbf{p}_t^i = \mathbf{p}_{t+1}^i \cdot \nabla_{\mathbf{x}} F(\mathbf{x}_t^i, \mathbf{u}_t^i, \boldsymbol{\theta}_t) - \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}_t^i, \mathbf{u}_t^i, \mathcal{E}_t(\mathbf{x}_t^i))$
  end for
end for

perturbed states originating from a possible attacked or corrupted data sample $\mathbf{x}_{\epsilon,0} = \mathbf{x}_0 + \mathbf{z}$. Within our self-healing neural network design, the controlled state is represented as $\overline{\mathbf{x}}_{\epsilon,t} = \mathbf{x}_{\epsilon,t} + \mathbf{u}_t$. The focus of the theoretical error analysis is to assess $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|$, which quantifies the discrepancy between $\mathbf{x}_t$ and $\overline{\mathbf{x}}_{\epsilon,t}$. In Section 3.4.1, we present error estimates within a framework consisting of a linear dynamical system and a linear embedding function. Meanwhile, Section 3.4.2 focuses on establishing an upper limit for errors that arise from the linear approximation of a nonlinear dynamical system and embedding manifold. The error estimation in the most general case is constructed as follows,

$$\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\| \leq \textbf{Linear control system} + \textbf{Linearization error}.$$

3.4.1 Error Estimation For The Linear Case

Now we analyze the error of the self-healing neural network for a simplified case with linear activation functions. We denote $\boldsymbol{\theta}_t$ as the Jacobian matrix of the nonlinear transformation $F_t(\cdot)$ centered at $\mathbf{x}_t$, such that $\boldsymbol{\theta}_t = F_t'(\mathbf{x}_t)$. In the linear case, the solution of the running loss in Equation (3.2) is a projection onto the linear subspace, which admits a closed-form solution. For a perturbed input, $\mathbf{q}_0 = \mathbf{x}_0 + \mathbf{z}$ with some perturbation $\mathbf{z}$, we denote $\{\mathbf{q}_{\epsilon,t}\}_{t=0}^{T-1}$ as sequence of states of the linear system, and $\{\overline{\mathbf{q}}_{\epsilon,t}\}_{t=0}^{T-1}$ as the states adjusted by the linear control. The perturbation $\mathbf{z} \in \mathbb{R}^d$ admits a direct sum of two orthogonal components, $\mathbf{z} = \mathbf{z}^{\parallel} \oplus \mathbf{z}^{\perp}$. Here $\mathbf{z}^{\parallel} \in \mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$ is a perturbation within the tangent space, and $\mathbf{z}^{\perp}$ lies in the orthogonal complement of $\mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$.

The following theorem provides an upper bound of $\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2$.

Theorem 3.4.1 *For $t \geq 1$, we have an error estimation for the linear system*

$$\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 \leq \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \left(\alpha^{2t} \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\parallel}\|_2^2 + \gamma_t \|\mathbf{z}\|_2^2 \big(\gamma_t \alpha^2 (1 - \alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big) \right).$$

where $\gamma_t := \max\limits_{s \leq t} \big(1 + \kappa(\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)^2\big) \|\mathbf{I} - (\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)^T (\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)\|_2$, $\kappa(\boldsymbol{\theta})$ is condition number of $\boldsymbol{\theta}$, $\alpha = \frac{c}{1+c}$, and c represents the control regularization. In particular, the equality

$$\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 = \alpha^{2t} \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\parallel}\|_2^2$$

holds when all $\boldsymbol{\theta}_t$ are orthogonal.

The detailed derivation is presented in Section 3.7.2. The error upper bound is tight since it becomes the actual error if all the linear transformations are orthogonal matrices. Note that the above bound from the greedy control solution is a strict upper bound of the optimal control solution. The greedy solution does not consider the dynamic, and it optimizes each running loss individually.

3.4.2 Error Analysis of Nonlinear Networks with Closed-loop Control

Here we provide an error analysis for the self-healing neural network with general nonlinear activation functions. For a 3-dimensional tensor, e.g. the Hessian $F''(\mathbf{x})$, we define the 2-norm of $F''(\mathbf{x})$ as

$$\|F''(\mathbf{x})\|_* := \sup_{\mathbf{z}\neq\mathbf{0}} \frac{\|F''(\mathbf{x})^{i,j,k}\mathbf{z}_j\mathbf{z}_k\|_2}{\|\mathbf{z}\|_2^2}.$$

For the nonlinear transformation $F_t(\cdot) \in \mathcal{C}^2$ at layer t, we assume its Hessian $F_t''(\cdot)$ is uniformly bounded, i.e., $\sup_{\mathbf{x}\in\mathbb{R}^d}\|F_t''(\mathbf{x})\|_* \leq \beta_t$. Let $f_t \in \mathcal{C}^2 : \mathbb{R}^d \to \mathbb{R}^{d-r}$ be the submersion of the embedding manifold $\mathcal{M}_t$, we assume its Hessian is uniformly bounded, i.e., $\sup_{\mathbf{x}\in\mathbb{R}^d}\|f_t''(\mathbf{x})\|_* \leq \sigma_t$. We use $\mathbf{x}_t$, $\mathbf{x}_{\epsilon,t}$ and $\overline{\mathbf{x}}_{\epsilon,t}$ to denote the clean states, perturbed states without control and the states adjusted with closed-loop control, respectively. The initial perturbation $\mathbf{z} = \epsilon \cdot \mathbf{v}$, where $\|\mathbf{v}\|_2 = 1$ and $\mathbf{v} = \mathbf{v}^{\|} \oplus \mathbf{v}^{\perp}$. Let

- $k_t = 4\sigma_t\|(f_t'(\mathbf{x}_t)f_t'(\mathbf{x}_t)^T)^{-1}\|_2 \cdot (\|f_t'(\mathbf{x}_t)\|_2 + 2\sigma_t)$,
- $\delta_{\mathbf{x}_t} = \|\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0\|_2^2 \cdot \Big(\alpha^{2t}\|\mathbf{v}^{\perp}\|_2^2 + \|\mathbf{v}^{\|}\|_2^2 + \gamma_t\|\mathbf{v}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha-\alpha^t)\big)\Big)$.

The following theorem provides an error estimation between $\overline{\mathbf{x}}_{\epsilon,t}$ and $\mathbf{x}_t$.

Theorem 3.4.2 *If the initial perturbation satisfies*

$$\epsilon^2 \leq \frac{1}{\Big(\sum_{i=0}^{T-1}\delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i)\prod_{j=i+1}^{T-1}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\Big)}.$$

for $1 \leq t \leq T$, we have the following error bound for the closed-loop controlled system

$$\begin{aligned}
&\|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1}\|_2 \\
&\leq \|\boldsymbol{\theta}_t\cdots\boldsymbol{\theta}_0\|_2\Big(\alpha^{t+1}\|\mathbf{z}^{\perp}\|_2 + \|\mathbf{z}^{\|}\|_2 + \|\mathbf{z}\|_2\big(\gamma_{t+1}\alpha(1-\alpha^t) + \sqrt{2\gamma_{t+1}(\alpha-\alpha^{t+1})}\big)\Big) \\
&\quad + \Big(\sum_{i=0}^{t}\delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i)\prod_{j=i+1}^{t}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\Big)\epsilon^2.
\end{aligned}$$

The detailed proof is provided in Section 3.7.3, 3.7.4, and 3.7.5. From Theorem 3.4.2, we have the following intuitions:

- The error estimation has two main components: a linearization error in the order of $\mathcal{O}(\epsilon^2)$, and the error of $\mathcal{O}(\epsilon)$ of the linearized system. Specifically, the linearization error becomes smaller when the activation functions and embedding manifolds behave more linearly (k_t and β_t become smaller).

- The closed-loop control minimizes the perturbation components $\mathbf{z}^{\perp}$ within the orthogonal complements of the tangent spaces. This is consistent with the manifold hypothesis, the robustness improvement is more significant if the underlying data are embedded in a lower dimensional manifold ($\|\mathbf{z}^{\|}\|_2 \to 0$).

- The above error estimation improves as the control regularization c goes to 0 (so $\alpha \to 0$). It is not the sharpest possible as it relies on a greedily optimal control at each layer. The globally optimal control defined by the Ricatti equation may achieve a lower loss when $c \neq 0$.

- The error estimation is done via linearizing both the underlying dynamical system and embedding manifolds. This may result in a loose error bound when the underlying trajectory is diverging due to the non-negligible linearization error. The goal of this error estimation is to explain the working principle behind the proposed method in the general nonlinear case, which does not conflict with the linearization error.

Remark 3.4.1 *The derivation of the error estimation depends on the assumption that the ground-truth manifold is given. To account for the approximation from the estimated embedding manifold that has non-zero reconstruction loss, the error from the imperfect embedding manifold should propagate in the same way as the linearization error at every layer. Specifically, the embedding error at* t^{th} *layer contributes to both the linearization of the dynamical system and the tangent space approximation of the nonlinear embedding manifold at* $(\mathrm{t}+1)^{\mathrm{th}}$ *layer, then this error is accumulated towards the terminal state.*

3.5 Numerical Experiments

In this section, we test the performance of the proposed self-healing framework. Specifically, we show that using only one set of embedding functions can improve the robustness of many pre-trained models consistently. Section 3.5.1 shows that the proposed method can significantly improve the robustness of both standard and robustly trained models on CIFAR-10 against various perturbations. Furthermore, in the same experimental setting, Sections 3.5.2 and 3.5.3 evaluate our method on CIFAR-100 and Tiny-ImageNet datasets, which empirically verifies the effectiveness and generalizability of the self-healing machinery. Section 3.5.4 provides a summary of these numerical results. Section 3.5.5 evaluates the proposed control method on a multi-label classification task. Finally, Section 3.5.6 conducts an ablation study that shows both intuitive and exploratory justifications for the control method.

3.5.1 Experiments On CIFAR-10 Dataset

We evaluate all controlled models under an "oblivious attack" setting [2]. In this setting, the pre-trained models are fully accessible to an attacker, but the control information is not released. Meanwhile, the controllers do not know the incoming attack algorithms. We will show that by using one set of embedding functions, our self-healing method can improve the robustness of many pre-trained models against a broad class of perturbations. Our experimental setup is summarized below.

- **Baseline models.** We showcase that *one set of controllers* can consistently increase the robustness of many pre-trained ResNets when those models are trained via standard training (momentum SGD) and adversarial training (TRADES [82]). Specifically, we use Pre-activated ResNet-18 (**RN-18**), -34 (**RN-34**), -50 (**RN-50**), wide ResNet-28-8 (**WRN-28-8**), -34-8 (**WRN-34-8**) as the testing benchmarks.

[2]This consideration is general, e.g. [81] has adopted this setting in the previous NIPS competition on defense against adversarial attacks.

Table 3.1: CIFAR-10 accuracy measure: baseline model / controlled model

$\ell_\infty : \epsilon = 8/255$, $\ell_2 : \epsilon = 0.5$, $\ell_1 : \epsilon = 12$				
Standard models				
	None	AA (ℓ_∞)	AA (ℓ_2)	AA (ℓ_1)
RN-18	**94.71** / 92.81	0. / **63.89**	0. / **82.1**	0. / **75.75**
RN-34	**94.91** / 92.84	0. / **64.92**	0. / **83.64**	0. / **78.05**
RN-50	**95.08** / 92.81	0. / **64.31**	0. / **83.33**	0. / **77.15**
WRN-28-8	**95.41** / 92.63	0. / **75.39**	0. / **86.71**	0. / **84.5**
WRN-34-8	**94.05** / 92.77	0. / **64.14**	0. / **82.32**	0. / **73.54**
Robust models (trained with ℓ_∞ perturbations)				
	None	AA (ℓ_∞)	AA (ℓ_2)	AA (ℓ_1)
RN-18	82.39 / **87.51**	48.72 / **66.61**	58.8 / **79.88**	9.86 / **42.85**
RN-34	84.45 / **87.93**	49.31 / **65.49**	57.27 / **78.81**	7.21 / **40.74**
RN-50	83.99 / **87.57**	48.68 / **65.17**	57.25 / **78.26**	6.83 / **39.44**
WRN-28-8	85.09 / **87.66**	48.13 / **64.44**	54.38 / **77.08**	5.38 / **41.78**
WRN-34-8	84.95 / **87.14**	48.47 / **64.55**	54.36 / **77.15**	4.67 / **42.65**

- **Robustness evaluations.** We evaluate the performance of all models with clean testing data (**None**), and auto-attack (**AA**) [2] that is measured by ℓ_∞, ℓ_2 and ℓ_1 norms. Auto-attack that is an ensemble of two gradient-based auto-PGD attacks [2], fast adaptive boundary attack [83] and a black-box square attack [84].

- **Embedding functions.** We choose the fully convolutional networks (FCN) [85] as an input embedding function and a 2-layer auto-encoder as an embedding function for the hidden states. Specifically, we use one set of embedding functions for all 5 pre-trained models. The training objective function of the t^{th} embedding function follows Equation (3.4), where both model and data information are used.

- **Pontryagin's Maximum Principle hyper-parameters setting.** We choose 3 outer iterations and 10 inner iterations with 0.001 as control regularization parameters in Pontryagin's Maximum Principle solver. As in Algorithm 1, maxIte=3, InnerItr=10, and $c = 0.001$.

As shown in Table 3.1, for standard trained baseline models, despite the high accuracy

of clean data, their robustness against strong auto-attack degrades to 0% accuracy under all measurements. The self-healing process is attack-agnostic, and it improves the robustness against all perturbations with negligible degradation on clean data. Specifically, the controlled models have more than 80% and near 80% accuracies against perturbations measured by ℓ_2 and ℓ_1 norms respectively.

On adversarially trained baseline models. Since all robust baseline models are pre-trained with ℓ_∞ measured adversarial examples, they show strong robustness against ℓ_∞ auto-attack. Surprisingly, models that trained using ℓ_∞ as adversarial training objective preserve strong robustness against ℓ_2 perturbations. However, a ℓ_1 measured perturbation can significantly degrade their robustness. On average, our proposed control method has achieved 20% accuracy improvements against ℓ_∞ and ℓ_2 perturbations, and a near 40% improvement against ℓ_1 perturbation. Surprisingly, by applying the proposed control module, all adversarially trained models have achieved higher accuracy on clean testing data.

3.5.2 Experiments On CIFAR-100 Dataset

In this section, we investigate the effectiveness of self-healing on the more challenging CIFAR-100 dataset. We summarize our experiment settings below.

- **Baseline models.** We consider different variants of Wide-ResNet. Specifically, we use Wide-ResNet-28-10 (**WRN-28-10**), -34-10 (**WRN-34-10**), -76-10 (**WRN-76-10**). We show that one set of controllers can consistently increase the robustness of all 3 pre-trained models when those models are trained via momentum SGD and adversarial training (TRADES [82]).

- **Other settings.** The embedding functions and Pontryagin's Maximum Principle settings follow the same.

In Table 3.2, the proposed self-healing framework consistently improves the robustness of adversarially trained models on the CIFAR-100 dataset. On average, the self-healing models

Table 3.2: CIFAR-100 accuracy measure: baseline model/self-healing

Standard models				
$\ell_\infty : \epsilon = 8/255$, $\ell_2 : \epsilon = 0.5$, $\ell_1 : \epsilon = 12$				
	None	AA (ℓ_∞)	AA (ℓ_2)	AA (ℓ_1)
WRN-28-10	**79.53** / 75.80	0.04 / **11.43**	0.06 / **32.70**	0.03 / **28.53**
WRN-34-10	**79.12** / 72.70	0.02 / **13.89**	0.03 / **29.78**	0.02 / **31.78**
WRN-76-10	**79.28** / 71.10	0.01 / **19.31**	0.03 / **28.96**	0.01 / **35.13**
Robust models (trained with ℓ_∞ perturbations)				
	None	AA (ℓ_∞)	AA (ℓ_2)	AA (ℓ_1)
WRN-28-10	**56.96** / 56.84	24.97 / **30.81**	29.54 / **39.18**	3.24 / **16.43**
WRN-34-10	**57.32** / 56.91	25.35 / **31.04**	29.68 / **39.64**	2.99 / **17.66**
WRN-76-10	**57.58** / 57.11	24.84 / **29.96**	27.81 / **38.05**	2.41 / **19.13**

have achieved $10\% \sim 20\%$ accuracy improvement with almost no effects on the clean data performance. Although the improvements are not as significant as in the CIFAR-10 experiment, this is due to the hardness of constructing embedding manifolds for this more challenging dataset. Specifically, it is more difficult to distinguish the controlled data point among 100 different classes than 10 classes on a single embedding manifold.

3.5.3 Experiments On Tiny-ImageNet

Finally, we examine the proposed self-healing framework on the Tiny-ImageNet dataset. Tiny-ImageNet contains $100,000$ and $10,000$ of 64×64 sized training and validation images with 200 different classes. Although over-fitting is more significant in this dataset, we show that the proposed self-healing framework can consistently improve the robustness of pre-trained models. The experimental settings are summarized below.

- **Baseline models.** We consider **EfficientNet-b0**, **EfficientNet-b1** and **EfficientNet-b2** trained via momentum SGD and adversarial training (TRADES [82]) as testing benchmarks.

- **Embedding functions.** We choose SegNet [86] as an input embedding function, and a 2-layer auto-encoder as an embedding function for the hidden states. The training objective

Table 3.3: Tiny-ImageNet accuracy measure: baseline model / controlled

$\ell_\infty : \epsilon = 4/255$, $\ell_2 : \epsilon = 0.8$, $\ell_1 : \epsilon = 10$				
Standard models				
	None	AA (ℓ_∞)	AA (ℓ_2)	AA (ℓ_1)
EfficientNet-b0	57.68 / **59.92**	0.21 / **46.08**	1.73 / **49.86**	5.86 / **50.4**
EfficientNet-b1	57.99 / **59.72**	0.13 / **44.35**	1.24 / **48.26**	4.43 / **48.86**
EfficientNet-b2	58.06 / **59.3**	0.25 / **44.33**	1.40 / **47.86**	4.58 / **48.39**
Robust models (trained with ℓ_∞ perturbations)				
EfficientNet-b0	**45.16** / 41.09	22.56 / **30.69**	26.86 / **34.57**	24.42 / **34.51**
EfficientNet-b1	**46.29** / 41.18	22.70 / **30.91**	26.60 / **34.10**	22.30 / **33.67**
EfficientNet-b2	**45.64** / 41.58	23.26 / **31.42**	26.77 / **34.45**	21.59 / **34.00**

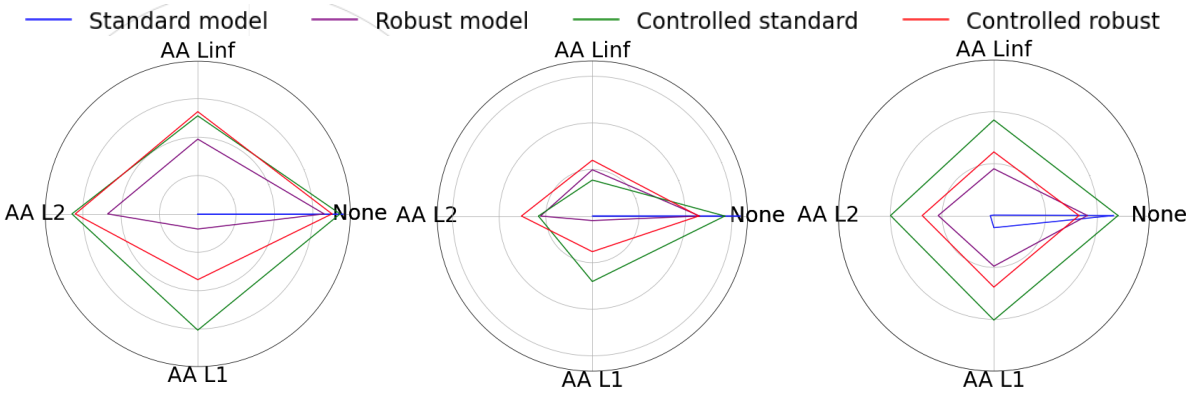
function of the t^{th} embedding function follows Equation (3.4), where both model and data information are used.

- **Pontryagin's Maximum Principle hyper-parameters setting.** The Pontryagin's Maximum Principle setting follows the same.

In this task, we aim to validate the practical applicability of the proposed method on a generally large dataset and deep network architectures. As shown in Table 3.3, on the challenging Tiny-ImageNet dataset, despite the high accuracy of clean data, as expected, all pre-trained models result in an extremely poor performance against auto-attacks. The proposed framework can improve all three pre-trained EfficientNets consistently against auto-attacks. Specifically, the controlled models have shown $45\% \sim 50\%$ robustness improvements against all perturbations.

3.5.4 Summary On Numerical Experiments

Figure 3.4 shows the radar plots of accuracy against many perturbations on some chosen baseline models. Overall, the self-healing via closed-loop control consistently improves the baseline model performance. Notice that adversarial training can effectively improve the robustness of baseline models against a certain type of perturbation (e.g. Auto-attack measured in ℓ_∞).



(a): CIFAR-10 (b): CIFAR-100 (c): Tiny-Imagenet

Figure 3.4: (a), (b) and (c) are radar plots that summarize RN-18 in Table 3.1, WRN-76-10 in Table 3.2, and EfficientNet-b0 in Table 3.3 respectively.

However, those seemingly robust models are extremely vulnerable against other types of perturbations (e.g. Auto-attack measured in ℓ_1). The proposed method is attack-agnostic and can consistently improve the robustness of many baseline models against various perturbations.

3.5.5 Experiment On Multi-Label Classification

The robustness issue of multi-label classification is little explored. We consider the PASCAL Visual Object Detection (VOC) dataset and adopt the standard training protocol where we consider a union of the VOC 2007 and 2012 training dataset following [87]. For testing, we use the VOC 2007 test with 4952 test images and 20 classes [88]. We resize the original images to $128 \times 128 \times 3$ for computational efficiency. We use average precision as a measurement for all models.

We apply the proposed method on **EfficientNet-b0**, **b1** and **b2** that are trained via momentum SGD. For control settings, we choose fully convolutional networks (FCN) [85] as an input embedding function and a 2-layer auto-encoder as an embedding function for the hidden states. The Pontryagin's Maximum Principle hyper-parameter settings are the same as in previous experiments. We evaluate the performance of all models with clean data (**None**), project gradient descent (**PGD**) measured by ℓ_∞ and ℓ_2 norms, and an out-of-distribution test (**OOD**) where the testing images are transformed by Gaussian blurring. In Table 3.4, despite the high

Table 3.4: VOC average precision: baseline model / controlled

Standard models, $\ell_\infty : \epsilon = 8/255$, $\ell_2 : \epsilon = 0.5$				
	None	PGD (ℓ_∞)	PGD (ℓ_2)	OOD
EfficientNet-b0	**0.794** / 0.772	0.181 / **0.245**	0.452 / **0.548**	0.566 / **0.597**
EfficientNet-b1	**0.810** / 0.785	0.170 / **0.199**	0.478 / **0.558**	0.580 / **0.634**
EfficientNet-b2	**0.796** / 0.786	0.202 / **0.233**	0.471 / **0.561**	0.602 / **0.630**

performance of clean data, those models are extremely vulnerable to adversarial attacks and out-of-distribution shifts. By equipping the proposed control method, on average, the adversarial robustness of all models has been improved by $\sim 5\%$, and $\sim 3\%$ on out-of-distribution shift.

3.5.6 Ablation Study

In this section, we show both intuitive and exploratory justifications for the control method. Then we empirically validate the margin-based analysis in Section 3.2.2. Finally, we analyze how the numerical approximation errors affect the controlled model performance.

Intuitive justification of the control method. We provide an intuitive justification of how a manifold-based recovery is beneficial for robustness. We use the VOC dataset as an example and build the control algorithm with a fully convolutional network as used in Section 3.5.5. Figure 3.5 (left) shows an image that belongs to the classes of person and dog, and this clean image is located in both the estimated embedding manifold (red line) and the ground-truth manifold (blue line). In Figure 3.5 (middle), an adversarial attack drives the clean image out of the embedding manifold (middle plot). Notice how the adversarial perturbation changes the texture of the image background as highlighted in the red circle. In Figure 3.5 (right), the closed-loop control adjusts the perturbed image back to the estimated embedding manifold and removes the texture perturbation partially. Notice that the controlled image is still different from the original clean image, as shown by the red and green dots. However, the controlled

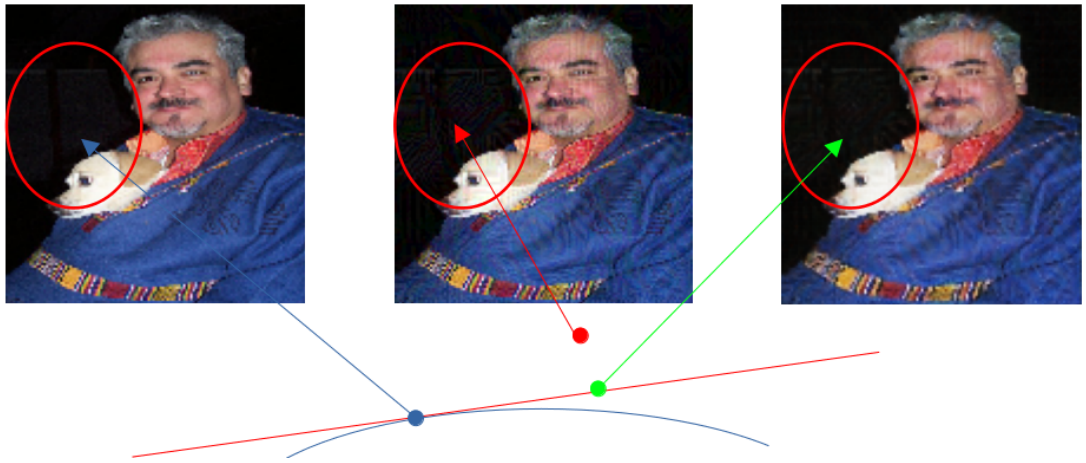


Figure 3.5: Demonstration of manifold-based control method on VOC input data. Left: a clean image that belongs to classes person and dog. Middle: a perturbed image that is predicted as cow. Right: controlled image. The main difference between those three images is highlighted in the red circle. The red and blue lines represent the estimated embedding manifold and the ground-truth manifold respectively, and blue, red and green dots show the locations of clean, perturbed and controlled images respectively.

image from the manifold-based method approaches the clean image compared with its perturbed counterpart.

Exploratory justification of the manifold-based control method. Since the proposed closed-loop control method depends on the "manifold hypothesis" that real-world data generally lies in a low-dimensional manifold [75], we validate a hypothesis that inaccurate embedding manifold and poor controlled model performance are correlated.

To see this, we use a r-dimensional linear embedding subspace (generated from principle component analysis) with a basis $\mathbf{V} \in \mathbb{R}^{d\times r}$ to estimate the ground-truth manifold. The accuracy of an embedding subspace can be measured by reconstruction loss $\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{V}\mathbf{V}^t\mathbf{x}_i - \mathbf{x}_i\|$. In the linear setting, the accuracy of a linear embedding subspace can be tuned by adjusting its dimension r. In Figure 3.6, we use linear orthogonal projection as embedding function to implement the closed-loop control in Algorithm 1. In each plot, we fix the embedding of the hidden state and tune the dimension of the embedding subspace of input data to show the behavior of reconstruction loss (red) and accuracy of the controlled model (green).

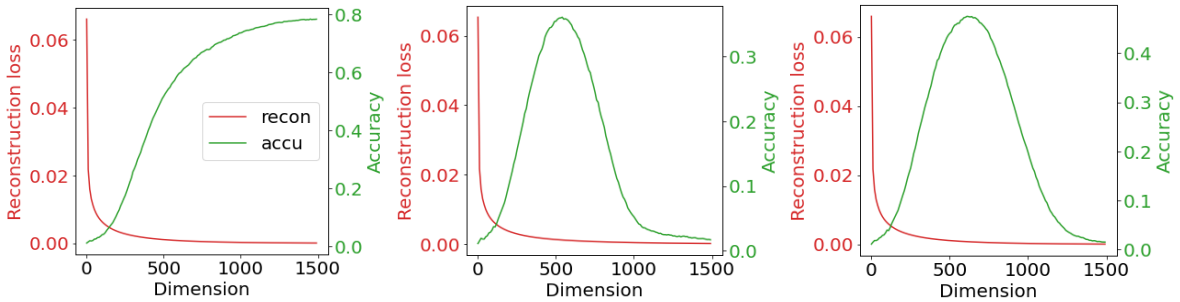
Generally, the set of estimated manifolds from a chosen manifold learning setting may not contain the ground-truth manifold. For instance, with a fixed dimension r, there might not exist a linear embedding subspace that results in 0 reconstruction loss on the given dataset. In this case, the ground-truth manifold cannot be correctly estimated by the chosen manifold learning method. Figure 3.6 (a) shows reconstruction loss versus controlled model performance with respect to varying dimensions on the CIFAR-100 clean test set. As can be seen, when the chosen linear embedding subspace has a low dimension and cannot contain the ground-truth manifold, the prediction accuracy is low due to inaccurate reconstruction.

As the dimension of the linear embedding subspace increases, the reconstruction loss of the embedding subspace decreases. A linear embedding subspace with 0 reconstruction loss contains the ground-truth manifold. However, an accurate embedding subspace that contains the ground-truth manifold may lead to low robustness improvement. To see this, we further increase the dimension of the linear embedding subspace. Figure 3.6 (b) and (c) show the reconstruction loss versus controlled model performance on perturbed data. As the dimension further increases, the robustness improvement reduces significantly. This happens because the perturbation lies within the embedding subspace and the perturbed data cannot be distinguished from the clean counterpart.

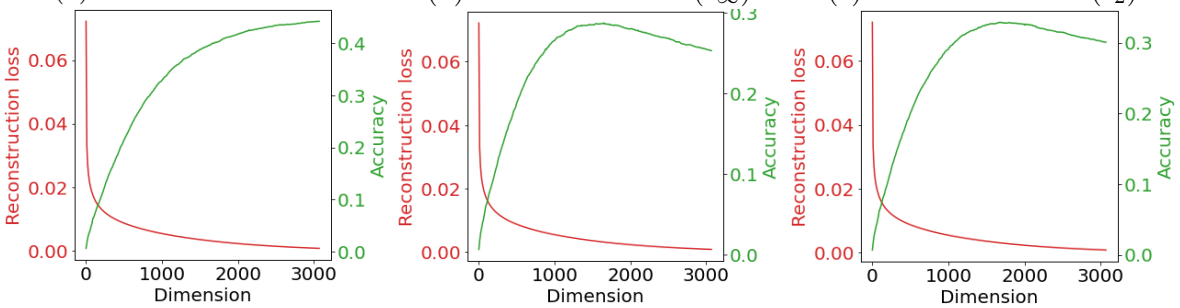
Similar behaviour can be seen in the Tiny-Imagenet dataset as shown in Figure 3.6 (d), (e), and (f). This supports the correlation between inaccurate embedding manifold and poor model performance

Empirical validation for the margin-based analysis. The margin-based analysis in Section 3.2.2 has shown that the composition of a classifier and a manifold-based embedding function can increase the Euclidean margin to a manifold margin. Although the analysis is conducted in a simplified case that considers a linear classifier with a random decision boundary, the implication of this analysis can be empirically demonstrated in more general settings.

Recall Proposition 3.2.1, if the estimated linear embedding subspace $\mathcal{M}$ contains the ground-truth manifold $\mathcal{M}^*$, for a linear classifier with a random decision boundary $F(\cdot)$, we have



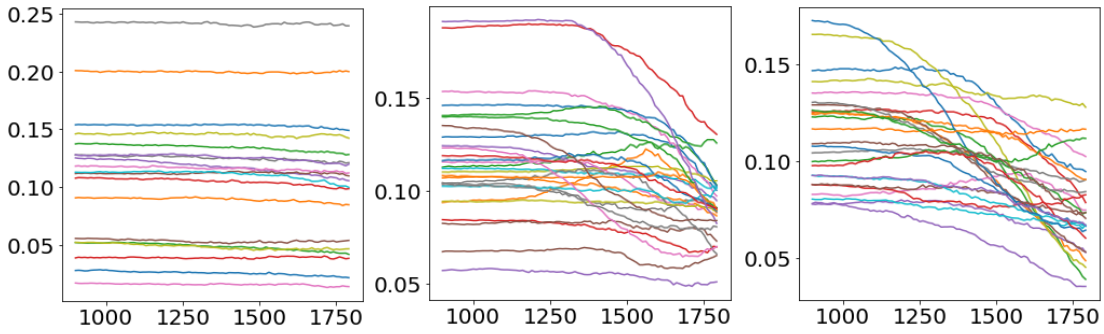
(a): CIFAR-100 Clean (b): CIFAR-100 AA (ℓ_∞) (c): CIFAR-100 AA (ℓ_2)



(d): T-Imagenet Clean (e): T-Imagenet AA (ℓ_∞) (f): T-Imagenet AA (ℓ_2)

Figure 3.6: (a) plots reconstruction loss (red y-axis) versus controlled model accuracy (green y-axis) on CIFAR-100 clean test set. The results on ℓ_∞ and ℓ_2 auto-attack perturbed data are shown in (b) and (c) respectively. (d), (e) and (f) conduct the same experiment on Tiny-Imagenet dataset.

$\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}$. To verify this analysis in more general settings, we choose a linear embedding subspace to embed the input data, and study the model performance and robustness with respect to varying dimensions of the linear embedding subspace. We randomly sample 20 linear classifiers to replace a pre-trained ResNet-18 on the CIFAR-10 dataset. The modified model is $F_{\text{lin}} \circ F_{\text{feature}} \circ \mathbf{V}\mathbf{V}^T$, where F_{lin} is a randomly sampled linear classifier, F_{feature} is the pre-trained feature extractor, $\mathbf{V} \in \mathbb{R}^{d \times r}$ is a basis of a r-dimensional linear embedding subspace, $\mathbf{V}\mathbf{V}^T$ is the orthogonal projection operator. As shown in Figure 3.7 (b) and (c), as the dimension r of the embedding subspace increases, the model robustness against both ℓ_∞ and ℓ_2 perturbations decreases. This validates Proposition 3.2.1 since $\sqrt{\frac{r}{d}}$ approaches to 1 as r increases, and the manifold margin is close to the Euclidean margin, which means the gained robustness decreases. Furthermore, the margin variation does not significantly affect the model performance on clean data, as shown in Figure 3.7 (a).



(a): CIFAR-10 Clean (b): CIFAR-10 AA (ℓ_∞) (c): CIFAR-10 AA (ℓ_2)

Figure 3.7: (a), (b) and (c) study the performance of a linear classifier with a random decision boundary on clean, ℓ_∞ and ℓ_2 auto-attack perturbed data respectively.

3.6 Conclusion

This chapter has improved the robustness of neural networks from a new self-healing perspective. By formulating the problem as a closed-loop control problem, we show that it is possible for a neural network to automatically detect and fix the potential errors caused by various perturbations and attacks. We have provided a margin-based analysis to explain why the designed control loss function can improve robustness. Moreover, we have also presented efficient numerical solvers to mitigate the computational overhead in inference. Our theoretical analysis has also provided a strict error bound of the neural network trajectory error under data perturbations. Numerical experiments have shown that this method can significantly increase the robustness of neural networks under various types of perturbations or attacks that were unforeseen in the training process.

3.7 Detailed Theoretical Proofs

3.7.1 Manifold Projection On Classifier Margin

Proposition 3.2.1 *Let $\mathcal{M} \subset \mathbb{R}^d$ be a r-dimensional ($r \leq d$) linear subspace that contains the ground-truth manifold $\mathcal{M}^*$, such that $\mathcal{M}^* \subset \mathcal{M}$, $F(\cdot)$ a linear classifier with a random decision boundary, then $\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}$.*

Proof: We define the ground-truth manifold as follows,

$$\mathcal{M}^* = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{0}, |\mathbf{c}^T\mathbf{x}| \geq d_{\text{margin}}\},$$

where $\mathbf{A}$ defines a subspace where the ground-truth manifold belongs, $\mathbf{c} \in \mathbb{R}^d$ is a unit vector and $|\mathbf{c}^T\mathbf{x}| \geq d_{\text{margin}}$ defines two half-spaces. That is, the ground-truth manifold $\mathcal{M}^*$ consists of two half-spaces corresponding to the two classes. Let $\mathcal{M}$ be an linear subspace $\mathcal{M} = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{0}\}$, in which case, $\mathcal{M}^* \subset \mathcal{M}$. We consider a linear classifier with a random decision boundary. Let $\mathbf{B}$ be a hyperplane that represents the decision boundary of this linear classifier, and $\hat{\mathbf{n}}$ a d-dimensional normal vector such that $\hat{\mathbf{n}}^T\mathbf{B} = \mathbf{0}$. A random linear classifier can be represented by $\hat{\mathbf{n}} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I})$.

Manifold and Euclidean margins attain the same $\mathbf{x}^*$. In this linear case, the following shows that the manifold margin $d_{\mathcal{M}}(F(\cdot))$ in Equation (3.6) is equivalent to $d_e(F \circ \mathcal{E}(\cdot))$ in Equation (3.7) where $\mathcal{E}(\cdot)$ is the orthogonal projection onto the subspace $\mathcal{M}$. The embedding manifold $\mathcal{M}$ is a linear subspace, the geodesics defined on the manifold are equivalent to the Euclidean norm,

$$\begin{aligned}\mathcal{R}_{\mathcal{M}}(\mathbf{a}, \mathbf{b}) &:= \inf_{\gamma \in \Gamma_{\mathcal{M}}(\mathbf{a},\mathbf{b}),} \int_0^1 \sqrt{\langle \gamma'(t), \gamma'(t) \rangle_{\gamma(t)}} dt, \\ &= \|\mathbf{a} - \mathbf{b}\|_2,\end{aligned}$$

the manifold margin can be shown as follows,

$$\begin{aligned}d_{\mathcal{M}}(F(\cdot)) &= \frac{1}{2} \inf_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}^*} \mathcal{R}_{\mathcal{M}}(\mathbf{x}_1, \mathbf{x}_2), \quad \text{s.t. } F(\mathbf{x}_1) \neq F(\mathbf{x}_2), \\ &= \frac{1}{2} \inf_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}^*} \|\mathbf{x}_1 - \mathbf{x}_2\|_2, \quad \text{s.t. } F(\mathbf{x}_1) \neq F(\mathbf{x}_2).\end{aligned}$$

Furthermore,

$$\begin{aligned}
d_e(F \circ \mathcal{E}(\cdot)) &= \inf_{\mathbf{x} \in \mathcal{M}^*} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \|\boldsymbol{\delta}\|_2, \quad \text{s.t. } F \circ \mathcal{E}(\mathbf{x}) \neq F \circ \mathcal{E}(\mathbf{x} + \boldsymbol{\delta}), \\
&= \inf_{\mathbf{x} \in \mathcal{M}^*} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \|\boldsymbol{\delta}\|_2, \quad \text{s.t. } F(\mathbf{x}) \neq F(\mathbf{x} + \mathcal{E}(\boldsymbol{\delta})), \\
&= \inf_{\mathbf{x} \in \mathcal{M}^*} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \inf_{\boldsymbol{\delta}' = \mathcal{E}(\boldsymbol{\delta})} \|\boldsymbol{\delta}'\|_2, \quad \text{s.t. } F(\mathbf{x}) \neq F(\mathbf{x} + \boldsymbol{\delta}'), \\
&= \inf_{\mathbf{x} \in \mathcal{M}^*} \inf_{\boldsymbol{\delta}' \in \mathcal{M}} \|\boldsymbol{\delta}'\|_2, \quad \text{s.t. } F(\mathbf{x}) \neq F(\mathbf{x} + \boldsymbol{\delta}'), \\
&= \frac{1}{2} \inf_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}^*} \|\mathbf{x}_1 - \mathbf{x}_2\|_2, \quad \text{s.t. } F(\mathbf{x}_1) \neq F(\mathbf{x}_2), \\
&= d_{\mathcal{M}}(F(\cdot)).
\end{aligned}$$

where the embedding function $\mathcal{E}(\cdot)$ is replaced by restricting $\boldsymbol{\delta} \in \mathcal{M}$.

The Euclidean margin in Equation (3.7) can be shown as follows,

$$d_e(F(\cdot)) = \inf_{\mathbf{x} \in \mathcal{M}^*} \|\mathbf{x}^T \hat{\mathbf{n}}\|_2.$$

Since $\mathcal{E}(\cdot)$ is a linear orthogonal projection, recall that $d_{\mathcal{M}}(F(\cdot)) = d_e(F \circ \mathcal{E}(\cdot))$,

$$d_{\mathcal{M}}(F(\cdot)) = d_e(F \circ \mathcal{E}(\cdot)) = \inf_{\mathbf{x} \in \mathcal{M}^*} \frac{\|\mathbf{x}^T \mathcal{E}(\hat{\mathbf{n}})\|_2}{\|\mathcal{E}(\hat{\mathbf{n}})\|_2} = \inf_{\mathbf{x} \in \mathcal{M}^*} \frac{\|(\mathcal{E}(\mathbf{x}))^T \hat{\mathbf{n}}\|_2}{\|\mathcal{E}(\hat{\mathbf{n}})\|_2} = \inf_{\mathbf{x} \in \mathcal{M}^*} \frac{\|\mathbf{x}^T \hat{\mathbf{n}}\|_2}{\|\mathcal{E}(\hat{\mathbf{n}})\|_2},$$

since $\mathbf{x} \in \mathcal{M}^* \in \mathcal{M}$, the orthogonal projection $\mathcal{E}(\mathbf{x}) = \mathbf{x}$. Therefore, the manifold margin is the Euclidean margin divided by a constant scalar $\|\mathcal{E}(\hat{\mathbf{n}})\|$, $d_{\mathcal{M}}(F(\cdot))$ and $d_e(F(\cdot))$ are achieved at the same optimum $\mathbf{x}^*$.

Relationship between manifold and Euclidean margins. Let $\mathbf{V} \in \mathbb{R}^{d \times r}$ be a orthonormal basis of the r-dimensional embedding subspace. An angle θ between the classifier hyperplane and the embedding subspace describes the relationship between $d_e(F(\cdot))$ and $d_{\mathcal{M}}(F(\cdot))$,

$$\mathbb{E}\big[\sin\theta\big] = \mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right].$$

Denote ω as the angle between $\hat{\mathbf{n}}$ and the embedding subspace, $\theta = \frac{\pi}{2} - \omega$,

$$\sin\theta = \cos\omega = \|\mathbf{V}^T\mathbf{n}\|.$$

Moreover, when the linear classifier forms a random decision boundary, we consider its orthogonal normal vector $\hat{\mathbf{n}} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I})$. Therefore, $\mathbf{V}^T\hat{\mathbf{n}} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{V}^T\mathbf{V})$.

$$\mathbb{E}\big[\|\mathbf{V}^T\hat{\mathbf{n}}\|_2^2\big] = \frac{1}{d}Tr(\mathbf{V}^T\mathbf{V}) = \frac{r}{d}.$$

Then

$$\mathbb{E}\big[(\sin\theta)^2\big] = \mathbb{E}\big[(\cos\omega)^2\big] = \frac{r}{d},$$

and from Jensen's inequality,

$$\big(\mathbb{E}\big[\sin\theta\big]\big)^2 \leq \mathbb{E}\big[(\sin\theta)^2\big].$$

Therefore,

$$\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}.$$

■

3.7.2 Error Estimation of Linear System

This section derives the error estimation of the closed-loop control framework in linear cases. Given a sequence of states $\{\mathbf{x}_t\}_{t=0}^{T-1}$, such that $\mathbf{x}_t \in \mathcal{M}_t$ for all t, we denote $\boldsymbol{\theta}_t$ as the linearized transformation of the nonlinear transformation $F_t(\cdot)$ centered at $\mathbf{x}_t$. We represent the t^{th} embedding manifold $\mathcal{M}_t = f_t^{-1}(\mathbf{0})$, where $f_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$ is a submersion of class $\mathcal{C}^2$. Recall Proposition 3.7.1, the kernel of $f_t'(\mathbf{x}_t)$ is equivalent to $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$, and the orthogonal projection onto $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ (Equation (3.16)) is

$$\mathbf{P}_t := \mathbf{I} - f_t'(\mathbf{x}_t)^T(f_t'(\mathbf{x}_t)f_t'(\mathbf{x}_t)^T)^{-1}f_t'(\mathbf{x}_t),$$

and the orthogonal projection onto orthogonal complement of $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ is

$$\mathbf{Q}_t = \mathbf{I} - \mathbf{P}_t = f_t'(\mathbf{x}_t)^T (f_t'(\mathbf{x}_t) f_t'(\mathbf{x}_t)^T)^{-1} f_t'(\mathbf{x}_t).$$

For simplicity, a orthonormal basis of $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ is denoted as $\mathbf{V}_t \in \mathbb{R}^{d\times d}$, in which case, the orthogonal projection $\mathbf{P}_t = \mathbf{V}_t\mathbf{V}_t^T$, and $\mathbf{Q}_t = \mathbf{I} - \mathbf{V}_t\mathbf{V}_t^T$.

We consider a set of tangent spaces $\{\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t\}_{t=0}^{T-1}$, that is, each $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ is the tangent space of $\mathcal{M}_t$ at $\mathbf{x}_t$. Recall the running loss in Equation (3.2), the linear setting uses projection onto a tangent space rather than a nonlinear embedding manifold.

$$J(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2}\|\mathbf{Q}_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2, \tag{3.11}$$

it measures the magnitude of the controlled state $\mathbf{x}_t + \mathbf{u}_t$ within the orthogonal complement of $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$, and the magnitude of applied control $\mathbf{u}_t$.

The optimal feedback control under Equation (3.11) is defined as

$$\mathbf{u}_t^P(\mathbf{x}_t) = \arg\min_{\mathbf{u}_t} J(\mathbf{x}_t, \mathbf{u}_t),$$

it admits an exact solution by setting the gradient of performance index (Equation (3.11)) to $\mathbf{0}$.

$$\begin{aligned}\nabla_{\mathbf{u}} J(\mathbf{x}_t, \mathbf{u}_t) &= \nabla_{\mathbf{u}}\left(\frac{1}{2}\|\mathbf{Q}_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2\right), \\ &= \mathbf{Q}_t^T\mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t^T\mathbf{Q}_t\mathbf{u}_t + c\cdot\mathbf{u}_t,\end{aligned}$$

which leads to the exact solution of $\mathbf{u}_t^P$ (Equation (3.18)) as

$$\mathbf{u}_t^P = -(c\cdot\mathbf{I} + \mathbf{Q}_t^T\mathbf{Q}_t)^{-1}\mathbf{Q}_t^T\mathbf{Q}_t\mathbf{x}_t = -\mathbf{K}_t\mathbf{x}_t, \tag{3.12}$$

where the feedback gain matrix $\mathbf{K}_t = (c\cdot\mathbf{I} + \mathbf{Q}_t^T\mathbf{Q}_t)^{-1}\mathbf{Q}_t^T\mathbf{Q}_t$. Thus, the one-step feedback

control can be represented as $\mathbf{u}_t^P = -\mathbf{K}_t\mathbf{x}_t$.

Given a sequence $\{\mathbf{x}_t\}_{t=0}^{T-1}$, we denote $\{\mathbf{q}_{\epsilon,t}\}_{t=0}^{T-1}$ as another sequence of states resulted from the linear system, $\mathbf{q}_{\epsilon,0} = \mathbf{x}_0 + \mathbf{z}$, for some perturbation $\mathbf{z}$, and $\{\overline{\mathbf{q}}_{\epsilon,t}\}_{t=0}^{T-1}$ as the adjusted states by the linear control,

$$\begin{aligned}\overline{\mathbf{q}}_{\epsilon,t+1} &= \boldsymbol{\theta}_t(\overline{\mathbf{q}}_{\epsilon,t} + \mathbf{u}_t^P),\\ &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)\overline{\mathbf{q}}_{\epsilon,t}.\end{aligned}$$

The difference between the controlled system applied with perturbation at the initial condition and the uncontrolled system without perturbation is as follows,

$$\begin{aligned}\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\overline{\mathbf{q}}_{\epsilon,t} + \mathbf{u}_t - \mathbf{x}_t),\\ &= \boldsymbol{\theta}_t(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{K}_t\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t).\end{aligned} \tag{3.13}$$

The control objective is to minimize the state components that lie in the orthogonal complement of the tangent space. When the data locates on the embedding manifold, $\mathbf{x}_t \in \mathcal{M}_t$, this results in $\mathbf{Q}_t\mathbf{x}_t = \mathbf{0}$, consequently, its feedback control $\mathbf{K}_t\mathbf{x}_t = \mathbf{0}$. The state difference of Equation (3.13) can be further shown by adding a $\mathbf{0}$ term of $(\boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t)$

$$\begin{aligned}\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)\overline{\mathbf{q}}_{\epsilon,t} - \boldsymbol{\theta}_t\mathbf{x}_t + \boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t,\\ &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t).\end{aligned} \tag{3.14}$$

In the following, we show a transformation on $(\mathbf{I} - \mathbf{K}_t)$ based on its definition.

Lemma 3.7.1 *For $t \geq 0$, we have*

$$\mathbf{I} - \mathbf{K}_t = \alpha \cdot \mathbf{I} + (1 - \alpha) \cdot \mathbf{P}_t,$$

where $\mathbf{P}_t := \mathbf{V}_t^r(\mathbf{V}_t^r)^T$, which is the orthogonal projection onto $Z_{\|}^t$, and $\alpha := \frac{c}{1+c}$ such that

$\alpha \in [0, 1]$.

Proof: Recall that $\mathbf{K}_t = (c \cdot \mathbf{I} + \mathbf{Q}_t^T \mathbf{Q}_t)^{-1} \mathbf{Q}_t^T \mathbf{Q}_t$, and $\mathbf{Q}_t = \mathbf{I} - \mathbf{V}_t^r (\mathbf{V}_t^r)^T$, $\mathbf{Q}_t$ can be diagonalized as following

$$\mathbf{Q}_t = \mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \mathbf{V}_t^T,$$

where the first r diagonal elements have a common value of 0 and the last $(d-r)$ diagonal elements have a common value of 1. Furthermore, the feedback gain matrix $\mathbf{K}_t$ can be diagonalized as

$$\mathbf{K}_t = \mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \frac{1}{1+c} & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{1+c} \end{bmatrix} \mathbf{V}_t^T,$$

where the last $(d-r)$ diagonal elements have a common value of $\frac{1}{1+c}$. The control term $(\mathbf{I}-\mathbf{K}_t)$ thus can be represented as

$$\mathbf{I} - \mathbf{K}_t = \mathbf{V}_t \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \frac{c}{1+c} & 0 \\ 0 & 0 & \cdots & 0 & \frac{c}{1+c} \end{bmatrix} \mathbf{V}_t^T,$$

where the first r diagonal elements have common value of 1 and the last $(d-r)$ diagonal

elements have common value of $\frac{c}{1+c}$. By denoting the projection of first r columns as $\mathbf{V}_t^r$ and last $(d-r)$ columns as $\hat{\mathbf{V}}_t^r$, it can be further shown as

$$\begin{aligned}
\mathbf{I} - \mathbf{K}_t &= \mathbf{V}_t^r(\mathbf{V}_t^r)^T + \frac{c}{1+c}\big(\hat{\mathbf{V}}_t^r(\hat{\mathbf{V}}_t^r)^T\big), \\
&= \mathbf{P}_t + \alpha\big(\mathbf{I} - \mathbf{P}_t\big), \\
&= \alpha \cdot \mathbf{I} + (1-\alpha) \cdot \mathbf{P}_t.
\end{aligned}$$

■

Lemma 3.7.2 *Define for $t \geq 0$*

$$\begin{cases}
\mathbf{P}_t^0 := \mathbf{P}_t, \\
\mathbf{P}_t^{(s+1)} := \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}, \quad s = 0, 1, \ldots, t-1,
\end{cases}$$

for $0 \leq s \leq t$. Then

1. *$\mathbf{P}_t^s$ is a projection.*
2. *$\mathbf{P}_t^s$ is a projection onto $Z_{\|}^{t-s}$, i.e. range$(\mathbf{P}_t^s) = Z_{\|}^{t-s}$.*

Proof:

1. We prove it by induction on s for each t. For $s = 0$, $\mathbf{P}_t^0 = \mathbf{P}_t$, which is a projection by its definition. Suppose it is true for s such that $\mathbf{P}_t^s = \mathbf{P}_t^s\mathbf{P}_t^s$, then for $(s+1)$,

$$\begin{aligned}
(\mathbf{P}_t^{s+1})^2 &= \big(\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}\big)^2, \\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\big(\mathbf{P}_t^s\big)^2\boldsymbol{\theta}_{t-s-1}, \\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}, \\
&= \mathbf{P}_t^{s+1}.
\end{aligned}$$

2. We prove it by induction on s for each t. For $s = 0$, $\mathbf{P}_t^0 = \mathbf{P}_t$, which is the orthogonal projection onto $Z_{\|}^t$. Suppose that it is true for s such that $\mathbf{P}_t^s$ is a projection onto $Z_{\|}^{t-s}$, then for $(s+1)$, $\mathbf{P}_t^{s+1} = \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}$, which implies

$$\begin{aligned} range(\mathbf{P}_t^{s+1}) &= range(\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s), \\ &= \{\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{x} : \mathbf{x} \in Z_{\|}^{t-s}\}, \\ &= Z_{\|}^{t-s-1}. \end{aligned}$$

■

The following Lemma reformulates the state difference equation.

Lemma 3.7.3 *Define for* $0 \leq s \leq t$,

$$\mathbf{G}_t^s := \alpha \cdot \mathbf{I} + (1-\alpha)\mathbf{P}_t^s.$$

The state difference equation, $\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t)$*, can be written as*

$$\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t = (\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \;\; t \geq 1.$$

Proof: We prove it by induction on t. For $t = 1$,

$$\begin{aligned} \overline{\mathbf{q}}_{\epsilon,1} - \mathbf{x}_1 &= \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \\ &= \boldsymbol{\theta}_0(\alpha \cdot \mathbf{I} + (1-\alpha)\cdot\mathbf{P}_0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \qquad \text{Lemma 3.7.1,} \\ &= \boldsymbol{\theta}_0\mathbf{G}_0^0(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0). \end{aligned}$$

Recall the definitions of $\mathbf{P}_t^{(s+1)} := \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}$, and $\mathbf{G}_t^s := \alpha\cdot\mathbf{I} + (1-\alpha)\mathbf{P}_t^s$,

$$\begin{aligned}
\mathbf{G}_t^{s+1} &= \alpha\cdot\mathbf{I} + (1-\alpha)\cdot\mathbf{P}_t^{(s+1)},\\
&= \alpha\cdot\mathbf{I} + (1-\alpha)\cdot\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1},\\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\big(\alpha\cdot\mathbf{I} + (1-\alpha)\cdot\mathbf{P}_t^s\big)\boldsymbol{\theta}_{t-s-1},\\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{G}_t^s\boldsymbol{\theta}_{t-s-1},
\end{aligned}$$

which results in $\boldsymbol{\theta}_{t-s-1}\mathbf{G}_t^{(s+1)} = \mathbf{G}_t^s\boldsymbol{\theta}_{t-s-1}$. Suppose that it is true for $(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t)$,

$$\begin{aligned}
\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t),\\
&= \boldsymbol{\theta}_t(\alpha\cdot\mathbf{I} + (1-\alpha)\cdot\mathbf{P}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t), && \text{Lemma 3.7.1,}\\
&= \boldsymbol{\theta}_t\mathbf{G}_t^0(\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0),\\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1})\mathbf{G}_t^1(\boldsymbol{\theta}_{t-2}\boldsymbol{\theta}_{t-3}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0),\\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_t^t\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0).
\end{aligned}$$

■

Lemma 3.7.4 *For $t \geq 1$,*

$$\mathbf{G}_{t-1}^{(t-1)}\mathbf{G}_{t-2}^{(t-2)}\cdots\mathbf{G}_0^0 = \alpha^t\cdot\mathbf{I} + (1-\alpha)\sum_{s=0}^{t-1}\alpha^s\mathbf{P}_s^s.$$

Proof: We prove it by induction on t. Recall the definition of $\mathbf{G}_t^s := \alpha\cdot\mathbf{I} + (1-\alpha)\cdot\mathbf{P}_t^s$. When $t = 1$,

$$\mathbf{G}_0^0 = \alpha\cdot\mathbf{I} + (1-\alpha)\cdot\mathbf{P}_0^0.$$

Suppose that it is true for t such that

$$\mathbf{G}_{t-1}^{(t-1)}\mathbf{G}_{t-2}^{(t-2)}\cdots\mathbf{G}_0^0 = \alpha^t\cdot\mathbf{I} + (1-\alpha)\sum_{s=0}^{t-1}\alpha^s\mathbf{P}_s^s,$$

for $(t+1)$,

$$\begin{aligned}
&\mathbf{G}_t^t \mathbf{G}_{t-1}^{(t-1)} \cdots \mathbf{G}_0^0 \\
&= \mathbf{G}_t^t (\alpha^t \cdot \mathbf{I} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s \mathbf{P}_s^s), \\
&= (\alpha \cdot \mathbf{I} + (1-\alpha) \cdot \mathbf{P}_t^t)(\alpha^t \cdot \mathbf{I} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s \mathbf{P}_s^s), \\
&= \alpha^{t+1} \cdot \mathbf{I} + \alpha^t (1-\alpha) \mathbf{P}_t^t + (1-\alpha)^2 \sum_{s=0}^{t-1} \alpha^s \cdot \mathbf{P}_t^t \mathbf{P}_s^s + \alpha(1-\alpha) \sum_{s=0}^{t-1} \alpha^s \cdot \mathbf{P}_s^s.
\end{aligned}$$

Recall Lemma 3.7.2, $range(\mathbf{P}_t^t) = range(\mathbf{P}_s^s) = Z_{\|}^0$. Since $\mathbf{P}_t^t$ and $\mathbf{P}_s^s$ are projections onto the same space, $\mathbf{P}_t^t \mathbf{P}_s^s = \mathbf{P}_s^s$. Therefore,

$$\begin{aligned}
\mathbf{G}_t^t \mathbf{G}_{t-1}^{(t-1)} \cdots \mathbf{G}_0^0 &= \alpha^{t+1} \cdot \mathbf{I} + \alpha^t (1-\alpha) \cdot \mathbf{P}_t^t + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s \cdot \mathbf{P}_s^s, \\
&= \alpha^{t+1} \cdot \mathbf{I} + (1-\alpha) \sum_{s=0}^{t} \alpha^s \cdot \mathbf{P}_s^s.
\end{aligned}$$

■

Lemma 3.7.5 *Let $\mathbf{P} = \mathbf{V}\mathbf{V}^T$ be the orthogonal projection onto a subspace $\mathcal{D}$, and $\boldsymbol{\theta}$ to be invertible. Denote by $\hat{\mathbf{P}}$ the orthogonal projection onto $\boldsymbol{\theta}\mathcal{D} := \{\boldsymbol{\theta}\mathbf{x} : \mathbf{x} \in \mathcal{D}\}$. Then*

$$\|\boldsymbol{\theta}^{-1} \hat{\mathbf{P}} \boldsymbol{\theta} - \mathbf{P}\|_2 \leq \left(1 + \kappa(\boldsymbol{\theta})^2\right) \cdot \|\mathbf{I} - \boldsymbol{\theta}^T \boldsymbol{\theta}\|_2.$$

Proof:

$$\begin{aligned}
\hat{\mathbf{P}} &= \boldsymbol{\theta}\mathbf{V} \left[(\boldsymbol{\theta}\mathbf{V})^T (\boldsymbol{\theta}\mathbf{V})\right]^{-1} (\boldsymbol{\theta}\mathbf{V})^T, \\
&= \boldsymbol{\theta}\mathbf{V} \left[\mathbf{V}^T \boldsymbol{\theta}^T \boldsymbol{\theta} \mathbf{V}\right]^{-1} \mathbf{V}^T \boldsymbol{\theta}^T.
\end{aligned}$$

Furthermore, the difference between the oblique projection and the orthogonal projection can

be bounded by the following

$$
\begin{aligned}
\|\boldsymbol{\theta}^{-1}\hat{\mathbf{P}}\boldsymbol{\theta} - \mathbf{P}\|_2 &= \|\mathbf{V}\big[\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}\big]^{-1}\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{V}\mathbf{V}^T\|_2, \\
&\leq \|\mathbf{V}\big[\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}\big]^{-1}\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{V}\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\mathbf{V}\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{V}\mathbf{V}^T\|_2, \\
&\leq \|\mathbf{V}\big([\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}]^{-1} - \mathbf{I}\big)\mathbf{V}^T\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&\leq \|[\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}]^{-1}\|_2 \cdot \|\mathbf{I} - \mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&\leq \|[\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}]^{-1}\|_2 \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&= \big(\lambda_{min}(\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V})\big)^{-1} \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&= \big(\inf_{\|\mathbf{x}\|_2=1} \mathbf{x}^T\mathbf{V}^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{V}\mathbf{x}\big)^{-1} \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&\leq \big(\inf_{\|\mathbf{x}'\|_2=1} (\mathbf{x}')^T\boldsymbol{\theta}^T\boldsymbol{\theta}\mathbf{x}'\big)^{-1} \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&= \big(\lambda_{min}(\boldsymbol{\theta}^T\boldsymbol{\theta})\big)^{-1} \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&= \|(\boldsymbol{\theta}^T\boldsymbol{\theta})^{-1}\|_2 \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 \cdot \|\boldsymbol{\theta}^T\boldsymbol{\theta}\|_2 + \|\boldsymbol{\theta}^T\boldsymbol{\theta} - \mathbf{I}\|_2, \\
&= \big(1 + \kappa(\boldsymbol{\theta})^2\big) \cdot \|\mathbf{I} - \boldsymbol{\theta}^T\boldsymbol{\theta}\|_2.
\end{aligned}
$$

■

Corollary 3.7.1 *Let $t \geq 1$. Then for each $s = 0, 1, \cdots, t$, we have*

$$\|\mathbf{P}_s^s - \mathbf{P}_0\|_2 \leq \big(1 + \kappa(\overline{\boldsymbol{\theta}}_s)^2\big) \cdot \|\mathbf{I} - \overline{\boldsymbol{\theta}}_s^T\overline{\boldsymbol{\theta}}_s\|_2,$$

where

- $\overline{\boldsymbol{\theta}}_s := \boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0,\ s \geq 1,$
- $\overline{\boldsymbol{\theta}}_s := \mathbf{I},\ s = 0.$

The following theorem provides an error estimation for the linear dynamic system with linear controls.

Theorem 3.4.1 *For $t \geq 1$, we have an error estimation for the linear system*

$$\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 \leq \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \Big(\alpha^{2t} \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + \gamma_t \|\mathbf{z}\|_2^2 \big(\gamma_t \alpha^2 (1 - \alpha^{t-1})^2 + 2(\alpha - \alpha^t) \big) \Big).$$

where $\gamma_t := \max\limits_{s \leq t} \big(1 + \kappa(\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)^2\big) \|\mathbf{I} - (\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)^T (\boldsymbol{\theta}_{s-1} \cdots \boldsymbol{\theta}_0)\|_2$, $\kappa(\boldsymbol{\theta})$ is condition number of $\boldsymbol{\theta}$, $\alpha = \frac{c}{1+c}$, and c represents the control regularization. In particular, the equality

$$\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 = \alpha^{2t} \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2$$

holds when all $\boldsymbol{\theta}_t$ are orthogonal.

Proof: The input perturbation $\mathbf{z} = \overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0$ can be written as $\mathbf{z} = \mathbf{z}^{\|} + \cdot \mathbf{z}^{\perp}$, where $\mathbf{z}^{\|} \in Z_{\|}$ and $\mathbf{z}^{\perp} \in Z_{\perp}$, where $\mathbf{z}^{\|}$ and $\mathbf{z}^{\perp}$ are vectors such that

- $\mathbf{z}^{\|} \cdot \mathbf{z}^{\perp} = 0$ almost surely.
- $\mathbf{z}^{\|}$, $\mathbf{z}^{\perp}$ have uncorrelated components.

Recall Lemma 3.7.3,

$$\begin{aligned} \|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 &= \|(\boldsymbol{\theta}_{t-1} \boldsymbol{\theta}_{t-2} \cdots \boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2, \\ &\leq \|\boldsymbol{\theta}_{t-1} \boldsymbol{\theta}_{t-2} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \|(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2, \end{aligned} \tag{3.15}$$

For the term $\|(\mathbf{G}_{t-1}^{t-1} \mathbf{G}_{t-2}^{t-2} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2$, recall Lemma 3.7.4,

$$\begin{aligned} \|(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2 &= \|\Big(\alpha^t \cdot \mathbf{I} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s \cdot \mathbf{P}_s^s\Big)\mathbf{z}\|_2^2, \\ &= \|\alpha^t \mathbf{z} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s \mathbf{P}_0 \mathbf{z} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z}\|_2^2, \\ &= \|\alpha^t \mathbf{z} + (1-\alpha^t)\mathbf{z}^{\|} + (1-\alpha) \sum_{s=0}^{t-1} \alpha^s (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z}\|_2^2, \end{aligned}$$

in the above, $\mathbf{P}_0$ is an orthogonal projection on $t = 0$ (input data space), therefore, $\mathbf{P}_0\mathbf{z} = \mathbf{z}^{\|}$. Furthermore, when $s = 0$, $\mathbf{P}_s^s - \mathbf{P}_0 = \mathbf{0}$. Thus,

$$
\begin{aligned}
&\|(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2 \\
&= \alpha^{2t}\|\mathbf{z}\|_2^2 + (1-\alpha^t)^2\|\mathbf{z}^{\|}\|_2^2 + (1-\alpha)^2 \sum_{s,q=1}^{t-1} \alpha^s \alpha^q \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)^T (\mathbf{P}_q^q - \mathbf{P}_0)\mathbf{z} \\
&\quad + 2\alpha^t(1-\alpha^t)\|\mathbf{z}^{\|}\|_2^2 + 2\alpha^t(1-\alpha) \sum_{s=1}^{t-1} \alpha^s \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z} \\
&\quad + 2(1-\alpha^t)(1-\alpha) \sum_{s=1}^{t-1} \alpha^s (\mathbf{z}^{\|})^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z}, \\
&= \alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \left(\alpha^{2t} + 2\alpha^t(1-\alpha^t) + (1-\alpha^t)^2\right)\|\mathbf{z}^{\|}\|_2^2 \\
&\quad + (1-\alpha)^2 \sum_{s,q=1}^{t-1} \alpha^s \alpha^q \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)^T (\mathbf{P}_q^q - \mathbf{P}_0)\mathbf{z} + 2\alpha^t(1-\alpha) \sum_{s=1}^{t-1} \alpha^s \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z} \\
&\quad + 2(1-\alpha^t)(1-\alpha) \sum_{s=1}^{t-1} \alpha^s (\mathbf{z}^{\|})^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z}, \\
&= \alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + (1-\alpha)^2 \sum_{s,q=1}^{t-1} \alpha^s \alpha^q \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)^T (\mathbf{P}_q^q - \mathbf{P}_0)\mathbf{z} \\
&\quad + 2\alpha^t(1-\alpha) \sum_{s=1}^{t-1} \alpha^s \mathbf{z}^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z} + 2(1-\alpha^t)(1-\alpha) \sum_{s=1}^{t-1} \alpha^s (\mathbf{z}^{\|})^T (\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z}.
\end{aligned}
$$

Using Corollary 3.7.1, we have

- $$
\begin{aligned}
\mathbf{z}^T(\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z} &\leq \|\mathbf{z}\|_2^2 \cdot \|\mathbf{P}_s^s - \mathbf{P}_0\|, \\
&\leq \gamma_t \|\mathbf{z}\|_2^2.
\end{aligned}
$$

- $$\begin{aligned}\mathbf{z}^T(\mathbf{P}_s^s - \mathbf{P}_0)^T(\mathbf{P}_q^q - \mathbf{P}_0)\mathbf{z} &\leq \|\mathbf{z}\|_2^2 \cdot \|\mathbf{P}_s^s - \mathbf{P}_0\| \cdot \|\mathbf{P}_q^q - \mathbf{P}_0\|, \\ &\leq \gamma_t^2 \|\mathbf{z}\|_2^2.\end{aligned}$$

- $$\begin{aligned}(\mathbf{z}^{\|})^T(\mathbf{P}_s^s - \mathbf{P}_0)\mathbf{z} &\leq \gamma_t \|\mathbf{z}^{\|}\|_2 \cdot \|\mathbf{z}\|_2, \\ &\leq \gamma_t \|\mathbf{z}\|_2^2.\end{aligned}$$

Thus, we have

$$\begin{aligned}\|(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2 &\leq \alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + \alpha^2(1-\alpha^{t-1})^2\gamma_t^2\|\mathbf{z}\|_2^2 + 2\alpha^{t+1}(1-\alpha^{t-1})\gamma_t\|\mathbf{z}\|_2^2 \\ &\quad + 2\alpha(1-\alpha^t)(1-\alpha^{t-1})\gamma_t\|\mathbf{z}\|_2^2, \\ &= \alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + \gamma_t\|\mathbf{z}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big).\end{aligned}$$

Recall the error estimation in Equation (3.15),

$$\begin{aligned}\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 &\leq \|\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0\|_2^2 \cdot \|(\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)\mathbf{z}\|_2^2, \\ &\leq \|\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0\|_2^2 \cdot \left(\alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + \gamma_t\|\mathbf{z}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha-\alpha^t)\big)\right).\end{aligned}$$

In the specific case, when all $\boldsymbol{\theta}_t$ are orthogonal,

$$\begin{aligned}\gamma_t :&= \max_{s\leq t}\big(1+\kappa(\overline{\boldsymbol{\theta}}_s)^2\big)\|\mathbf{I} - \overline{\boldsymbol{\theta}}_s^T\overline{\boldsymbol{\theta}}_s\|_2 \\ &= 0.\end{aligned}$$

Thus,

$$\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 = \alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2.$$

■

3.7.3 Analysis On Nonlinear Manifold Projection

Definition for the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ based on the submersion $f(\cdot)$.

Proposition 3.7.1 *Let $\mathcal{M} \subset \mathbb{R}^d$ be an r-dimensional smooth manifold and $\mathbf{x} \in \mathcal{M}$. Given a submersion $f(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$ of class $\mathcal{C}^1$, such that $\mathcal{M} = f^{-1}(\mathbf{0})$. Then the tangent space at any $\mathbf{x} \in \mathcal{M}$ is the kernel of the linear map $f'(\mathbf{x})$, i.e., $\mathcal{T}_{\mathbf{x}}\mathcal{M} = \mathrm{Ker} f'(\mathbf{x})$.*

Proof: For any $\mathbf{x} \in \mathcal{M}$ and $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$, suppose that there is an open interval $\mathcal{J} \in \mathbb{R}$ such that $0 \in \mathcal{J}$, and a smooth curve $\gamma : \mathcal{J} \to \mathcal{M}$ such that $\gamma(0) = \mathbf{x}$, $\gamma'(0) = \mathbf{v}$. Since $f(\mathbf{x}) = \mathbf{0},\ \forall \mathbf{x} \in \mathcal{M}$, and $\gamma(\lambda) \in \mathcal{M},\ \forall \lambda \in \mathcal{J}$,

$$f \circ \gamma(\lambda) = \mathbf{0},\ \lambda \in \mathcal{J}.$$

Therefore, $f \circ \gamma(\lambda)$ is a constant map for all $\lambda \in \mathcal{J}$,

$$\mathbf{0} = (f \circ \gamma)'(0) = f'(\gamma(0))\gamma'(0) = f'(\mathbf{x})\mathbf{v},$$

since $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ is arbitrarily chosen from $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, $f'(\mathbf{x})\mathbf{v} = \mathbf{0},\ \forall \mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$. Therefore, $\mathcal{T}_{\mathbf{x}}\mathcal{M} \in ker f'(\mathbf{x})$ (the kernel of linear map $f'(\mathbf{x})$).

Recall that $f : \mathbb{R}^d \to \mathbb{R}^{d-r}$ is a submersion, its differential $f'(\mathbf{x})$ is a surjective linear map with constant rank for all $\mathbf{x} \in \mathcal{M}$.

$$dim(ker f'(\mathbf{x})) = dim(\mathbb{R}^r) - rank(f'(\mathbf{x})) = d - (d - r) = r.$$

Since $\mathcal{T}_{\mathbf{x}}\mathcal{M} \in ker f'(\mathbf{x})$ and $dim(\mathcal{T}_{\mathbf{x}}\mathcal{M}) = dim(ker f'(\mathbf{x}))$, $\mathcal{T}_{\mathbf{x}}\mathcal{M} = ker f'(\mathbf{x})$. ■

Definitions for the control solutions of running loss. Given a smooth manifold $\mathcal{M}$, we can attach to every point $\mathbf{x} \in \mathcal{M}$ a tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. Proposition 3.7.1 has shown the equivalence between the kernel of $f'(\mathbf{x})$ and the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. Therefore, $f'(\mathbf{x})$ consists a basis of the complement of the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$. For simplicity, we assume the submersion to be normalized such that the columns of $f'(\mathbf{x})$ consist of a orthonormal basis. In this case, the orthogonal projection onto $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ can be defined as following,

$$\mathbf{P}_{\mathbf{x}} := \mathbf{I} - f'(\mathbf{x})^T f'(\mathbf{x}). \tag{3.16}$$

In general cases, when $f'(\mathbf{x})$ does not consist of an orthonormal basis, the orthogonal projection in Equation (3.16) can be defined by adding a scaling factor as following,

$$\mathbf{P}_{\mathbf{x}} := \mathbf{I} - f'(\mathbf{x})^T (f'(\mathbf{x}) f'(\mathbf{x})^T)^{-1} f'(\mathbf{x}).$$

The orthogonal projection onto the orthogonal complement of $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ is defined as follows,

$$\mathbf{Q}_{\mathbf{x}} := \mathbf{I} - \mathbf{P}_{\mathbf{x}} = f'(\mathbf{x})^T f'(\mathbf{x}).$$

Recall that a general embedding manifold is defined by a submersion, such that $\mathcal{M} = f^{-1}(\mathbf{0})$. In the linear case, an embedding manifold is considered as a linear sub-space. This linear sub-space can be defined by a submersion $\mathcal{M} = (f'(\mathbf{x}))^{-1}\mathbf{0} = f'(\mathbf{x})^T\mathbf{0}$, in which case, the submersion is a linear operator $f'(\mathbf{x})$. In this linear case, we denote $\mathbf{u}_{\mathbf{x}}^P(\mathbf{x}_\epsilon)$ as the minimizer of running loss $\mathcal{L}(\mathbf{x}_\epsilon, \mathbf{u}, \mathcal{E}(\cdot))$ in Equation (3.2),

$$\mathbf{u}_{\mathbf{x}}^P(\mathbf{x}_\epsilon) = \arg\min_{\mathbf{u}\in\mathbb{R}^d} \frac{1}{2} \cdot \|f'(\mathbf{x})(\mathbf{x}_\epsilon + \mathbf{u})\|_2^2 + \frac{c}{2} \cdot \|\mathbf{u}\|_2^2. \tag{3.17}$$

Notice $(\mathbf{x}_\epsilon + \mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon)) = \mathbf{P}_\mathbf{x}(\mathbf{x}_\epsilon)$ when the regularization $c = 0$, $\mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon)$ admits an exact solution

$$\mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon) = -(c \cdot \mathbf{I} + \mathbf{Q}_\mathbf{x})^{-1}\mathbf{Q}_\mathbf{x}\mathbf{x}_\epsilon = -(c \cdot \mathbf{I} + f'(\mathbf{x})^T f'(\mathbf{x}))^{-1} f'(\mathbf{x})^T f'(\mathbf{x})\mathbf{x}_\epsilon. \tag{3.18}$$

In the nonlinear case, let $\mathcal{M} \subset \mathbb{R}^d$ be an embedding manifold such that $\mathcal{M} = f^{-1}(\mathbf{0})$, for a submersion $f(\cdot)$ of class $\mathcal{C}^2$, a constant σ be a uniform upper bound on the Hessian of $f(\cdot)$, such that $\sup_{\mathbf{x}\in\mathbb{R}^d} \|f''(\mathbf{x})\|_* \leq \sigma$. For simplicity, we assume a normalized submersion $f(\cdot)$ to be where $f'(\mathbf{x})$ is a orthonormal basis for the orthogonal complement of tangent space at $\mathbf{x} \in \mathcal{M}$. In this case, we denote $\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon)$ as the minimizer of the running loss $\mathcal{L}(\mathbf{x}_\epsilon, \mathbf{u}, \mathcal{E}(\cdot))$ in Equation (3.2),

$$\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon) = \arg\min_{\mathbf{u}\in\mathbb{R}^d} \frac{1}{2} \cdot \|f(\mathbf{x}_\epsilon + \mathbf{u})\|_2^2 + \frac{c}{2} \cdot \|\mathbf{u}\|_2^2. \tag{3.19}$$

In general, when the submersion is not normalized, we can always normalize it by replacing $f(\mathbf{x})$ as $f'(\mathbf{x})^T(f'(\mathbf{x})f'(\mathbf{x})^T)^{-1}f(\mathbf{x})$, where $f'(\mathbf{x})^T(f'(\mathbf{x})f'(\mathbf{x})^T)^{-1}$ is a scaling factor.

Error bound for linear and nonlinear control solutions. For a 3-dimensional tensor, e.g. the Hessian $f''(\mathbf{x})$, we define the 2-norm of $f''(\mathbf{x})$ as

$$\|f''(\mathbf{x})\|_* := \sup_{\mathbf{z}\neq\mathbf{0}} \frac{\|f''(\mathbf{x})^{i,j,k}\mathbf{z}_j\mathbf{z}_k\|_2}{\|\mathbf{z}\|_2^2}.$$

The following proposition shows an error bound between $\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon)$ and $\mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon)$.

Proposition 3.7.2 *Consider a data point $\mathbf{x}_\epsilon = \mathbf{x} + \epsilon \cdot \mathbf{v}$, where $\mathbf{x} \in \mathcal{M}$, $\|\mathbf{v}\|_2 = 1$ and ϵ sufficiently small $0 \leq \epsilon \leq 1$. The difference between the regularized manifold projection $\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon)$ and the regularized tangent space projection $\mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon)$ is upper bounded as following,*

$$\|\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon) - \mathbf{u}_\mathbf{x}^P(\mathbf{x}_\epsilon)\|_2 \leq 4\epsilon^2\sigma(1 + 2\sigma).$$

Proof: Recall the definition of regularized manifold projection in Equation (3.19), the optimal solution $\mathbf{u}^\mathcal{M}(\mathbf{x}_\epsilon)$ admits a exact solution by setting the gradient of Equation (3.19) to

$\mathbf{0}$,

$$\nabla_{\mathbf{u}}\left(\frac{1}{2}\cdot\|f(\mathbf{x}_\epsilon+\mathbf{u})\|_2^2+\frac{c}{2}\cdot\|\mathbf{u}\|_2^2\right)=\Big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\Big)^T\Big(f(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\Big)+c\cdot\mathbf{u}. \tag{3.20}$$

The control $\mathbf{u}$ is in the same order as the perturbation magnitude ϵ, we parametrize $\mathbf{u}=\epsilon\cdot\boldsymbol{\mu}$. By applying Taylor series expansion centered at $\epsilon=0$, and $f(\mathbf{x})=\mathbf{0}$ since $\mathbf{x}\in\mathcal{M}$,

$$\begin{aligned}
&\Big(f'(\mathbf{x}+\epsilon\mathbf{v}+\epsilon\boldsymbol{\mu})\Big)^T\Big(f(\mathbf{x}+\epsilon\mathbf{v}+\epsilon\boldsymbol{\mu})\Big)+c\cdot\epsilon\cdot\boldsymbol{\mu}\\
&=\Big(f'(\mathbf{x})+\epsilon\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\mathbf{v}+\boldsymbol{\mu})_k\big)\Big)^T\Big(\epsilon f'(\mathbf{x})(\mathbf{v}+\boldsymbol{\mu})+\epsilon^2\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\mathbf{v}+\boldsymbol{\mu})_j(\mathbf{v}+\boldsymbol{\mu})_k\big)\Big)\\
&\quad+c\cdot\epsilon\cdot\boldsymbol{\mu},
\end{aligned}$$

since $\boldsymbol{\mu}$ is a variable dependent on $\mathbf{u}$, the Hessian of $f(\cdot)$ is a function that depends on $\boldsymbol{\mu}$. There exists a $\mathbf{x}^{\boldsymbol{\mu}}$ satisfying the following,

$$f(\mathbf{x}+\epsilon\mathbf{v}+\epsilon\boldsymbol{\mu})=f(\mathbf{x})+\epsilon f'(\mathbf{x})(\mathbf{v}+\boldsymbol{\mu})+f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\mathbf{v}+\boldsymbol{\mu})_j(\mathbf{v}+\boldsymbol{\mu})_k.$$

Furthermore, recall that $\mathbf{u}=\epsilon\cdot\boldsymbol{\mu}$,

$$\begin{aligned}
&\Big(f'(\mathbf{x})+\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_k\big)\Big)^T\Big(f'(\mathbf{x})(\epsilon\mathbf{v}+\mathbf{u})+\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_j(\epsilon\mathbf{v}+\mathbf{u})_k\big)\Big)+c\cdot\mathbf{u},\\
&=f'(\mathbf{x})^Tf'(\mathbf{x})(\epsilon\mathbf{v}+\mathbf{u})+c\cdot\mathbf{u}+f'(\mathbf{x})^T\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_j(\epsilon\mathbf{v}+\mathbf{u})_k\big)\\
&\quad+\Big(\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_k\big)\Big)^T\Big(f'(\mathbf{x})(\epsilon\mathbf{v}+\mathbf{u})+\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_j(\epsilon\mathbf{v}+\mathbf{u})_k\big)\Big).
\end{aligned}$$

Setting the above to $\mathbf{0}$ results in an implicit solution for $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$,

$$\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)=-\Big(f'(\mathbf{x})^Tf'(\mathbf{x})+c\mathbf{I}\Big)^{-1}\Big(\epsilon f'(\mathbf{x})^Tf'(\mathbf{x})\mathbf{v}+\mathbf{E}_1+\mathbf{E}_2\Big),$$

where

$$\mathbf{E}_1 = f'(\mathbf{x})^T \big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k} (\epsilon \mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon))_j (\epsilon \mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon))_k \big),$$
$$\mathbf{E}_2 = \Big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k} (\epsilon \mathbf{v} + \mathbf{u})_k \Big)^T \Big(f'(\mathbf{x})(\epsilon \mathbf{v} + \mathbf{u}) + f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k} (\epsilon \mathbf{v} + \mathbf{u})_j (\epsilon \mathbf{v} + \mathbf{u})_k \Big).$$

Note that $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$ is an implicit solution since $\mathbf{E}_1$ and $\mathbf{E}_2$ both depend on the solution $\mathbf{u}$. Recall the definition of $\mathbf{u}^P_{\mathbf{x}}(\mathbf{x}_\epsilon)$ in Equation (3.18),

$$\begin{aligned} \mathbf{u}^P_{\mathbf{x}}(\mathbf{x}_\epsilon) &= -(c \cdot \mathbf{I} + \mathbf{Q}_{\mathbf{x}})^{-1} \mathbf{Q}_{\mathbf{x}} \mathbf{x}_\epsilon, \\ &= -(c \cdot \mathbf{I} + \mathbf{Q}_{\mathbf{x}})^{-1} \mathbf{Q}_{\mathbf{x}} (\mathbf{x} + \epsilon \cdot \mathbf{v}), \\ &= -\epsilon \Big(c \cdot \mathbf{I} + f'(\mathbf{x})^T f'(\mathbf{x}) \Big)^{-1} f'(\mathbf{x})^T f'(\mathbf{x}) \mathbf{v}, \end{aligned}$$

the difference between $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$ and $\mathbf{u}^P_{\mathbf{x}}(\mathbf{x}_\epsilon)$,

$$\|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon) - \mathbf{u}^P_{\mathbf{x}}(\mathbf{x}_\epsilon)\|_2 \leq \| \big(f'(\mathbf{x})^T f'(\mathbf{x}) + c \cdot \mathbf{I} \big)^{-1} \|_2 \cdot \|\mathbf{E}_1 + \mathbf{E}_2\|_2.$$

Let us simplify the above inequality.

- For any non-negative c,

$$\| \big(f'(\mathbf{x})^T f'(\mathbf{x}) + c \cdot \mathbf{I} \big)^{-1} \|_2 = \| \big(f'(\mathbf{x})^T f'(\mathbf{x}) + c \cdot \mathbf{I} \big)^{-1} \|_2 \leq 1.$$

- Recall the gradient of the running loss (Equation (3.20)),

$$\Big(f'(\mathbf{x} + \epsilon \mathbf{v} + \epsilon \boldsymbol{\mu}) \Big)^T \Big(f(\mathbf{x} + \epsilon \mathbf{v} + \epsilon \boldsymbol{\mu}) \Big) + c \cdot \epsilon \cdot \boldsymbol{\mu} = \Big(f'(\mathbf{x} + \epsilon \mathbf{v} + \mathbf{u}) \Big)^T \Big(f'(\mathbf{p})(\epsilon \mathbf{v} + \mathbf{u}) \Big) + c \cdot \mathbf{u},$$

  where $\mathbf{p} = \alpha \mathbf{x} + (1 - \alpha)(\mathbf{x} + \epsilon \mathbf{v} + \mathbf{u}^{\mathcal{M}})$ for $\alpha \in [0, 1]$ such that

$$f(\mathbf{x} + \epsilon \mathbf{v} + \epsilon \boldsymbol{\mu}) = f(\mathbf{x}) + \epsilon \cdot f'(\mathbf{p})(\epsilon \mathbf{v} + \epsilon \boldsymbol{\mu}).$$

Setting the gradient of running loss to $\mathbf{0}$ results in the optimal solution $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$,

$$\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon) = -\Big(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p}) + c\mathbf{I}\Big)^{-1}\Big(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p})\Big)(\epsilon\mathbf{v}).$$

Since $f'(\cdot)$ contains orthonormal basis, the solution $\|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|$ can be upper bounded by the follows,

$$\begin{aligned}
\|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\| &\leq \Big\|\Big(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p}) + c\mathbf{I}\Big)^{-1}\Big\|_2 \cdot \Big\|\Big(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p})\Big)\Big\|_2(\epsilon), \\
&\leq \big\|\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p})\big\|_2^2 \cdot (\epsilon), \\
&\leq \|f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})^T\|_2^2 \cdot \|f'(\mathbf{p})\|_2^2 \cdot (\epsilon), \\
&\leq \epsilon. \qquad (3.21)
\end{aligned}$$

- From above,

$$\|\epsilon\mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^2 = \|\epsilon\mathbf{v}\|_2^2 + 2\|\epsilon\mathbf{v}\|_2 \cdot \|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2 + \|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^2 \leq 4\epsilon^2,$$

$$\|\epsilon\mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^3 \leq 8\epsilon^3.$$

- Recall the $f'(\mathbf{x})$ is a orthnormal basis, $\|f'(\mathbf{x})\|_2 \leq 1$, the error terms can be bounded as follows,

$$\begin{aligned}
\|\mathbf{E}_1\|_2 &= \|f'(\mathbf{x})^T\big(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon))_j(\epsilon\mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon))_k\big)\|_2, \\
&\leq \|\epsilon\mathbf{v} + \mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^2 \cdot \|f''(\mathbf{x}^{\boldsymbol{\mu}})\|_* \cdot \|f'(\mathbf{x})^T\|_2, \\
&\leq 4\epsilon^2.
\end{aligned}$$

$$\begin{aligned}\|\mathbf{E}_2\|_2 &= \left\| \left(f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_k \right)^T \left(f'(\mathbf{x})(\epsilon\mathbf{v}+\mathbf{u}) + f''(\mathbf{x}^{\boldsymbol{\mu}})^{i,j,k}(\epsilon\mathbf{v}+\mathbf{u})_j(\epsilon\mathbf{v}+\mathbf{u})_k \right) \right\|_2 \\ &\leq \|\epsilon\mathbf{v}+\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^2 \cdot \|f''(\mathbf{x}^{\boldsymbol{\mu}})\|_* \cdot \|f'(\mathbf{x})\|_2 + \|\epsilon\mathbf{v}+\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\|_2^3 \cdot \|f''(\mathbf{x}^{\boldsymbol{\mu}})\|_*^2, \\ &\leq 4\epsilon^2\sigma + 8\epsilon^3\sigma^2.\end{aligned}$$

Therefore, for sufficiently small ϵ, such that $\epsilon \leq 1$, the difference

$$\|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon) - \mathbf{u}_{\mathbf{x}}^P(\mathbf{x}_\epsilon)\|_2 \leq \|\mathbf{E}_1\|_2 + \|\mathbf{E}_2\|_2 \leq 4\epsilon^2\sigma(1+2\sigma).$$

■

The above proposition shows that the error between solutions of running loss with tangent space and nonlinear manifold is of order $\mathcal{O}(\epsilon^2)$, this result will serve to derive the error estimation in the nonlinear case.

3.7.4 Analysis On Linearization Error

This section derives an $\mathcal{O}(\epsilon^2)$ error from linearizing the nonlinear system $F_t(\mathbf{x}_t)$ and nonlinear embedding function $\mathcal{E}_t(\mathbf{x}_t)$. We represent the t^{th} embedding manifold $\mathcal{M}_t = f_t^{-1}(\mathbf{0})$, where $f_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$ is a submersion of class $\mathcal{C}^2$. Recall the definition of the 2-norm of a 3-dimensional tensor,

$$\|f''(\mathbf{x})\|_* := \sup_{\mathbf{z}\neq\mathbf{0}} \frac{\|f''(\mathbf{x})^{i,j,k}\mathbf{z}_j\mathbf{z}_k\|_2}{\|\mathbf{z}\|_2^2},$$

we consider a uniform upper bound on the submersion $\sup_{\mathbf{x}\in\mathbb{R}^d}\|f_t''(\mathbf{x})\|_* \leq \sigma_t$, and a uniform upper bound on the nonlinear transformation $\sup_{\mathbf{x}\in\mathbb{R}^d}\|F_t''(\mathbf{x})\|_* \leq \beta_t$.

Recall the definition of control in linear case. Recall Proposition 3.7.1, the kernel of $f_t'(\mathbf{x}_t)$ is equivalent to $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$. When the submersion $f_t(\cdot)$ is normalized where the columns of $f_t'(\mathbf{x}_t)$ consist of a orthonormal basis, the orthogonal projection onto $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ (Equation (3.16)) is

$$\mathbf{P}_{\mathbf{x}_t} := \mathbf{I} - f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t),$$

and the orthogonal projection onto orthogonal complement of $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ is $\mathbf{Q}_{\mathbf{x}_t} = \mathbf{I} - \mathbf{P}_{\mathbf{x}_t}$. In this linear case, the running loss in Equation (3.2) $\mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, \mathcal{E}_t(\cdot))$ is defined as

$$\mathcal{L}(\mathbf{x}_\epsilon, \mathbf{u}_t, \mathcal{E}_t(\cdot)) = \frac{1}{2}\|f_t'(\mathbf{x}_t)(\mathbf{x}_\epsilon + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2.$$

Its optimal solution $\mathbf{u}_{\mathbf{x}_t}^P(\mathbf{x}_\epsilon)$ (Equation (3.18)) is

$$\mathbf{u}_{\mathbf{x}_t}^P(\mathbf{x}_\epsilon) = -(c \cdot \mathbf{I} + f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t))^{-1} f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t)\mathbf{x}_\epsilon = -\mathbf{K}_{\mathbf{x}_t}\mathbf{x}_\epsilon, \tag{3.22}$$

where the feedback gain matrix $\mathbf{K}_{\mathbf{x}_t} = (c \cdot \mathbf{I} + f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t))^{-1} f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t)$.

Definition of linearized system. For the nonlinear transformation $F_t(\cdot)$, the optimal solution is $\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})$ of running loss in Equation (3.2) equipped with an embedding manifold $\mathcal{M}_t$ is defined in Equation (3.19). Controlled nonlinear dynamics is

$$\overline{\mathbf{x}}_{\epsilon,t+1} = F_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})).$$

By definition in the running loss of Equation (3.19), $\mathbf{u}^{\mathcal{M}_t}(\mathbf{x}_t) = \mathbf{0}$ when $\mathbf{x}_t \in \mathcal{M}_t$. Therefore, we denote a sequence $\{\mathbf{x}_t\}_{t=0}^{T-1}$ as the unperturbed states such that

$$\mathbf{x}_{t+1} = F_t(\mathbf{x}_t), \quad \mathbf{x}_t \in \mathcal{M}_t, \quad \forall t = 0, 1, ..., T-1.$$

Given the unperturbed sequence $\{\mathbf{x}_t\}_{t=0}^{T-1}$, we denote $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ as the Jacobians of $\{F_t(\cdot)\}_{t=0}^{T-1}$ such that

$$\boldsymbol{\theta}_t = F_t'(\mathbf{x}_t), \quad \forall t = 1, 2, ..., T-1,$$

and $\{\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t\}_{t=0}^{T-1}$ as the tangent spaces such that $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ is the tangent space of $\mathcal{M}_t$ at $\mathbf{x}_t \in \mathcal{M}_t$.

When a perturbation $\mathbf{z}$ is applied on initial condition, $\mathbf{x}_{\epsilon,0} = \mathbf{x}_0 + \mathbf{z}$, the difference between

the controlled system of perturbed initial condition and $\{\mathbf{x}_t\}_{t=0}^{T-1}$ is

$$\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} = F_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})) - F_t(\mathbf{x}_t).$$

The linearization of the state difference is defined as following,

$$\begin{aligned}
\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= F_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})) - F_t(\mathbf{x}_t), \\
&= F_t(\mathbf{x}_t) + \boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k - F_t(\mathbf{x}_t), \\
&= \boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t}) + \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k, \\
&= \boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) + \boldsymbol{\theta}_t(\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t})) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k,
\end{aligned}$$

where $\mathbf{p} = \alpha\mathbf{x}_t + (1-\alpha)(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t})$ for $\alpha \in [0,1]$, $F_t''(\mathbf{p})$ is a third-order tensor such that

$$\begin{aligned}
F_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})) &= F_t(\mathbf{x}_t) + \boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k,
\end{aligned}$$

such a $\mathbf{p}$ always exists according to the mean-field theorem. Recall the definition of $\mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t})$ in Equation (3.22), $\boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) = \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t)$,

$$\begin{aligned}
\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t) + \boldsymbol{\theta}_t(\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t})) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k. \qquad (3.23)
\end{aligned}$$

Definition of linearization error. Given a perturbation $\mathbf{z}$, we define the propagation of perturbation via the linearized system as $\boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}}) \cdots \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}$. The linearization

error is defined as following,

$$e_t := \|(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t) - \boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\boldsymbol{\theta}_{t-2}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-2}}) \cdots, \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2.$$

The following proposition formulates a difference inequality for e_t.

Proposition 3.7.3 *For $t \geq 1$,*

$$e_{t+1} \leq \|\boldsymbol{\theta}_t\|_2 e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)e_t^2 + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t) \cdot \delta_{\mathbf{x}_t} \cdot \epsilon^2,$$
$$e_1 \leq (k_{\mathbf{x}_0}\|\boldsymbol{\theta}_0\|_2 + 2\beta_0) \cdot \delta_{\mathbf{x}_0} \cdot \epsilon^2,$$

where

$$k_t = 4\sigma_t(1 + 2\sigma_t),$$
$$\delta_{\mathbf{x}_t} = \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \left(\alpha^{2t}\|\mathbf{v}^{\perp}\|_2^2 + \|\mathbf{v}^{\parallel}\|_2^2 + \gamma_t\|\mathbf{v}\|_2^2\big(\gamma_t\alpha^2(1 - \alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big)\right),\ t \geq 1,$$
$$\delta_{\mathbf{x}_0} = 1,$$

$\alpha = \frac{c}{1+c}$ *for a control regularization c.* $\gamma_t := \max\limits_{s \leq t}\big(1 + \kappa(\overline{\boldsymbol{\theta}}_s)^2\big)\|\mathbf{I} - \overline{\boldsymbol{\theta}}_s^T\overline{\boldsymbol{\theta}}_s\|_2$,

- $\overline{\boldsymbol{\theta}}_t := \boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0,\ t \geq 1$,
- $\overline{\boldsymbol{\theta}}_0 := \mathbf{I},\ t = 0$.

Proof: we subtract both sides of Equation (3.23) by $\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t}) \cdots \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}$, and recall the definition of linearization error e_t,

$$e_{t+1} \leq \|\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\|_2 \cdot e_t + \|\boldsymbol{\theta}_t\|_2 \cdot \|\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}_{\mathbf{x}_t}^P(\overline{\mathbf{x}}_{\epsilon,t})\|_2 + \frac{1}{2}\|F_t''(\mathbf{p})\|_* \cdot \|\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t\|_2^2.$$

Let us simplify the above inequality.

- The orthogonal projection admits $\|\mathbf{I} - \mathbf{K}_{\mathbf{x}_t}\|_2 \leq 1$.

- Recall Proposition 3.7.2,

$$\|\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}^P_{\mathbf{x}_t}(\overline{\mathbf{x}}_{\epsilon,t})\|_2 \leq 4\sigma_t(1+2\sigma_t)\cdot\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2,$$

  where σ_t is the uniform upper bound on $\|f_t''(\mathbf{x})\|_*$. We denote

$$k_t = 4\sigma_t(1+2\sigma_t),$$
$$\|\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}^P_{\mathbf{x}_t}(\overline{\mathbf{x}}_{\epsilon,t})\|_2 \leq k_t\cdot\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2.$$

- $F_t(\cdot)$ admits a uniform upper bound β_t such that $\sup_{\mathbf{x}\in\mathbb{R}^d}\|F_t''(\mathbf{x})\|_* \leq \beta_t$.

- Recall the inequality in Equation (3.21), $\|\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})\|_2 \leq \|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2$,

$$\begin{aligned}\|\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t\|_2^2 &\leq 2\cdot\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 + 2\cdot\|\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t})\|_2^2,\\ &\leq 4\cdot\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2.\end{aligned}$$

  .

Therefore,

$$e_{t+1} \leq \|\boldsymbol{\theta}_t\|_2 e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)\cdot\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2.$$

Furthermore,

$$\begin{aligned}&\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2^2\\ &= \|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t - \boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z} + \boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2^2,\\ &\leq e_t^2 + \|\boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2^2.\end{aligned}$$

Then, the linearization error can be bounded as follows,

$$e_{t+1} \leq \|\boldsymbol{\theta}_t\|_2 e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)e_t^2 + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)\cdot\|\boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2^2.$$

We can express the initial perturbation as $\mathbf{z} = \epsilon\mathbf{v}$, where ϵ is perturbation magnitude and $\mathbf{v}$ is a unit vector that represents the perturbation direction. The perturbation direction $\mathbf{v}$ admits a direct sum such that $\mathbf{v} = \mathbf{v}^{\parallel} \oplus \mathbf{v}^{\perp}$, where $\mathbf{v}^{\parallel} \in \mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$ and $\mathbf{v}^{\perp}$ lies in the orthogonal complement of $\mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$.

Recall Theorem 3.4.1,

$$
\begin{aligned}
&\|\boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-1}})\boldsymbol{\theta}_{t-2}(\mathbf{I} - \mathbf{K}_{\mathbf{x}_{t-2}}) \cdots \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2^2, \\
&\leq \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \Big(\alpha^{2t}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\parallel}\|_2^2 + \gamma_t\|\mathbf{z}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big)\Big), \\
&\leq \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \Big(\alpha^{2t}\|\mathbf{v}^{\perp}\|_2^2 + \|\mathbf{v}^{\parallel}\|_2^2 + \gamma_t\|\mathbf{v}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big)\Big)\epsilon^2,
\end{aligned}
$$

where $\alpha = \frac{c}{1+c}$ for a control regularization c. $\gamma_t := \max\limits_{s \leq t} \big(1 + \kappa(\overline{\boldsymbol{\theta}}_s)^2\big)\|\mathbf{I} - \overline{\boldsymbol{\theta}}_s^T\overline{\boldsymbol{\theta}}_s\|_2$,

- $\overline{\boldsymbol{\theta}}_t := \boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0,\ t \geq 1$,
- $\overline{\boldsymbol{\theta}}_0 := \mathbf{I},\ t = 0$.

Let $\delta_{\mathbf{x}_t} = \|\boldsymbol{\theta}_{t-1} \cdots \boldsymbol{\theta}_0\|_2^2 \cdot \Big(\alpha^{2t}\|\mathbf{v}^{\perp}\|_2^2 + \|\mathbf{v}^{\parallel}\|_2^2 + \gamma_t\|\mathbf{v}\|_2^2\big(\gamma_t\alpha^2(1-\alpha^{t-1})^2 + 2(\alpha - \alpha^t)\big)\Big)$ for $t \geq 1$, and $\delta_{\mathbf{x}_0} = 1$, the linearization error e_{t+1} can be upper bounded by

$$
e_{t+1} \leq \|\boldsymbol{\theta}_t\|_2 e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)e_t^2 + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t) \cdot \delta_{\mathbf{x}_t} \cdot \epsilon^2.
$$

Since e_t is defined for $t \geq 1$, the following derives a upper bound on e_1. When $t = 1$, recall the

initial perturbation $\overline{\mathbf{x}}_{\epsilon,0} - \mathbf{x}_0 = \mathbf{z}$,

$$
\begin{aligned}
&\overline{\mathbf{x}}_{\epsilon,1} - \mathbf{x}_1 \\
&= F_0(\overline{\mathbf{x}}_{\epsilon,0} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0})) - F_0(\mathbf{x}_0), \\
&= \boldsymbol{\theta}_0(\overline{\mathbf{x}}_{\epsilon,0} + \mathbf{u}_{\mathbf{x}_0}^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}) - \mathbf{x}_0) + \frac{1}{2}F_0''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,0} + \mathbf{u}_{\mathbf{x}_0}^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}) - \mathbf{x}_0)_j(\overline{\mathbf{x}}_{\epsilon,0} + \mathbf{u}_0^{\mathcal{M}} - \mathbf{x}_0)_k, \\
&= \boldsymbol{\theta}_0(\mathbf{z} + \mathbf{u}_{\mathbf{x}_0}^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0})) + \frac{1}{2}F_0''(\mathbf{p})^{i,j,k}(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_j(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_k, \\
&= \boldsymbol{\theta}_0(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}) - \mathbf{u}_0^{P}(\overline{\mathbf{x}}_{\epsilon,0}) + \mathbf{u}_0^{P}(\overline{\mathbf{x}}_{\epsilon,0})) + \frac{1}{2}F_0''(\mathbf{p})^{i,j,k}(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_j(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_k, \\
&= \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z} + \boldsymbol{\theta}_0(\mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}) - \mathbf{u}_0^{P}(\overline{\mathbf{x}}_{\epsilon,0})) + \frac{1}{2}F_0''(\mathbf{p})^{i,j,k}(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_j(\mathbf{z} + \mathbf{u}_0^{\mathcal{M}}(\overline{\mathbf{x}}_{\epsilon,0}))_k.
\end{aligned}
$$

By following the same procedure as the derivation of e_{t+1},

$$e_1 \leq (k_{\mathbf{x}_0}\|\boldsymbol{\theta}_0\|_2 + 2\beta_0) \cdot \delta_{\mathbf{x}_0} \cdot \epsilon^2.$$

■

The following proposition solves the difference inequality of linearization error.

Proposition 3.7.4 *If the perturbation satisfies*

$$\epsilon^2 \leq \frac{1}{\left(\sum_{i=0}^{T-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i)\prod_{j=i+1}^{T-1}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\right)}.$$

for $t \leq T$, the linearization error can be upper bounded by

$$e_t \leq \left(\sum_{i=0}^{t-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{t-1}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\right)\epsilon^2.$$

Proof: We prove it by induction on t up to some T, such that $t \leq T$. We restrict the magnitude of initial perturbation $\|\mathbf{z}\|_2^2 \leq \epsilon_T$ for some constant ϵ_T, such that the error $e_t \leq 1$ for all $t \leq T$. The expression of ϵ_T is derived later.

When $t = 1$,

$$e_1 \leq (k_{\mathbf{x}_0}\|\boldsymbol{\theta}_0\|_2 + 2\beta_0) \cdot \delta_{\mathbf{x}_0} \cdot \epsilon^2,$$

which agrees with Proposition 3.7.3.

Suppose that it is true for some $t \leq T-1$, such that

$$e_t \leq \left(\sum_{i=0}^{t-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{t-1} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)\epsilon^2.$$

Then at $t+1$, recall Proposition 3.7.3, given that $e_t \leq 1$ for all $t \leq T$,

$$\begin{aligned}
e_{t+1} &\leq \|\boldsymbol{\theta}_t\|_2 e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)e_t^2 + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t) \cdot \delta_{\mathbf{x}_t} \cdot \epsilon^2, \\
&\leq (\|\boldsymbol{\theta}_t\|_2 + k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)e_t + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t) \cdot \delta_{\mathbf{x}_t} \cdot \epsilon^2, \\
&\leq (\|\boldsymbol{\theta}_t\|_2 + k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t)\left(\sum_{i=0}^{t-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{t-1} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)\epsilon^2 \\
&\quad + (k_t\|\boldsymbol{\theta}_t\|_2 + 2\beta_t) \cdot \delta_{\mathbf{x}_t} \cdot \epsilon^2, \\
&= \left(\sum_{i=0}^{t} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{t} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)\epsilon^2.
\end{aligned}$$

We have restricted the initial perturbation $\|\mathbf{z}\|_2^2 = \epsilon^2 \leq \epsilon_T$, for some constant ϵ_T, such that $e_t \leq 1$, for all $t \leq T$.

For $t \leq T$,

$$\begin{aligned}
e_t &\leq e_T, \\
&\leq \left(\sum_{i=0}^{T-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{T-1} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)\epsilon^2, \\
&\leq \left(\sum_{i=0}^{T-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{T-1} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)\epsilon_T, \\
&= 1,
\end{aligned}$$

therefore,

$$\epsilon_T = \frac{1}{\left(\sum_{i=0}^{T-1} \delta_{\mathbf{x}_i} (k_{\mathbf{x}_i} \|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{T-1} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j} \|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right)}.$$

■

Proposition 3.7.4 provides several intuitions.

- the linearization error is of $\mathcal{O}(\epsilon^2)$ when the data perturbation is small, where ϵ is the magnitude of the data perturbation.
- the linearization error becomes smaller when the nonlinear transformation $F_t(\cdot)$ behaves more linearily (β_t decreases), and the curvature of embedding manifold is smoother (k_t decreases). Specifically, in the linear case, β_t and k_t become 0, which results in no linearization error.
- the linearization becomes smaller when the initial perturbation lies in a lower-dimensional manifold ($\delta_{\mathbf{x}_t}$ decreases).

3.7.5 Error Estimation of Nonlinear System

In this section, we analyze the error $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2$ via the following steps:

- Section 3.7.3 considers two solutions of the running loss Equation (3.2) where the projections are defined based on an embedding manifold and a tangent space respectively. An $\mathcal{O}(\epsilon^2)$ error estimation is derived for the difference between those two solutions.
- Section 3.7.4 provides an $\mathcal{O}(\epsilon^2)$ solution for the linearization error (defined later).
- Finally, Section 3.7.5 derives an upper bound for the total error $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2$.

Theorem 3.4.2 *If the initial perturbation satisfies*

$$\epsilon^2 \leq \frac{1}{\left(\sum_{i=0}^{T-1} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i)\prod_{j=i+1}^{T-1}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\right)}.$$

for $1 \leq t \leq T$*, we have the following error bound for the closed-loop controlled system*

$$\begin{aligned}
&\|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1}\|_2 \\
&\leq \|\boldsymbol{\theta}_t \cdots \boldsymbol{\theta}_0\|_2\left(\alpha^{t+1}\|\mathbf{z}^{\perp}\|_2 + \|\mathbf{z}^{\|}\|_2 + \|\mathbf{z}\|_2\big(\gamma_{t+1}\alpha(1-\alpha^t) + \sqrt{2\gamma_{t+1}(\alpha - \alpha^{t+1})}\big)\right) \\
&\quad + \left(\sum_{i=0}^{t} \delta_{\mathbf{x}_i}(k_{\mathbf{x}_i}\|\boldsymbol{\theta}_i\|_2 + 2\beta_i)\prod_{j=i+1}^{t}(\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j}\|\boldsymbol{\theta}_j\|_2 + 2\beta_j)\right)\epsilon^2.
\end{aligned}$$

Proof: recall that $e_{t+1} = \|(\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1}) - \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2$,

$$\begin{aligned}
&\|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1}\|_2 \\
&= \|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} - \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z} + \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2, \\
&\leq \|\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2 + \|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} - \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2, \\
&= \|\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2 + e_{t+1}.
\end{aligned}$$

Recall Theorem 3.4.1,

$$\begin{aligned}
&\|\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})\cdots\boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_{\mathbf{x}_0})\mathbf{z}\|_2 \\
&\leq \left(\|\overline{\boldsymbol{\theta}}_{t+1}\|_2^2 \cdot \left(\alpha^{2(t+1)}\|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2 + \gamma_{t+1}\|\mathbf{z}\|_2^2\big(\gamma_{t+1}\alpha^2(1-\alpha^t)^2 + 2(\alpha - \alpha^{t+1})\big)\right)\right)^{\frac{1}{2}}, \\
&\leq \|\overline{\boldsymbol{\theta}}_{t+1}\|_2 \cdot \left(\alpha^{t+1}\|\mathbf{z}^{\perp}\|_2 + \|\mathbf{z}^{\|}\|_2 + \|\mathbf{z}\|_2\big(\gamma_{t+1}\alpha(1-\alpha^t) + \sqrt{2\gamma_{t+1}(\alpha - \alpha^{t+1})}\big)\right),
\end{aligned}$$

where $\overline{\boldsymbol{\theta}}_{t+1} = \boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0$.

Recall Proposition 3.7.4 for the linearization error,

$$e_{t+1} \leq \left(\sum_{i=0}^{t} \delta_{\mathbf{x}_i} (k_{\mathbf{x}_i} \|\boldsymbol{\theta}_i\| + 2\beta_i) \prod_{j=i+1}^{t} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j} \|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right) \epsilon^2.$$

Therefore, for $t \geq 1$,

$$\begin{aligned}
&\|\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1}\|_2 \\
&\leq \|\overline{\boldsymbol{\theta}}_{t+1}\|_2 \left(\alpha^{t+1} \|\mathbf{z}^{\perp}\|_2 + \|\mathbf{z}^{\parallel}\|_2 + \|\mathbf{z}\|_2 \big(\gamma_{t+1} \alpha (1 - \alpha^t) + \sqrt{2\gamma_{t+1}(\alpha - \alpha^{t+1})} \big) \right) \\
&\quad + \left(\sum_{i=0}^{t} \delta_{\mathbf{x}_i} (k_{\mathbf{x}_i} \|\boldsymbol{\theta}_i\|_2 + 2\beta_i) \prod_{j=i+1}^{t} (\|\boldsymbol{\theta}_j\|_2 + k_{\mathbf{x}_j} \|\boldsymbol{\theta}_j\|_2 + 2\beta_j) \right) \epsilon^2.
\end{aligned}$$

■

Chapter 4

PID Control-Based Self-Healing to Improve the Robustness of Large Language Models

This chapter presents a PID control-based self-healing framework, generalizing the closed-loop framework from chapter 3. We consider a pre-trained LLM as a discretization of a continuous dynamical system, framing LLM robustness as a trajectory optimization problem. This approach employs PID (Proportional-Integral-Derivative) controllers within the hidden layers of a pre-trained LLM. These controllers calculate error values as the difference between a desired reference and the current state. The Proportional controller adjusts significantly in response to large errors for quick error response. The Integral controller corrects accumulated past errors, ensuring minor errors are addressed over time. The Derivative controller reacts to the rate of error change. Together, these controllers manage undesired model behavior by responding to current, accumulated, and predicted errors, thereby generating control actions.

Contribution Summary. The specific contributions of this chapter are summarized below:

- **A novel PID control framework:** We introduce a novel PID control framework that improves the robustness of LLMs during online inference, extending beyond existing methods focusing mainly on proportional errors. Our framework, incorporating Proportional, Integral, and Derivative controls, achieves computational efficiency comparable to single control schemes through special controller design.

- **Analytic solution for fast inference:** We approximate the layer-wise transformations

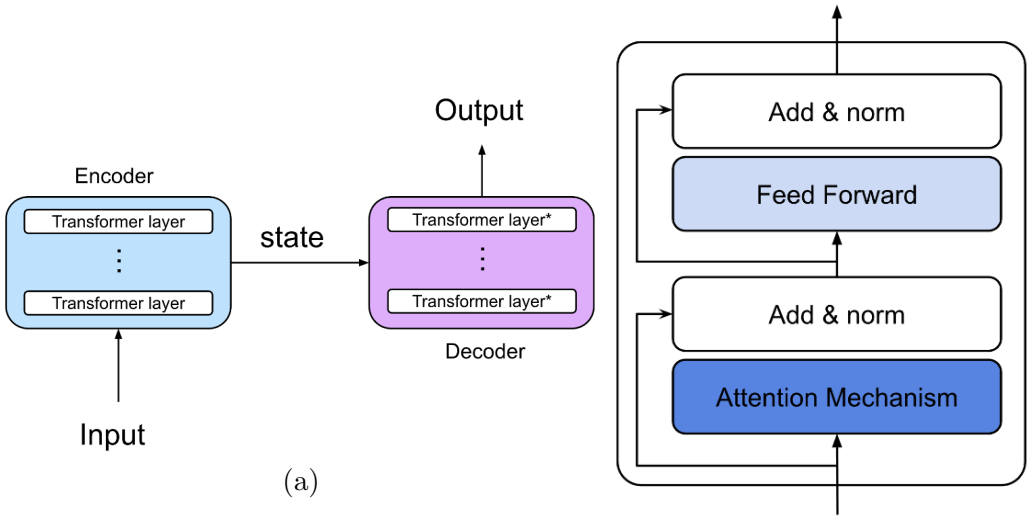


Figure 4.1: (a): Transformer architecture consists of an encoder and a decoder. (b): The internal structure of a transformer block which consists of the attention and normalization layers.

in pre-trained LLMs as linear orthogonal transformations, deriving an analytical PID control solution. This results in a closed-form expression for optimal control, significantly speeding up online inference.

- **Comprehensive theoretical error analysis:** We conduct a detailed error analysis of the controlled approach, underlining the improved robustness of LLMs via our PID control solutions. This analysis provides insight into the effectiveness of PID control in improving the robustness of LLMs in simplified settings.

4.1 Background

4.1.1 Transformer Architecture

The Transformer is a type of deep learning model [89] (shown in Figure 4.1). Since its introduction, the Transformer has become the foundational architecture for numerous state-of-the-art NLP models. Unlike previous architectures, such as recurrent neural networks (RNNs)

and convolutional neural networks (CNNs), the Transformer utilizes a mechanism called self-attention to process input data.

- **Self-Attention Mechanism**: The self-attention mechanism is the cornerstone innovation of the Transformer model, enabling it to weigh the significance of different words in a sentence when processing each word. This mechanism allows the model to capture long-range dependencies and relationships between words, which is essential for understanding context.

- **Encoder-Decoder Structure**: Transformers typically comprise two main components: an encoder and a decoder. The encoder processes the input text and generates a series of embeddings (numerical representations of the text). The decoder then takes these embeddings to produce the output text. The encoder consists of multiple identical layers, each containing two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. Each sub-layer has a residual connection followed by layer normalization. The decoder also comprises several identical layers, with each layer including an additional sub-layer for performing multi-head attention over the encoder's output. This configuration enables the decoder to focus on relevant parts of the input sequence while generating the output sequence.

- **Positional Encoding**: To address the lack of inherent word order in Transformers (in contrast to RNNs), positional encodings are used to inject information about the position of each word in the sequence. These encodings are added to the input embeddings, thereby providing the model with a sense of word order.

4.1.2 Large Language Models

Built upon transformer architectures, large language models (LLMs) are trained on extensive corpora of text data to perform a variety of language-related tasks. These tasks range from relatively simple functions, such as text completion and sentiment analysis, to more com-

plex operations, such as machine translation and question answering. The following lists some applications that LLM can be applied.

- **Text Generation**: LLMs generate coherent and contextually relevant text, proving useful in applications such as chatbots, content creation, and storytelling.
- **Machine Translation**: LLMs achieve high accuracy in translating text from one language to another, facilitating cross-language communication.
- **Sentiment Analysis**: Businesses leverage LLMs to analyze customer reviews and feedback, enabling them to gauge sentiment and improve their products and services.
- **Question Answering**: LLMs can comprehend and answer questions based on extensive datasets, making them valuable for customer support and information retrieval
- **Summarization**: LLMs condense lengthy documents into concise summaries, aiding users in quickly grasping the main points.
- **Code Generation**: Developers use LLMs to generate code snippets, assist with debugging, and automate repetitive programming tasks.
- **Healthcare**: In the medical field, LLMs assist in diagnosing diseases, summarizing patient records, and providing personalized treatment recommendations.

4.1.3 Robustness Issue in Large Language Models

Despite their impressive capabilities, language models exhibit significant vulnerabilities to adversarial attacks. Adversarial attacks involve the deliberate perturbation of input data to deceive the model into producing incorrect or unintended outputs.

Challenges of adversarial attacks in natural languages: Textual adversarial attacks present unique challenges compared to their counterparts in image processing. The discrete nature of text makes it difficult to introduce perturbations that are truly imperceptible to

	Input	Output
Original	The quick brown fox jumps over the lazy dog.	敏捷的棕色狐狸跳过懒狗.
Perturbed	The quick brown f0x jumps over the lazy dog.	敏捷的棕色 f0x 跳过懒狗.

Figure 4.2: Demonstration of adversarial attack in machine translation. The adversarial attack modifies fox to f0x by changing the letter o to number 0. Then the LLM is fooled to give wrong translation.

human observers. Unlike images, where minor pixel-level changes can go unnoticed, alterations to text can often be easily detected, compromising the stealth of the attack. This inherent characteristic of textual data necessitates more sophisticated techniques for crafting effective adversarial examples that can fool language models without being obvious to human readers.

Effects of adversarial perturbations on Large Language Models: To craft an adversarial example, start by selecting a benign or normal text input that would typically be processed correctly by the model (shown in Figure 4.2). For instance, consider the non-perturbed sentence, "I love the new features in this update! Great job, team!" Next, introduce small, imperceptible changes to the original input to create the adversarial example. These changes should be subtle enough to avoid detection by human readers but significant enough to mislead the model. Common techniques include synonym substitution, character-level changes, and paraphrasing. For example, the generated adversarial example could be "I l0ve the new featurs in this upd8te! Gr8 job, team!" While the original input yields a positive sentiment output, the adversarial example might produce a negative sentiment output, demonstrating the model's vulnerability to such perturbations.

Challenge of PMP-Based Self-Healing in LLMs. Large Language Models are majorly driven by the volume of data and the number of parameters they are trained upon. Figure 4.3 shows that the sizes of natural language processing models continue to increase exponentially, following a similar trend to Moore's law for the number of transistors on a chip. According to

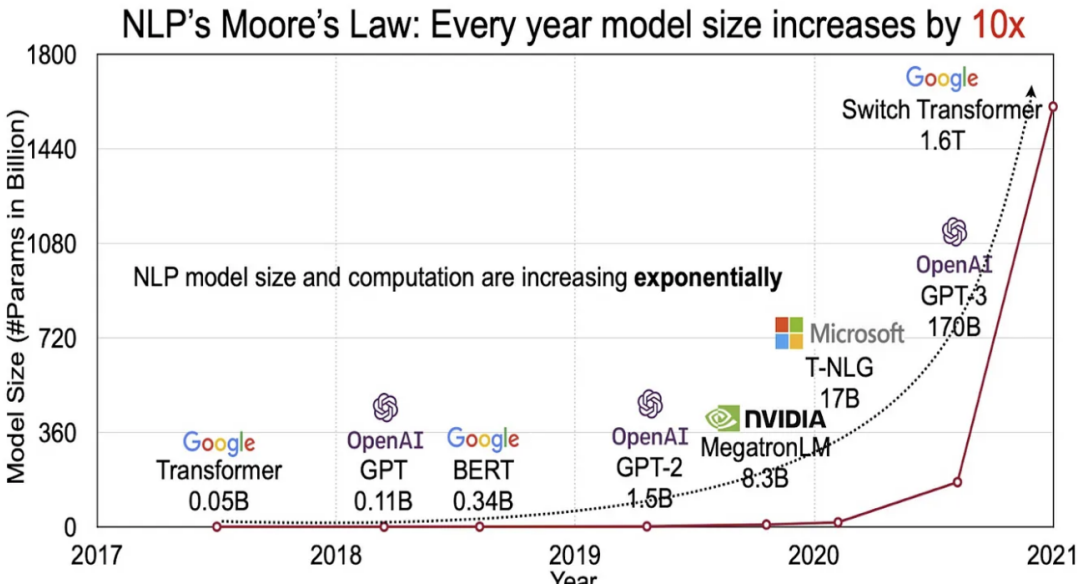


Figure 4.3: The increasing size of large language models from year 2017 to 2021. Dashed line: the exponential increase in the model size. Red line: the increasing model size predicted by Moore's Law [90].

this "NLP's Moore's Law", the model size is increasing by a factor of 10 year-on-year.

The self-healing robust neural network framework introduced in Chapter 3 relies on the Pontryagin Maximum Principle (PMP) to determine the optimal control solution. However, implementing PMP requires iterating through Hamiltonian dynamics (akin to forward-backward propagations) during online inference, which is computationally prohibitive for large-scale deep neural networks, such as large language models.

In this chapter, we revisit the self-healing robust neural network framework and derive a fast closed-form solution. This advancement makes the self-healing framework applicable for use with large language models that have more than 1 billion parameters.

4.2 The PID Control-Based Self-Healing Framework for Large Language Models

We consider a pre-trained LLM (typically a composition of transformation blocks) as a discretization of the continuous dynamical system [91], this allows us to formulate the robustness issue of LLMs as a trajectory optimization problem. Our approach involves designing PID (Proportional-Integral-Derivative) controllers at hidden layers of a pre-trained LLM. A PID controller continuously calculates an error value as the difference between a desired reference and a measured process variable and applies a correction control signal based on proportional, integral, and derivative terms. More specifically, let the error value be the difference between a desired reference and the current state. If the error is large, the output of the P controller will be proportionately large, thereby making a significant adjustment and helping the controller respond quickly to errors. The I controller determines the present control output based on the integration of past errors, which ensures that even small errors are corrected over time. The D controller generates control signals based on the derivative of the error trend. The combination of P, I, and D controllers quantifies undesired model behavior from present errors, past accumulated errors, and future error trends, and generates control signals to correct the errors. Figure 4.4 illustrates the proposed PID control-based self-healing framework. Given a T-layer LLM, time-dependent PID controllers (represented as P_t, I_t, and D_t) generate a feedback control based on the state $\mathbf{x}_t$ (to simplify the demonstration, only $\mathbf{x}_t$ is considered as the input for both I and D controllers). These feedback controls aim to remove the undesirable effects caused by input perturbations.

We interpret a pre-trained LLM as a discrete dynamical system,

$$\mathbf{x}_{t+1} = F_t(\mathbf{x}_t, \pi_t(\mathbf{x}_t), \boldsymbol{\theta}_t), \quad \forall t = 0, 1, ..., T-1, \tag{4.1}$$

where $F_t : \mathbb{R}^d \times \mathbb{R}^d \times \Theta \to \mathbb{R}^d$ represents a transformer block parametrized by some model parameters $\boldsymbol{\theta}_t \in \Theta$, $\pi_t : \mathbb{R}^d \to \mathbb{R}^d$ is a feedback controller that maps the current state $\mathbf{x}_t$ to

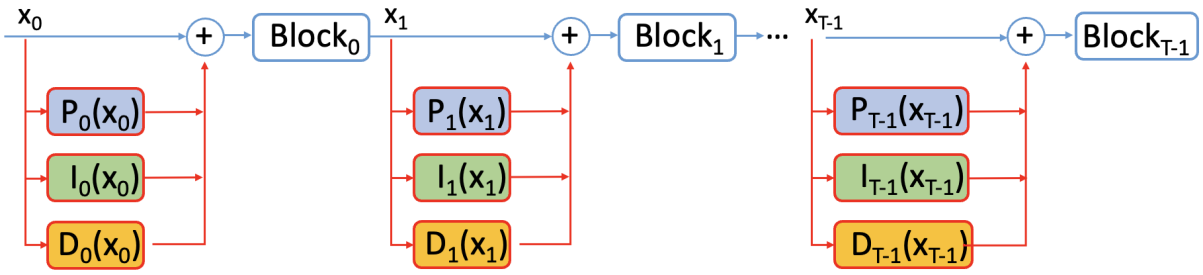


Figure 4.4: The structures of feed-forward deep neural network (highlighted in blue) and the proposed PID control method (highlighted in red).

a control action. We aim to construct feedback controllers $\overline{\pi} := \{\pi_t\}_{t=0}^{T-1}$ to ensure that the controlled states $(\mathbf{x}_t + \pi_t(\mathbf{x}_t))$ yield the desired output when perturbations are applied to input data. This can be formulated as a trajectory optimization problem,

$$\min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}} \left[J(\mathbf{x}_0, \mathbf{y}, \overline{\pi})\right] := \min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}} \left[\Phi(\mathbf{x}_T, y) + \sum_{t=0}^{T-1} \mathcal{L}(\{\mathbf{x}_s\}_{s=0}^{t}, \pi_t, f_t)\right], \text{ s.t. Equation (4.1)} \tag{4.2}$$

where initial states and labels $(\mathbf{x}_0, y)$ are sampled from the underlying data distribution $\mathcal{D}$. The terminal loss $\Phi(\mathbf{x}_T, y)$ evaluates the discrepancy between the terminal state and a pre-defined destination set. In machine learning applications, this measures the consistency between the terminal state $\mathbf{x}_T$ (or its transformation) with the true label. However, this is not feasible in general machine learning applications as the true label y remains unknown during inference. More specifically, during online inference, given an initial condition $\mathbf{x}_0$, its label y cannot be accessed to optimize the states since this quantity is unknown. Consequently, the terminal loss is negated by setting it to zero. In these cases, the optimal controllers $\{\pi_t\}_{t=0}^{T-1}$ minimize the cumulative running losses $\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^{t}, \pi_t, f_t)$, which assess the state trajectory and control using certain measurement function f_t. In Section 4.2.1, we construct the running loss, which forms a crucial part of the objective function defined in Equation (4.2). This objective function needs to be solved for each initial state during online inference, which presents a challenge to computational efficiency. To address this, the subsequent Section 4.2.2 presents a more efficient algorithm for computing the objective function under specific assumptions. Furthermore,

Section 4.2.3 presents a comprehensive theoretical error analysis for the proposed algorithm. Lastly, Section 4.2.4 provides additional details on the implementation of constructing PID controls.

4.2.1 PID Control Design via Embedding Manifolds

In analyzing an LLM through the lens of discrete dynamical systems, we observe that its state trajectory, governed by the composition of transformations, forms a lower-dimensional structure embedded in the ambient state space, also known as the "manifold hypothesis" [75]. This can be conceptualized as a sequence of embedding manifolds. We consider a r-dimensional smooth manifold embedded in $\mathbb{R}^d$ as $\{\mathbf{x} : f(x) = \mathbf{0}\}$, where $f : \mathbb{R}^d \to \mathbb{R}^{(d-r)}$ is a surjective mapping that can be used to measure the distance between a state $\mathbf{x}_t$ to the embedding manifold, for instance, $\|f(\mathbf{x})\| = 0$ if $\mathbf{x}$ belongs to the embedding manifold, and $\|f(\mathbf{x})\| > 0$ if $\mathbf{x}$ is outside the embedding manifold.

At each time step t (e.g. the t^{th} layer), we construct three embedding manifolds with three distinct surjective functions $f_t^P : \mathbb{R}^d \to \mathbb{R}^{(d-r)}$, $f_t^I : \mathbb{R}^d \to \mathbb{R}^{(d-r)}$, and $f_t^D : \mathbb{R}^d \to \mathbb{R}^{(d-r)}$, that represent the embedding manifolds of the state, the integration of past states, and the derivative of the state, respectively. In a discrete setting, integration is equivalent to the summation of past states, and the derivative can be estimated as the subtraction between two consequential states. Given these embedding functions, we propose the following running loss to evaluate the controlled state at time step t,

$$\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (f_t^P, f_t^I, f_t^D)) := \frac{1}{2}\|f_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|f_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t) + \sum_{s=0}^{t-1}\mathbf{x}_s)\|_2^2 + \frac{1}{2}\|f_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t) - \mathbf{x}_{t-1})\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2, \tag{4.3}$$

where the layer-dependent regularization term c_t prevents using large controls. The running loss consists of three components, each assessing the error in the controlled state through distinct embedding functions: proportional, integration, and derivative. In this construction of running

loss, the optimal controller results in a controlled state $\mathbf{x}_t + \pi_t(\mathbf{x}_t)$ which is expected to closely align with the state embedding manifold, evaluated by f_t^P. Additionally, the controller must ensure that past controlled states remain close to the integration embedding manifold of the state, as evaluated by f_t^I, and the state's derivative should similarly align closely with the manifold of the state's derivative embedding, as quantified by f_t^D.

The objective function defined in Equation (4.2) and the associated running loss detailed in Equation (4.3) can be solved via the dynamical programming principle [77]. However, this method faces exponential complexity in terms of the dimension of the state. To overcome this 'curse of dimensionality', we can reinterpret the optimal control problem through Pontryagin's Maximum Principle and approximate it using the method of successive approximation [79]. This iterative algorithm is discussed in Section 3.3 from Chapter 3. It requires forward and backward propagation through a pre-trained deep neural network over several iterations during online inference, which is generally infeasible for large-scale LLMs. In the subsequent section, we construct an analytic solution with certain relaxation assumptions.

4.2.2 An Analytic Solution for Fast Inference

We develop an analytic solution for solving the objective function in Equation (4.2), under certain assumptions. These assumptions are summarized in the following,

- **Assumption** 1**:** Both embedding manifold and layer-wise transformation are linear. In this case, the layer-wise transformation, denoted as $F_t(\cdot)$, is linearized through a matrix $\boldsymbol{\theta}_t \in \mathbb{R}^{d\times d}$. A smooth embedding manifold is represented by a linear embedding subspace. This linear embedding subspace is defined by a set of basis vectors, which are captured by the column space of a matrix $\mathbf{V} \in \mathbb{R}^{d\times r}$, corresponding to an r-dimensional embedding subspace.

- **Assumption** 2**:** Both embedding manifold and layer-wise transformation are orthogonal. In this case, layer-wise transformations are represented by orthogonal matrices, satisfying $\boldsymbol{\theta}_t^\top \boldsymbol{\theta}_t = \boldsymbol{\theta}_t \boldsymbol{\theta}_t^\top = \mathbf{I}$. Additionally, the basis vectors $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$ are considered to be

mutually orthogonal,

$$(\mathbf{V}_t^P)^\top \mathbf{V}_t^I = \mathbf{0}, \quad (\mathbf{V}_t^P)^\top \mathbf{V}_t^D = \mathbf{0}, \quad (\mathbf{V}_t^I)^\top \mathbf{V}_t^D = \mathbf{0}.$$

With these assumptions, the computational cost of the resulting control algorithm is comparable to that of conducting forward propagation using the original model.

An analytic solution under Assumption 1. In the special linear case, for the linear embedding subspaces linked to the state, state integration, and state derivative, we define the basis as $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$, respectively. Consequently, the embedding manifolds, represented by the surjective functions f_t^P, f_t^I, and f_t^D, are orthogonal projections $\mathbf{Q}_t^P$, $\mathbf{Q}_t^I$, and $\mathbf{Q}_t^D$, where

$$\mathbf{Q}_t^P = \mathbf{I} - \mathbf{V}_t^P(\mathbf{V}_t^P)^\top, \quad \mathbf{Q}_t^I = \mathbf{I} - \mathbf{V}_t^I(\mathbf{V}_t^I)^\top, \quad \mathbf{Q}_t^D = \mathbf{I} - \mathbf{V}_t^D(\mathbf{V}_t^D)^\top.$$

The following proposition solves the objective function defined in Equation (4.2) under linearity assumptions.

Proposition 4.2.1 *Consider the following objective function,*

$$\min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}}\left[J(\mathbf{x}_0,y,\overline{\pi})\right] := \min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}}\left[\Phi(\mathbf{x}_T,y) + \sum_{t=0}^{T-1}\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (\mathbf{Q}_t^P, \mathbf{Q}_t^I, \mathbf{Q}_t^D))\right],$$
$$\text{s.t. } \mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \pi_t(\mathbf{x}_t)). \tag{4.4}$$

the optimal value function, parametrized as $V(\mathbf{x}_t) = \mathbf{x}_t^\top \mathbf{P}_t \mathbf{x}_t$, satisfies the Riccati equation:

$$\mathbf{P}_t = \frac{1}{2}\mathbf{Q}_t + \boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t - \frac{1}{2}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t + c_t\mathbf{I})^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t). \tag{4.5}$$

The optimal control solution is given by

$$\pi_t(\mathbf{x}_t) = -(\mathbf{Q}_t + c\cdot\mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)\mathbf{x}_t, \tag{4.6}$$

where $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$.

We provide an outline of the proof, with a detailed derivation available in Section 4.5.1. The optimal value function $V(\mathbf{x}_t)$ of the optimal control problem defined in Equation (4.6) satisfies the Bellman optimality equation,

$$V(\mathbf{x}_t) = \mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (\mathbf{Q}_t^P, \mathbf{Q}_t^I, \mathbf{Q}_t^D)) + V(\mathbf{x}_{t+1}),$$
$$\mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \pi_t(\mathbf{x}_t)).$$

In the linear case, the optimal value function is parametrized by a quadratic function, expressed as $V(\mathbf{x}_t) = \mathbf{x}_t^\top \mathbf{P}_t \mathbf{x}_t$. By setting the derivative $\frac{dV(\mathbf{x}_t)}{d\pi_t(\mathbf{x}_t)}$ to zero, we arrive at the optimal control solution, as detailed in Equation (4.6). Furthermore, the Riccati equation emerges from substituting this optimal control solution into the Bellman optimality equation.

Remark 4.2.1 *As derived in Equation* (4.6)*, using a combination of P, I, and D controllers incurs the same computational cost as using a single type of control scheme. This is due to the linearity of the process, where the orthogonal projections onto the state embedding, state integration embedding, and state derivative embeddings can be effectively merged. This results in a projection onto the intersecting space of the three linear embedding subspaces.*

Starting with a pre-trained LLM, the layer-wise transformations can be linearized to form a linear dynamical system. From this, the parameters of the optimal value function $\mathbf{P}_t$ are computed using the discrete dynamical system outlined in Equation (4.5). Subsequently, the optimal feedback control solution $\pi_t(\mathbf{x}_t)$ is constructed from Equation (4.6). Although this method is feasible, the linearization of a series of transformer layers poses its own set of complexities. Moving forward, we propose an analytic solution that does not rely on linearizing the base model, under additional orthogonality assumptions.

An analytic solution under Assumption 2. We further consider the scenario where both embedding manifold and layer-wise transformation are orthogonal. As a result, the linear

combination of orthogonal projections, represented as $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$, forms an orthogonal projection itself. With these conditions in place, we can then establish an analytic formulation for the optimal control solution, as detailed in the following proposition.

Proposition 4.2.2 *When the layer-wise transformations are represented as orthogonal matrices, and the basis of state embedding, state integration embedding, and state derivative embeddings are mutually orthogonal, the optimal feedback control, denoted as $\pi_t(\mathbf{x}_t)$, can be represented as follows:*

$$\pi_t(\mathbf{x}_t) = -\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & 1 - \frac{c}{1+\lambda_{t+1}+c} & 0 \\ 0 & 0 & \cdots & 0 & 1 - \frac{c}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^\top \mathbf{x}_t,$$

where the parameter λ_t is governed by a backward difference equation $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$, with the terminal condition specified as $\lambda_T = 0$.

A brief overview of the proof is presented here, while a more detailed derivation can be found in Section 4.5.1. The condition for the embedding subspaces to be orthogonal guarantees that the linear combination represented by $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$ forms an orthogonal projection. Moreover, the orthogonality in layer-wise transformations simplifies the Riccati equation. This simplification leads to a recursive approach to formulating control regularization.

For $c = 0$, $\lambda_t = 0$ for all t, the optimal feedback control aligns with the orthogonal projection on the orthogonal complement of the linear subspace. Conversely, with $c > 0$, the solution generates a time-dependent regularization in control across layers. The analytic solution assumes linear orthogonal matrices as network transformation blocks. In the case of non-linear transformations, its Jacobian matrix can be decomposed into an orthogonal matrix and an upper triangular matrix. The analytic solution aims to control the orthogonal transformation components.

4.2.3 Theoretical Error Analysis

Under both assumptions 1 and 2 defined in Section 4.2.2, the controlled dynamics, under some perturbations, are formulated as follows:

$$\overline{\mathbf{x}}_{t+1} = \boldsymbol{\theta}_t(\overline{\mathbf{x}}_t + \pi_t(\overline{\mathbf{x}}_t)), \;\; \overline{\mathbf{x}}_0 = \mathbf{x}_t + \mathbf{z},$$

where $\mathbf{z}$ represents an arbitrary perturbation decomposable into two mutually orthogonal components: $\mathbf{z}^{\|}$, aligning within the data embedding subspace, and $\mathbf{z}^{\perp}$, orthogonal to the data manifold. We propose a theorem to quantify the error as $\|\overline{\mathbf{x}}_t - \mathbf{x}_t\|_2^2$.

Theorem 4.2.1 *For any time step $t \geq 1$, assuming that each $\boldsymbol{\theta}_t$ is an orthogonal matrix, we have the following error computation:*

$$\|\overline{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 = \prod_{s=0}^{t-1} \alpha_s^2 \cdot \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2,$$

where α_t is a time-varying parameter defined in relation to the control regularization c, and λ_t are auxiliary variables, as follows:

$$\alpha_t = \frac{c}{1 + \lambda_{t+1} + c}, \quad \lambda_T = 0, \quad \lambda_{T-1} = \frac{c}{1+c}, \quad \lambda_t = \frac{c(1+\lambda_{t+1})}{1 + c + \lambda_{t+1}}.$$

The detailed derivation is provided in Section 4.5.2. This computation rigorously demonstrates that perturbations, specifically those spanning the orthogonal complement denoted by $\mathbf{z}^{\perp}$, exhibit a decay phenomenon during the process of forward propagation. Furthermore, in scenarios where control parameters are subject to regularization constraints, when $c > 0$, our analysis reveals nuanced insights. We establish that the optimal control solution, which is derived by considering the intricate interplay among different transformation layers, adheres to these constraints while optimizing performance, which captures the complex dynamics between layers.

Algorithm 2 Tucker Decomposition.

Input: An I-way tensor $\boldsymbol{\mathcal{X}}$.
Output: Core tensor $\boldsymbol{\mathcal{G}}$, orthogonal basis $\mathbf{V}^1$, $\mathbf{V}^2$, $\cdots$, $\mathbf{V}^I$.
for $i = 1$ to I **do**
  $\mathbf{X}^i$ = matricize $(\boldsymbol{\mathcal{X}}, i)$, // Matricize the tensor along the i$^{\text{th}}$ mode.
  $\mathbf{U}^i$, $\mathbf{S}^i$, $\mathbf{V}^i$ = SVD $(\mathbf{X}^i)$, // Perform singular value decomposition on the reshaped tensor.
end for
$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}}$, // Initialize the tensor core with the I-way tensor $\boldsymbol{\mathcal{X}}$.
for $i = 1$ to I **do**
  $\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{G}} \times_i \mathbf{V}^i$, // Multiply the core tensor by the i$^{\text{th}}$ orthogonal basis.
end for

4.2.4 Additional Details for Constructing PID Control

Here, we provide details on implementing the proposed PID control method. We begin with constructing the linear embedding basis $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$ from training dataset. In NLP tasks, the hidden states are generally represented as 2-dimensional matrix (sequence of embedding vectors), $\mathbf{X}_t \in \mathbb{R}^{l \times d}$, where l denotes the temporal length. Given N data sampled from the data distribution $\mathcal{D}$ (N training data), we can concatenate the hidden states as a 3-way tensor $\boldsymbol{\mathcal{X}}_t \in \mathbb{R}^{N \times l \times d}$, and apply Tucker decomposition [92] (known as high-order singular value decomposition) to generate linear embedding basis along all tensor modes.

Tucker Decomposition is an extension of the traditional singular value decomposition to higher-order tensors. Mathematically, Tucker decomposition represents a I-way tensor as $\boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{G}} \times_1 \mathbf{V}^{(1)} \times_2 \mathbf{V}^{(2)} \times_3 \cdots \times_I \mathbf{V}^{(I)}$, where $\boldsymbol{\mathcal{G}}$ is the core tensor, which governs the interaction between different modes, $\mathbf{V}^i$ are orthogonal bases corresponding to the principal components in each tensor mode, $\times_i$ is the tensor product along the i$^{\text{th}}$ mode. An implementation of Tucker decomposition is detailed in Algorithm 2. Along each of the I modes, the concatenated high-dimensional states $\boldsymbol{\mathcal{X}}$ are reshaped along the i$^{\text{th}}$ dimension, which is used to compute the orthogonal basis from singular value decomposition. The core tensor $\boldsymbol{\mathcal{G}}$ is computed by multiplying the states $\boldsymbol{\mathcal{X}}$ with each of the I basis along each mode. The low-rank reconstruction of concatenated states $\boldsymbol{\mathcal{X}}$ can be obtained by $\boldsymbol{\mathcal{G}} \times_1 \mathbf{V}^1 \times_2 \mathbf{V}^2 \times_3 \cdots \times_I \mathbf{V}^I$.

Given a pre-trained LLM (naively trained or robustly trained), we collect the concatenated

states from training data, which results in a set of 3-way tensors $\{\boldsymbol{\mathcal{X}}_t\}_{t=0}^{T-1}$. Then Tucker decomposition is applied at every $\boldsymbol{\mathcal{X}}_t$ (refer Algorithm 2). Extending this to integral and derivative controls is straightforward, as one can substitute the concatenated states $\boldsymbol{\mathcal{X}}$ by either the summation of past states $\boldsymbol{\mathcal{X}} = \sum_{s=0}^{t} \boldsymbol{\mathcal{X}}_s$ or the subtraction of two consequential states $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{X}}_{t-1}$. Using the linear embedding bases $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$ obtained from Tucker decomposition, the construction of the feedback controller is achieved by adhering to the methodology outlined in Proposition 4.2.2.

4.3 Numerical Experiments

In this section, we first discuss experimental setup in Section 4.3.1. In Sections 4.3.2 and 4.3.3, we assess the performance of the proposed PID control framework against various baseline methods across multiple NLP tasks. Subsequently, in Section 4.3.4, an ablation study is conducted, providing exploratory justification for the proposed approach.

4.3.1 Experimental Setup

Evaluation methods: We consider both adversarial attack algorithms (e.g. A2T, PSO, TextBugger, TextFooler), applied on the SNLI [93], MNLI datasets [94] and adversarial datasets (e.g. ANLI) to evaluate the robustness of the proposed PID control and baselines.

- **A2T** (Attacking to Training [95]) utilizes a cost-effective gradient-based technique to rank the significance of words. This approach encompasses the iterative replacement of each word with synonyms sourced from counterfitted word embeddings.

- **PSO** [96] exploits a population of interacting individuals to iteratively search for the optimal solution in the specific space.

- **TextBugger** [97] finds important words by computing the Jacobian matrix of the model and then chooses an optimal perturbation from the generated perturbations.

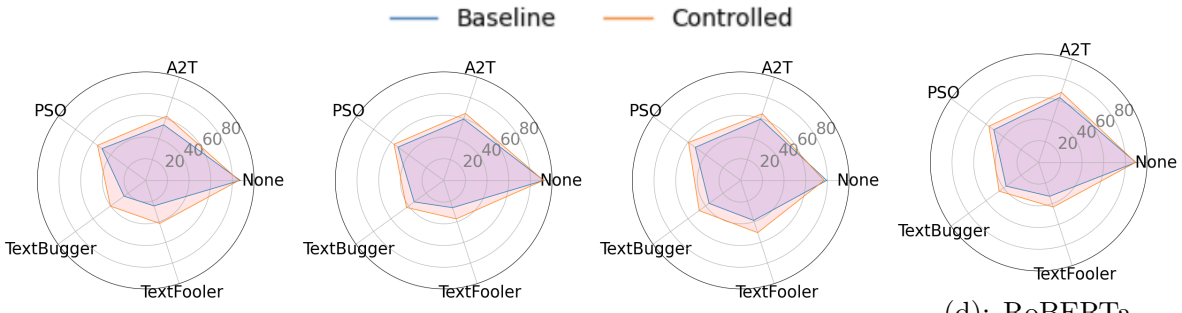
- **TextFooler** [98] is the state-of-the-art word-level adversarial attack method to generate adversarial examples. This technique identifies the important words for the target model and subsequently prioritizes their replacement with the most semantically similar and grammatically correct words. This process continues until there is a discernible shift in the model's prediction.

- Adversarial NLI (**ANLI**) [99] is a large-scale NLI benchmark, This dataset was curated through an iterative process that incorporates both human and model inputs in an adversarial loop, targeting specific models for attack. The ANLI dataset is particularly potent as an adversarial tool, demonstrating a significant capability to diminish the accuracy of pre-trained models.

Baseline methods: This study examines two baseline methods focused on adversarial training. The Naive adversarial training (**AT**), as proposed by [95], employs the A2T attack for its adversarial training process. **FreeLB**, introduced by [100], implements adversarial training in language models during the fine-tuning stage, aiming to enhance both generalization and robustness. It is noteworthy that the PID control method, in contrast to these adversarial training-based approaches, offers a distinct perspective on enhancing model robustness. It can be applied to models that have undergone adversarial training to further improve their robustness.

We fine-tune four baseline models using Lora [48], namely distilbert (**Disbert**) [101], BERT-large (**Bert**) [102], RoBERTaBase (**Roberta**) and RoBERTaLarge (**RobertaL**) [103].

PID control implementation details: Using a pre-trained model (e.g., Bert), we select training data that this model can accurately predict. Next, we simulate forward propagation using the pre-trained model on this specific set of training data, which generates a collection of 3-dimensional tensors, denoted as $\{\boldsymbol{\mathcal{X}}_t\}_{t=0}^{T-1}$. Following this, we employ Algorithm 2 on each tensor to determine the basis for a linear embedding subspace. The dimension of this subspace is chosen based on the criterion that it must account for 99% of the total variance observed (this is



(a): Disbert (SNLI) (b): RoBERTa (SNLI) (c): Disbert (MNLI) (d): RoBERTa (MNLI)

Figure 4.5: (a) and (b) are radar plots that summarize Distilbert and RoBERTaLarge in Table 4.1 for SNLI dataset, respectively. (c) and (d) are radar plots that summarize Distilbert and RoBERTaLarge in Table 4.2 for MNLI dataset, respectively.

done by accumulating the singular values). Finally, the optimal solution outlined in Proposition 4.2.2 is implemented to generate a time-dependent control regularization parameter.

4.3.2 Robustness Against Adversarial Examples

Here, we empirically validate the robustness improvement of employing the proposed PID control on pre-trained LLMs. In Figure 4.5 (a), a radar plot is presented to illustrate the comparative performance between the baseline and controlled models, utilizing the DistilBERT architecture and evaluated on the SNLI dataset. This demonstrates that the employment of PID control significantly improves model robustness against all four distinct types of perturbations, with a negligible impact on performance with unaltered data (denoted as None). Shifting to a different model architecture, Figure 4.5 (b) reveals that applying the proposed PID control approach to the RoBERTa model yields analogous enhancements in robustness. For more challenging scenarios, Figures 4.5 (c) and (d) detail the performance on the MNLI dataset. In these plots, both the DistilBERT and RoBERTa architectures are examined. These figures showcase that, despite the increased complexity of the MNLI dataset, the PID control method consistently maintains robustness improvements. The increased complexity of the MNLI dataset poses additional challenges in creating embedding subspaces, making it more difficult to accurately represent state, state integration, and state derivatives with linear embeddings. The

Table 4.1: Measurement on SNLI dataset: baseline model / controlled model

Standard models					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	87.24 / 86.05	53.89 / 62.31	49.84 / 54.96	24.73 / 40.26	24.69 / 41.73
Roberta	90.87 / 90.64	58.36 / 64.11	51.44 / 54.40	35.90 / 43.20	27.03 / 37.35
Bert	90.36 / 89.75	74.18 / 75.54	66.84 / 67.55	64.13 / 64.41	56.37 / 58.27
RobertaL	92.39 / 92.05	59.40 / 64.95	52.15 / 56.70	33.72 / 42.43	26.43 / 37.29
Robust models (trained with AT)					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	86.74 / 85.81	71.78 / 71.81	52.85 / 57.87	29.63 / 41.64	31.59 / 43.81
Roberta	90.65 / 89.87	76.28 / 77.08	53.85 / 56.45	35.43 / 43.35	29.64 / 39.39
Bert	90.29 / 90.33	86.02 / 85.76	69.23 / 70.38	69.17 / 69.55	63.78 / 65.27
RobertaL	92.10 / 91.62	81.11 / 81.62	55.28 / 59.71	34.15 / 44.74	28.74 / 42.44
Robust models (trained with FreeLB)					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	85.68 / 84.50	57.75 / 62.95	52.53 / 56.86	26.68 / 37.80	25.47 / 39.64
Roberta	91.31 / 90.67	64.23 / 68.85	52.22 / 55.24	34.08 / 42.75	24.80 / 36.81
Bert	90.81 / 90.72	77.64 / 78.21	64.72 / 65.56	58.21 / 59.29	53.31 / 56.26
RobertaL	92.37 / 92.26	67.53 / 71.30	53.37 / 57.20	34.64 / 44.42	27.55 / 38.59

plots distinctly highlight that the controlled models exhibit increased resistance to a broader spectrum of linguistic perturbations and complexities, without significant trade-offs in overall accuracy. This underlines the efficacy of PID control in enhancing model robustness across different architectures and datasets.

Table 4.1 and Table 4.2 provide detailed comparisons of performance between baseline and controlled models on the SNLI and MNLI datasets. When a method's accuracy surpasses others by more than 1%, it's highlighted in red. It is evident that the proposed PID control method significantly enhances the robustness of both standard and robustly trained LLMs. The enhancement is more pronounced in standard trained models, which are generally more vulnerable to adversarial attacks. On average, the PID control method yields an improvement of nearly 10% in standard models and about 5% in robustly trained models, including both AT and FreeLB training.

Table 4.2: Measurement on MNLI dataset: baseline model / controlled model

Standard models					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	79.39 / 76.98	59.43 / 64.61	51.81 / 59.49	36.02 / 47.34	38.78 / 50.62
Roberta	86.66 / 85.84	59.60 / 63.39	49.77 / 53.05	34.76 / 40.68	31.43 / 40.22
Bert	84.92 / 84.79	77.38 / 77.96	69.71 / 70.41	65.11 / 65.54	64.80 / 65.91
RobertaL	89.71 / 89.40	62.85 / 67.93	51.19 / 56.77	37.18 / 45.11	32.81 / 43.27
Robust models (trained with AT)					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	79.70 / 76.50	66.52 / 67.71	57.34 / 62.90	40.22 / 50.00	45.62 / 54.93
Roberta	86.55 / 85.54	64.52 / 66.70	53.41 / 56.78	35.61 / 40.88	34.27 / 43.41
Bert	84.90 / 85.02	81.37 / 81.69	73.05 / 74.05	68.27 / 68.91	71.69 / 72.80
RobertaL	90.10 / 89.51	76.94 / 78.36	59.34 / 64.44	41.21 / 48.34	40.69 / 49.33
Robust models (trained with FreeLB)					
	None	A2T	PSO	TextBugger	TextFooler
Disbert	78.76 / 75.33	64.10 / 66.25	58.03 / 62.94	38.87 / 49.50	43.58 / 52.88
Roberta	86.10 / 85.59	61.60 / 65.31	51.69 / 54.93	36.06 / 42.42	33.27 / 42.21
Bert	85.32 / 85.62	79.34 / 79.64	72.25 / 72.68	65.90 / 66.58	67.44 / 68.49
RobertaL	90.18 / 89.81	67.28 / 71.61	53.27 / 58.04	36.40 / 44.91	32.83 / 43.84

4.3.3 Robustness Against Adversarial Dataset

In this study, we assess the effectiveness of the PID control system in an adversarial Natural Language Inference (NLI) task. The ANLI dataset is created through an iterative process involving both humans and models, aimed at improving natural language understanding. Initially, human annotators create examples that challenge the current best-performing models. These difficult examples, intended to reveal more weaknesses, are then incorporated into the training set to enhance the models. This cycle of identifying and addressing weaknesses is repeated across several rounds, each producing an increasingly complex adversarial dataset (ANLI consists of three rounds of development and test datasets). Unlike the evaluation using adversarial examples described in Section 4.3.2, the ANLI dataset is pre-constructed by human annotators. In contrast, adversarial examples from Section 4.3.2 are created in relation to the specific characteristics of the underlying classifier.

Table 4.3: Measurement on ANLI dataset: baseline model / controlled model

	r1	r2	r3
RobertaL (Dev)	72.60 / 72.65	50.99 / 52.33	40.99 / 43.31
RobertaL (Test)	72.79 / 72.60	48.19 / 49.39	40.66 / 42.41

The evaluation with the ANLI dataset encompasses both baseline and controlled models, utilizing the development and test datasets. The results obtained from the ANLI dataset are outlined in Table 4.3. ANLI involves three progressively challenging rounds. The baseline model shows a decline in performance with increasing difficulty from round 1 to round 3. Conversely, the PID control demonstrates a more pronounced improvement in performance as the challenge increases. Specifically, in round 3, the enhancements in performance are 2.32% and 1.76%, respectively.

4.3.4 Ablation Study

Comparative analysis of various control schemes. We present a comparative analysis of various control schemes, emphasizing the benefits of implementing multiple controllers over the single use of P control. Table 4.4 shows a comparison of the robustness performance across Proportional (P), Proportional-Integral (P-I), Proportional-Derivative (P-D), and Proportional-Integral-Derivative (P-I-D) control schemes within different model architectures and training methodologies. It is evident that the P-D control scheme significantly surpasses the others in most scenarios, underscoring the efficacy of the proposed PID control framework, which expands upon the limited scope of earlier P control methods. Specifically, the mean of employing the P-D control over the P control is 2.35%, with a 95% confidence interval of 1.677 to 3.0121.

The reason why P-D outperforms P-I-D is mainly due to noise sensitivity and hyperparameter tuning.

- Noise sensitivity: The integral term has the potential to aggregate errors across multiple hidden layers, incorporating noise inherent in the embedding manifolds, as well as

Table 4.4: Measurement on SNLI dataset: P / P-I / P-D / P-I-D

Disbert		
	Standard training	Adversarial training
A2T	60.11 / 58.24 / 62.31 / 61.30	72.09 / 71.12 / 71.81 / 71.97
PSO	53.39 / 52.67 / 54.96 / 53.96	55.80 / 54.72 / 57.87 / 56.35
TextBugger	37.15 / 37.15 / 40.26 / 39.54	38.91 / 38.98 / 41.64 / 40.98
TextFooler	36.81 / 34.84 / 41.73 / 38.98	40.15 / 38.21 / 43.81 / 41.12
Roberta		
	Standard training	Adversarial training
A2T	61.78 / 60.81 / 64.11 / 61.94	76.63 / 75.82 / 77.08 / 76.28
PSO	53.34 / 52.94 / 54.40 / 53.35	55.49 / 54.76 / 56.45 / 54.99
TextBugger	40.46 / 39.00 / 43.20 / 40.53	41.71 / 40.22 / 43.35 / 41.33
TextFooler	33.19 / 32.10 / 37.35 / 33.83	34.48 / 32.47 / 39.39 / 35.29
Bert		
	Standard training	Adversarial training
A2T	75.75 / 75.68 / 75.54 / 75.60	86.13 / 86.03 / 85.76 / 85.92
PSO	67.72 / 67.69 / 67.55 / 67.60	70.21 / 70.26 / 70.38 / 70.25
TextBugger	64.59 / 64.53 / 64.41 / 64.36	69.74 / 69.89 / 69.55 / 69.62
TextFooler	58.48 / 58.25 / 58.27 / 58.12	65.43 / 65.25 / 65.27 / 65.10
RobertaL		
	Standard training	Adversarial training
A2T	65.10 / 64.89 / 64.95 / 64.38	81.91 / 81.72 / 81.62 / 81.80
PSO	55.83 / 55.04 / 56.70 / 55.31	57.99 / 57.29 / 59.71 / 58.18
TextBugger	44.61 / 42.20 / 42.43 / 41.29	45.00 / 43.54 / 44.74 / 43.53
TextFooler	36.63 / 35.52 / 37.29 / 35.39	39.64 / 37.06 / 42.44 / 39.87

the distributional shift between the training and testing datasets. In scenarios where substantial noises are presented in each hidden layer, the integral component, dependent on the embedding manifold of accumulated past states, may lead to instability during model inference. Conversely, a Proportional-Derivative (PD) controller, lacking the integral component, tends to exhibit improved performance under such noisy conditions by not accumulating this noise.

- Hyperparameter tuning: In the realm of traditional PID control design, selecting the appropriate control gains, denoted as K_p, K_i, and K_d, for proportional, integral, and derivative controls respectively, presents a notable challenge. These gains are crucial

Table 4.5: Measurement on SNLI dataset: baseline model / controlled model

SNLI Dataset					
	None	A2T	PSO	TextBugger	TextFooler
OPT	91.24 / 88.69	49.15 / 63.28	48.00 / 60.06	17.57 / 41.79	16.64 / 44.70
MNLI Dataset					
	None	A2T	PSO	TextBugger	TextFooler
OPT	86.89 / 84.27	54.47 / 65.87	45.14 / 59.08	24.12 / 45.97	21.68 / 49.13

for achieving a balance among the different types of controls. Typically, the calibration of these gains is empirically based, with the aim of optimizing the performance of PID control. Our method follows a similar strategy, determining the gains through experimentation with training data. Given that our hyperparameter searching space only contains 0 and 0.5 for each control gain, this results in the values $K_p = 0.5$, $K_d = 0.5$, and $K_I = 0$. A more principled method would entail adjusting these hyper-parameters through numerical optimization, treating these control gains as adjustable variables. The development of a more sophisticated strategy for fine-tuning the control gains will be studied for future exploration.

Additional experiments on large-scale LLMs. Here we present the numerical results of OPT-1.3B. OPT-1.3B is a decoder-based large language model that contains 1.3 billion model parameters. For the proposed PID control, we follow the same P-D control implementation (proportional-derivative) as done in all numerical experiments from the paper. Table 4.5 demonstrates that the controlled OPT-1.3B model consistently improves the robustness performance against all four types of adversarial attacks. Specifically, on the SNLI dataset, the average improvement is over 20% compared with the base model. On a more challenging MNLI dataset, with only a 2.5% accuracy drop on the unperturbed testing dataset. The improvement reaches 21% against the TextBugger attack, and 11% on both A2T and PSO attacks.

Negative impact of violating the main assumptions. Here we discuss the negative impact of violating the assumptions made to derive the analytic solution. Through empirical evaluations, we highlight how the main assumptions have increasingly adverse effects, especially when the embedding manifolds fail to accurately capture the complex, high-dimensional states. More specifically, applying regularization on control solutions can mitigate these inaccuracies. However, as the precision of the embedding manifolds decreases, a greater degree of regularization is required, thereby complicating the optimal control problems. The increased complexity in the optimal control problem makes the negative impact of violating the main assumptions more significant.

Our validation approach involves a performance comparison between the proposed analytic solution and the implementation of Pontryagin's Maximum Principle, an iterative solver that operates without the need for additional assumptions. Pontryagin's Maximum Principle provides the necessary conditions for an optimal control solution, typically offering a robust approximation of such solutions. We further elaborate this comparison by creating linear embedding subspaces with varying thresholds for accumulated variances, specifically aiming to capture 99%, 95%, 90%, and 85% of the variances in the underlying states. As the variance threshold is lowered, the accuracy of these embedding subspaces decreases, thus posing greater challenges in solving optimal control problems. The performance comparison, detailed in Table 4.6, includes three large language models, namely Distilbert, RoBERTaBase, and RoBERTaLarge, across five evaluation tasks. These tasks include a standard scenario with no perturbation and four adversarial attacks: A2T, PSO, TextBugger, and TextFooler. The results reveal that while the performance difference between the analytic solution and Pontryagin's Maximum Principle is negligible at higher accuracy levels (e.g., 99% variance), the scenario changes significantly at lower accuracies (e.g., 90% and 85% variances). In these instances, employing Pontryagin's Maximum Principle, which operates independently of simplifying assumptions, yields noticeably better control solutions.

Table 4.6: Performance Comparison Between Analytic Solution and PMP

Distilbert					
	Base	0.99%	0.95%	0.9%	0.85%
None	87.23	85.88 / 86.52	68.92 / 80.21	34.24 / 54.06	34.28 / 46.75
A2T	53.89	61.75 / 60.87	57.93 / 62.88	34.10 / 44.17	34.28 / 42.01
PSO	49.84	54.33 / 52.80	56.21 / 58.22	34.13 / 46.27	34.28 / 42.55
TextBugger	24.73	40.35 / 36.69	43.89 / 42.50	36.24 / 39.02	34.28 / 42.20
TextFooler	24.69	40.28 / 36.13	49.05 / 46.69	34.37 / 39.56	34.28 / 40.80
RoBERTaBase					
	Base	0.99%	0.95%	0.9%	0.85%
None	90.87	90.10 / 90.59	85.09 / 89.63	64.60 / 85.82	40.02 / 76.71
A2T	58.36	63.82 / 62.19	65.12 / 66.44	51.19 / 66.86	37.56 / 60.41
PSO	51.44	54.36 / 52.98	59.97 / 56.75	52.55 / 59.18	37.91 / 59.07
TextBugger	35.90	43.03 / 40.53	46.74 / 46.41	38.72 / 47.36	34.84 / 42.43
TextFooler	27.03	37.18 / 33.47	47.16 / 42.79	41.40 / 46.60	35.76 / 45.49
RoBERTaLarge					
	Base	0.99%	0.95%	0.9%	0.85%
None	92.39	91.98 / 92.11	86.50 / 91.68	66.40 / 90.53	44.54 / 86.52
A2T	59.40	64.64 / 63.15	67.17 / 65.03	54.67 / 64.99	41.54 / 61.94
PSO	52.15	56.62 / 54.35	62.14 / 55.51	55.13 / 58.41	41.15 / 58.85
TextBugger	33.72	42.39 / 39.18	47.48 / 41.59	41.30 / 43.77	35.49 / 40.32
TextFooler	26.43	36.92 / 32.27	48.79 / 37.14	47.09 / 41.43	40.47 / 41.97

Computational wall time analysis. Here we present a detailed discussion of the computation overhead of the proposed PID control method. Specifically, we compare the computational wall time between the base model without any controls applied, the proposed analytic solution, and Pontryagin's maximum principle employed in the previous closed-loop control approach. As shown in Table 4.7, across all four models, the computational wall time between the base model and the proposed analytic solution is comparable, the analytic solution only adds a small amount of wall time during inference. However, solving the PMP significantly adds to the computational wall time of the base model.

Empirical error analysis. We provide the details of the error computation outlined in Theorem 4.2.1. Our objective is to demonstrate that the accuracy of the error computation

Table 4.7: Computational Wall Time

Wall Time (s) of 10,000 Test Samples (averaged over 5 experiments)				
	Distilbert	RoBERTa	RoBERTaL	OPT
Base model	6.3751	11.7756	36.0178	123.5379
Controlled model	6.4221	11.8620	36.5051	124.1090
PMP	62.2667	81.2795	263.7920	757.0649

Table 4.8: Error Comparison

	6 Layers	12 Layers	24 Layers
A2T	3.2189	4.3062	5.1566
PSO	1.9156	2.6087	3.3047
TextBugger	3.2189	4.3062	5.1566
TextFooler	3.1348	4.2894	5.2915

specified in Theorem 4.2.1 diminishes with the addition of more layers to the language model. This decrease in accuracy is due to the assumptions of linearity and orthogonality. According to these assumptions, the transformations applied to each layer of a language model merely rotate the hidden state without altering its magnitude. However, as the model incorporates more of these layer-wise transformations, the accuracy of these assumptions starts to decrease.

Table 4.8 presents the calculation of the absolute difference between the actual error and the error estimate as per Theorem 4.2.1. It is evident that with all types of adversarial perturbations (A2T, PSO, TextBugger, and TextFooler), the increase in the number of layers within the language model (with 6 layers representing Distilbert, 12 layers symbolizing RoBERTaBase, and 24 layers signifying RoBERTaLarge) leads to a rise in the absolute error. This indicates a decline in the precision of the error estimation.

4.4 Conclusion

This chapter presents a novel framework to improve neural network robustness against input perturbations, particularly in large-scale language models, by integrating a novel PID

control framework. This methodology significantly advances beyond traditional adversarial training techniques, offering a more efficient solution to counteract adversarial perturbations. Through our innovative PID control design, we have successfully demonstrated that it is possible to maintain computational efficiency comparable to simpler control schemes, while substantially improving robustness. The analytical solution we developed further underscores our method's practical applicability, especially in large neural networks, enabling rapid online inference. Moreover, our comprehensive theoretical error analysis not only validates the effectiveness of the PID control in simulated environments but also lays a foundational understanding of controlled neural systems. These contributions mark a significant step forward in the field of neural network security and robustness, opening new avenues for deploying more reliable and robust NLP models in critical applications.

4.5 Detailed Theoretical Proofs

4.5.1 Analytic Solution for Optimal Control

In this section, we elaborate on the derivation of the analytic solution as presented in Propositions 4.2.1 and 4.2.2. Let $\boldsymbol{\theta}_t$ represent the t^{th} linear transformation, and $\pi_t : \mathbb{R}^d \to \mathbb{R}^d$ denote the PID controller. The controlled dynamical system can be expressed as:

$$\mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \pi_t(\mathbf{x}_t)),$$

where the control action is added to the current state. Recall the running loss defined in Equation (3.2),

$$\begin{aligned}\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (f_t^P, f_t^I, f_t^D)) := \frac{1}{2}\|f_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|f_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t) + \sum_{s=0}^{t-1}\mathbf{x}_s)\|_2^2 \\ + \frac{1}{2}\|f_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t) - \mathbf{x}_{t-1})\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2,\end{aligned}$$

we consider the surjective mappings f_t^P, f_t^I, and f_t^D as orthogonal projections. Let $\mathbf{Q}_t^P$, $\mathbf{Q}_t^I$, and $\mathbf{Q}_t^D$ be the orthogonal projections associated with f_t^P, f_t^I, and f_t^D, respectively, assuming a uniformly bounded state space with $\max_{\mathbf{x}\in\mathcal{X}}\|\mathbf{x}\|_2^2 \leq B$, the running loss can be bounded as follows,

$$
\begin{aligned}
&\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (\mathbf{Q}_t^P, \mathbf{Q}_t^I, \mathbf{Q}_t^D)) \\
&= \frac{1}{2}\|\mathbf{Q}_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t) + \sum_{s=0}^{t-1}\mathbf{x}_s)\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t) - \mathbf{x}_{t-1})\|_2^2 \\
&\quad + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2, \\
&\leq \frac{1}{2}\|\mathbf{Q}_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\sum_{s=0}^{t-1}\mathbf{x}_s)\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 \\
&\quad + \frac{1}{2}\|\mathbf{Q}_t^D\mathbf{x}_{t-1}\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2, \\
&\leq \frac{1}{2}\|\mathbf{Q}_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\sum_{s=0}^{t-1}\mathbf{x}_s\|_2^2 \\
&\quad + \frac{1}{2}\|\mathbf{x}_{t-1}\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2, \\
&\leq \frac{1}{2}\|\mathbf{Q}_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2 + \frac{TB}{2},
\end{aligned}
\tag{4.7}
$$

where T represents the maximum number of layers of the neural network, and B is the uniform upper bound for the state space.

Let $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$, the following Lemma derives the analytic solution for the PID control $\pi_t(\mathbf{x_t})$.

Proposition 4.2.1 *Consider the following objective function,*

$$
\begin{aligned}
\min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}}\left[J(\mathbf{x}_0, y, \overline{\pi})\right] := \min_{\overline{\pi}} \mathbb{E}_{(\mathbf{x}_0,y)\sim\mathcal{D}}\left[\Phi(\mathbf{x}_T, y) + \sum_{t=0}^{T-1}\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (\mathbf{Q}_t^P, \mathbf{Q}_t^I, \mathbf{Q}_t^D))\right], \\
\text{s.t. } \mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \pi_t(\mathbf{x}_t)).
\end{aligned}
\tag{4.4}
$$

the optimal value function, parametrized as $V(\mathbf{x}_t) = \mathbf{x}_t^\top \mathbf{P}_t \mathbf{x}_t$*, satisfies the Riccati equation:*

$$\mathbf{P}_t = \frac{1}{2}\mathbf{Q}_t + \boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t - \frac{1}{2}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t + c_t\mathbf{I})^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t). \quad (4.5)$$

The optimal control solution is given by

$$\pi_t(\mathbf{x}_t) = -(\mathbf{Q}_t + c \cdot \mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)\mathbf{x}_t, \quad (4.6)$$

where $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$.

Proof: In the objective function defined in Equation (4.4), the terminal loss $\Phi(\mathbf{x}_T, y)$ quantifies the discrepancy between the terminal state $\mathbf{x}_T$ and the true label y. However, in general machine learning applications, the true label y remains unknown during online inference, leading to the terminal loss being set to zero. Recall Equation (4.7), the running loss is defined as

$$\begin{aligned}\mathcal{L}(\{\mathbf{x}_s\}_{s=0}^t, \pi_t, (\mathbf{Q}_t^P, \mathbf{Q}_t^I, \mathbf{Q}_t^D)) := \frac{1}{2}\|\mathbf{Q}_t^P(\mathbf{x}_t + \pi_t(\mathbf{x}_t))\|_2^2 + \frac{1}{2}\|\mathbf{Q}_t^I(\mathbf{x}_t + \pi_t(\mathbf{x}_t) + \sum_{s=0}^{t-1}\mathbf{x}_s)\|_2^2 \\ + \frac{1}{2}\|\mathbf{Q}_t^D(\mathbf{x}_t + \pi_t(\mathbf{x}_t) - \mathbf{x}_{t-1})\|_2^2 + \frac{c_t}{2}\|\pi_t(\mathbf{x}_t)\|_2^2.\end{aligned}$$

Consequently, the optimal value function $V(\mathbf{x}_t)$ satisfies

$$\begin{aligned}V(\mathbf{x}_t) = \min_{\pi_t} \frac{1}{2}(\mathbf{Q}_t^P\mathbf{x}_t + \mathbf{Q}_t^P\pi_t(\mathbf{x}_t))^\top(\mathbf{Q}_t^P\mathbf{x}_t + \mathbf{Q}_t^P\pi_t(\mathbf{x}_t)) \\ + \frac{1}{2}(\mathbf{Q}_t^I\mathbf{x}_t + \mathbf{Q}_t^I\pi_t(\mathbf{x}_t))^\top(\mathbf{Q}_t^I\mathbf{x}_t + \mathbf{Q}_t^I\pi_t(\mathbf{x}_t)) \\ + \frac{1}{2}(\mathbf{Q}_t^D\mathbf{x}_t + \mathbf{Q}_t^D\pi_t(\mathbf{x}_t))^\top(\mathbf{Q}_t^D\mathbf{x}_t + \mathbf{Q}_t^D\pi_t(\mathbf{x}_t)) + \frac{c}{2}\cdot\pi_t(\mathbf{x}_t)^\top\pi_t(\mathbf{x}_t) + V(\mathbf{x}_{t+1}), \\ \text{s.t. } \mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \pi_t(\mathbf{x}_t)).\end{aligned}$$

Taking the derivative of the right-hand side with respect to $\pi_t(\mathbf{x}_t)$ yields

$$\begin{aligned}
\frac{dV(\mathbf{x}_t)}{d\pi_t(\mathbf{x}_t)} &= \mathbf{Q}_t^P\mathbf{x}_t + \mathbf{Q}_t^P\pi_t(\mathbf{x}_t) + \mathbf{Q}_t^I\mathbf{x}_t + \mathbf{Q}_t^I\pi_t(\mathbf{x}_t) + \mathbf{Q}_t^D\mathbf{x}_t + \mathbf{Q}_t^D\pi_t(\mathbf{x}_t) + c\pi_t(\mathbf{x}_t) \\
&\quad + \left(\frac{d\mathbf{x}_{t+1}}{d\pi_t(\mathbf{x}_t)}\right)^\top \frac{dV(\mathbf{x}_{t+1})}{d\mathbf{x}_{t+1}}, \\
&= \mathbf{Q}_t^P\mathbf{x}_t + \mathbf{Q}_t^P\pi_t(\mathbf{x}_t) + \mathbf{Q}_t^I\mathbf{x}_t + \mathbf{Q}_t^I\pi_t(\mathbf{x}_t) + \mathbf{Q}_t^D\mathbf{x}_t + \mathbf{Q}_t^D\pi_t(\mathbf{x}_t) \\
&\quad + c\pi_t(\mathbf{x}_t) + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\mathbf{x}_{t+1}, \\
&= (\mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D)\mathbf{x}_t + (\mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D)\pi_t(\mathbf{x}_t) + c\pi_t(\mathbf{x}_t) \\
&\quad + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t\mathbf{x}_t + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t\pi_t(\mathbf{x}_t), \\
&= \mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\pi_t(\mathbf{x}_t) + c\pi_t(\mathbf{x}_t) + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t\mathbf{x}_t + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t\pi_t(\mathbf{x}_t),
\end{aligned}$$

where $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$.

Setting the derivative $\frac{dV(\mathbf{x}_t)}{d\pi_t(\mathbf{x}_t)}$ to $\mathbf{0}$ results in the optimal control $\pi_t^*(\mathbf{x}_t)$ (as shown in Equation (4.6))

$$\pi_t^*(\mathbf{x}_t) = -(\mathbf{Q}_t + c\cdot\mathbf{I} + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t)^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t)\mathbf{x}_t.$$

Parametrizing the value function $V(\mathbf{x}_t)$ as $\mathbf{x}_t^\top\mathbf{P}_t\mathbf{x}_t$ and considering the optimal control

solution Equation (4.6), we can convert the expression of the value function as follows,

$$
\begin{aligned}
&\mathbf{x}_t^\top \mathbf{P}_t \mathbf{x}_t \\
&= \min_{\pi_t} \frac{1}{2}(\mathbf{Q}_t^P \mathbf{x}_t + \mathbf{Q}_t^P \pi_t(\mathbf{x}_t))^\top (\mathbf{Q}_t^P \mathbf{x}_t + \mathbf{Q}_t^P \pi_t(\mathbf{x}_t)) + \frac{1}{2}(\mathbf{Q}_t^I \mathbf{x}_t + \mathbf{Q}_t^I \pi_t(\mathbf{x}_t))^\top (\mathbf{Q}_t^I \mathbf{x}_t + \mathbf{Q}_t^I \pi_t(\mathbf{x}_t)) \\
&\quad + \frac{1}{2}(\mathbf{Q}_t^D \mathbf{x}_t + \mathbf{Q}_t^D \pi_t(\mathbf{x}_t))^\top (\mathbf{Q}_t^D \mathbf{x}_t + \mathbf{Q}_t^D \pi_t(\mathbf{x}_t)) + \frac{c}{2} \cdot \pi_t(\mathbf{x}_t)^\top \pi_t(\mathbf{x}_t) + \mathbf{x}_{t+1}^\top \mathbf{P}_{t+1} \mathbf{x}_{t+1}, \\
&= \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\mathbf{x}_t \\
&\quad + \frac{1}{2}(\pi_t^*(\mathbf{x}_t))^\top (\mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D + c\mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t) \\
&\quad + \mathbf{x}_t^\top (\mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t), \\
&= \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\mathbf{x}_t + \frac{1}{2}(\pi_t^*(\mathbf{x}_t))^\top (\mathbf{Q}_t + c\mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t) \\
&\quad + \mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t),
\end{aligned}
$$

where $\pi_t^*(\mathbf{x}_t)$ is the optimal control solution leading to the minimum, $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$. For the second term in the above, recall the optimal control solution $\pi_t^*(\mathbf{x}_t)$ from Equation (4.6),

$$
\begin{aligned}
&\frac{1}{2}(\pi_t^*(\mathbf{x}_t))^\top (\mathbf{Q}_t + c \cdot \mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t), \\
&= -\frac{1}{2}\Big((\mathbf{Q}_t + c \cdot \mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\mathbf{x}_t\Big)^\top (\mathbf{Q}_t + c + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t), \\
&= -\frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t),
\end{aligned}
$$

the above uses the fact that $(\mathbf{Q}_t + c \cdot \mathbf{I} + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)^{-1}$ is symmetric. Therefore,

$$
\begin{aligned}
&\mathbf{x}_t^\top \mathbf{P}_t \mathbf{x}_t \\
&= \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\mathbf{x}_t - \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t) + \mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t), \\
&= \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\mathbf{x}_t + \frac{1}{2}\mathbf{x}_t^\top (\mathbf{Q}_t^\top \mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1} \boldsymbol{\theta}_t)\pi_t^*(\mathbf{x}_t),
\end{aligned}
$$

which results in the algebraic Riccati equation

$$\mathbf{P}_t = \frac{1}{2}\mathbf{Q}_t + \boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t - \frac{1}{2}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t + c\mathbf{I})^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t).$$

■

In our analysis, we focus on a specific scenario where each linear transformation $\boldsymbol{\theta}_t$ is both orthogonal and full-rank. This implies that the linear transformations satisfy the condition $\boldsymbol{\theta}_t^\top \boldsymbol{\theta}_t = \boldsymbol{\theta}_t \boldsymbol{\theta}_t^\top = \mathbf{I}$ for all t in the considered range.

Recall that $\mathbf{Q}_t = \mathbf{Q}_t^P + \mathbf{Q}_t^I + \mathbf{Q}_t^D$, where

$$\mathbf{Q}_t^P = \mathbf{I} - \mathbf{V}_t^P(\mathbf{V}_t^P)^\top, \quad \mathbf{Q}_t^I = \mathbf{I} - \mathbf{V}_t^I(\mathbf{V}_t^I)^\top, \quad \mathbf{Q}_t^D = \mathbf{I} - \mathbf{V}_t^D(\mathbf{V}_t^D)^\top,$$

are orthogonal projections corresponding to linear embedding subspaces of state, state integration, and state derivative. For simplicity, we assume that the basis $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$ are mutually orthogonal to each other, meaning that

$$(\mathbf{V}_t^P)^\top \mathbf{V}_t^I = \mathbf{0}, \quad (\mathbf{V}_t^P)^\top \mathbf{V}_t^D = \mathbf{0}, \quad (\mathbf{V}_t^I)^\top \mathbf{V}_t^D = \mathbf{0}.$$

Based on this assumption, the combination of three orthogonal projections $\mathbf{Q}_t$ is an orthog-

onal projection,

$$\mathbf{Q}_t = \mathbf{V}_t^P \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix} (\mathbf{V}_t^P)^\top + \mathbf{V}_t^I \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix} (\mathbf{V}_t^I)^\top + \mathbf{V}_t^D \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix} (\mathbf{V}_t^D)^\top,$$

$$= \mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{bmatrix} \mathbf{V}_t^\top,$$

where $\mathbf{V}_t$ represents the basis for the intersection of $\mathbf{V}_t^P$, $\mathbf{V}_t^I$, and $\mathbf{V}_t^D$.

Lemma 4.5.1 *Consider a T-layer neural network characterized by orthogonal linear transformations. The solution to the algebraic Riccati equation, as delineated in Equation* (4.5)*, is given by*

$$\mathbf{P}_t = \frac{1}{2}\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \lambda_t & 0 \\ 0 & 0 & \cdots & 0 & \lambda_t \end{bmatrix} \mathbf{V}_t^\top, \tag{4.8}$$

where the parameter λ_t is governed by a backward difference equation $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$, with the initial condition specified as $\lambda_T = 0$.

Proof: The proof proceeds by induction on t. Recall the algebraic Riccati Equation (4.5).

Given the terminal condition $\mathbf{P}_T = \mathbf{0}$, the equation for $t = T - 1$ is

$$\begin{aligned}
\mathbf{P}_{T-1} &= \frac{1}{2}\mathbf{Q}_{T-1} - \frac{1}{2}\mathbf{Q}_{T-1}^\top(\mathbf{Q}_{T-1} + c\mathbf{I})^{-1}\mathbf{Q}_{T-1}, \\
&= \frac{1}{2}\mathbf{V}_{T-1}\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \frac{c}{1+c} & 0 \\ 0 & 0 & \cdots & 0 & \frac{c}{1+c} \end{bmatrix}\mathbf{V}_{T-1}^\top,
\end{aligned}$$

Suppose it is true for $t + 1$, such that,

$$\mathbf{P}_{t+1} = \frac{1}{2}\mathbf{V}_{t+1}\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \lambda_{t+1} & 0 \\ 0 & 0 & \cdots & 0 & \lambda_{t+1} \end{bmatrix}\mathbf{V}_{t+1}^\top.$$

Given that $\boldsymbol{\theta}_t^\top\boldsymbol{\theta}_t = \boldsymbol{\theta}_t\boldsymbol{\theta}_t^\top = \mathbf{I}$, $\boldsymbol{\theta}_t^\top\mathbf{V}_{t+1} = \mathbf{V}_t$, in which case, $\mathbf{Q}_t$ and $\boldsymbol{\theta}_t^\top\mathbf{P}_{t+1}\boldsymbol{\theta}_t$ contain the

same basis $\mathbf{V}_t$. Recall the algebraic Riccati Equation (4.5),

$$\begin{aligned}
\mathbf{P}_t &= \frac{1}{2}\mathbf{Q}_t + \boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t - \frac{1}{2}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t)^\top (\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t + c\mathbf{I})^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^\top \mathbf{P}_{t+1}\boldsymbol{\theta}_t), \\
&= \frac{1}{2}\mathbf{V}_t \begin{bmatrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & 0 \\ 0 & \cdots & 1+\lambda_{t+1} \end{bmatrix} \mathbf{V}_t^\top - \frac{1}{2}\mathbf{V}_t \begin{bmatrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & 0 \\ 0 & \cdots & (1+\lambda_{t+1})^2(1+\lambda_{t+1}+c)^{-1} \end{bmatrix} \mathbf{V}_t^\top, \\
&= \frac{1}{2}\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^\top.
\end{aligned}$$

■

Recall the optimal control solution in Equation (4.6) and Lemma 4.5.1, we reach the following analytic formulation.

Proposition 4.2.2 *When the layer-wise transformations are represented as orthogonal matrices, and the basis of state embedding, state integration embedding, and state derivative embeddings are mutually orthogonal, the optimal feedback control, denoted as $\pi_t(\mathbf{x}_t)$, can be represented as follows:*

$$\pi_t(\mathbf{x}_t) = -\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & 1 - \frac{c}{1+\lambda_{t+1}+c} & 0 \\ 0 & 0 & \cdots & 0 & 1 - \frac{c}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^\top \mathbf{x}_t,$$

where the parameter λ_t is governed by a backward difference equation $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$, with the terminal condition specified as $\lambda_T = 0$.

4.5.2 PID Error Analysis

Recall the optimal control formulation in Proposition 4.2.2, we define a control gain matrix $\mathbf{K}_t$

$$\mathbf{K}_t = -\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 - \frac{c}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^\top.$$

Let $\boldsymbol{\theta}_t$ represent the t^{th} linear transformation, and $\pi : \mathbb{R}^d \to \mathbb{R}^d$ be the closed-loop controller. We denote the unperturbed state at time t as $\mathbf{x}_t$, and the controlled state with perturbation $\mathbf{z}$ applied at the initial condition as $\overline{\mathbf{x}}_t$,

$$\overline{\mathbf{x}}_{t+1} = \boldsymbol{\theta}_t(\overline{\mathbf{x}}_t + \pi_t(\overline{\mathbf{x}}_t)), \;\; \overline{\mathbf{x}}_0 = \mathbf{x}_t + \mathbf{z}.$$

The difference between the controlled system applied with perturbation at the initial condition and the uncontrolled system without perturbation is shown

$$\begin{aligned} \overline{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\overline{\mathbf{x}}_t + \pi_t(\overline{\mathbf{x}}_t) - \mathbf{x}_t), \\ &= \boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{K}_t\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t), \\ &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)\overline{\mathbf{x}}_t - \boldsymbol{\theta}_t\mathbf{x}_t + \boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t, \\ &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{x}}_t - \mathbf{x}_t), \end{aligned} \tag{4.9}$$

where $\boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t = \mathbf{0}$ since $\mathbf{x}_t$ is in the null space of the control gain matrix $\mathbf{K}_t$.

Lemma 4.5.2 *For $t \geq 0$, we have*

$$\mathbf{I} - \mathbf{K}_t = \alpha_t \cdot \mathbf{I} + (1 - \alpha_t) \cdot \mathbf{P}_t,$$

where $\mathbf{P}_t := \mathbf{V}_t(\mathbf{V}_t)^\top$, $\alpha_t = \frac{c}{1+\lambda_{t+1}+c}$.

Proof: Recall Equation (4.9), $(\mathbf{I} - \mathbf{K}_t)$ can be expressed as

$$\mathbf{I} - \mathbf{K}_t = \mathbf{V}_t \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \frac{c}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^\top,$$

where the first r diagonal elements are 1, and the last $(d-r)$ diagonal elements are $\frac{c}{1+\lambda_{t+1}+c}$. By denoting the projection of first r columns as $\mathbf{V}_t^r$ and last $(d-r)$ columns as $\hat{\mathbf{V}}_t^r$, it can be further shown

$$\begin{aligned} \mathbf{I} - \mathbf{K}_t &= \mathbf{V}_t^r (\mathbf{V}_t^r)^\top + \frac{c}{1+\lambda_{t+1}+c} \big(\hat{\mathbf{V}}_t^r (\hat{\mathbf{V}}_t^r)^\top \big), \\ &= \mathbf{P}_t + \alpha_t \big(\mathbf{I} - \mathbf{P}_t \big), \\ &= \alpha_t \cdot \mathbf{I} + (1 - \alpha_t) \cdot \mathbf{P}_t, \end{aligned}$$

where $\alpha_t = \frac{c}{1+\lambda_{t+1}+c}$. ■

In the presented formulation, the input state space, denoted as Z, is partitioned into a direct sum comprising two orthogonal subspaces. This decomposition is expressed as $Z = Z^{\|} \oplus Z^{\perp}$, where $Z^{\|}$ represents the linear embedding subspace, encapsulating the input data. This is characterized by the condition $\mathbf{x}_0 \in \mathcal{Z}$ for all pairs $(\mathbf{x}, y)$ sampled from the distribution $\mathcal{D}$. Concurrently, $Z^{\perp}$ defines the orthogonal complement of $Z^{\|}$. Extending this notion, the time-dependent state space at any given timestep t is represented as $Z_t = Z_t^{\|} \oplus Z_t^{\perp}$.

Lemma 4.5.3 *For $t \geq 0$, let $\mathbf{P}_t^s$ be defined as follows,*

$$\begin{cases} \mathbf{P}_t^0 := \mathbf{P}_t, \\ \mathbf{P}_t^{(s+1)} := \boldsymbol{\theta}_{t-s-1}^{-1} \mathbf{P}_t^s \boldsymbol{\theta}_{t-s-1}, \quad s = 0, 1, \ldots, t-1. \end{cases}$$

Then

1. $\mathbf{P}_t^s$ *is a projection.*

2. $\mathbf{P}_t^s$ *is a projection onto* $Z_{t-s}^{\|}$*, i.e.* $range(\mathbf{P}_t^s) = Z_{t-s}^{\|}$.

3. *If all* $\boldsymbol{\theta}_t$ *are orthogonal, then* $\mathbf{P}_t^t = \mathbf{P}_0, \quad \forall t$*, where* $\mathbf{P}_0$ *is the orthogonal projection onto* $Z_0^{\|}$.

Proof:

1. We prove it by induction on s for each t. For $s = 0$, $\mathbf{P}_t^0 = \mathbf{P}_t$, which is a projection by its definition. Suppose it is true for s such that $\mathbf{P}_t^s = \mathbf{P}_t^s\mathbf{P}_t^s$ ($\mathbf{P}$ is a projection if $\mathbf{P} = \mathbf{P}^2$), then for $(s+1)$,

$$\begin{aligned}(\mathbf{P}_t^{s+1})^2 &= \left(\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}\right)^2,\\ &= \boldsymbol{\theta}_{t-s-1}^{-1}\left(\mathbf{P}_t^s\right)^2\boldsymbol{\theta}_{t-s-1},\\ &= \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1},\\ &= \mathbf{P}_t^{s+1}.\end{aligned}$$

2. We prove it by induction on s for each t. For $s = 0$, $\mathbf{P}_t^0 = \mathbf{P}_t$, which is the orthogonal projection onto $Z_t^{\|}$. Suppose that it is true for s such that $\mathbf{P}_t^s$ is a projection onto $Z_{t-s}^{\|}$, then for $(s+1)$, $\mathbf{P}_t^{s+1} = \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}$, which implies

$$\begin{aligned}range(\mathbf{P}_t^{s+1}) &= range(\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s),\\ &= \{\boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{x} : \mathbf{x} \in Z_{t-s}^{\|}\},\\ &= Z_{t-s-1}^{\|}.\end{aligned}$$

3. If $\boldsymbol{\theta}_t$ is orthogonal,

$$\begin{aligned}\mathbf{P}_t^{s+1} &= \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1},\\ &= \boldsymbol{\theta}_{t-s-1}^{T}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1},\\ &= (\mathbf{P}_t^{s+1})^\top.\end{aligned}$$

$\mathbf{P}_t^{s+1}$ is a orthogonal projection onto range $Z_{t-s-1}^{\|}$. Therefore, $\mathbf{P}_t^T$ is a orthogonal projection onto $Z_0^{\|}$, orthogonal projection onto the same range is unique, $\mathbf{P}_t^T = \mathbf{P}_0$, $\forall t$.

■

The following Lemma uses the concept of oblique projection to show a recursive relationship to project any t^{th} state space of Equation (4.9) back to the input data space.

Lemma 4.5.4 *Define for* $0 \leq s \leq t$,

$$\mathbf{G}_t^s := \alpha_t \cdot \mathbf{I} + (1-\alpha_t)\mathbf{P}_t^s.$$

Then, Equation (4.9) *can be written as*

$$\overline{\mathbf{x}}_t - \mathbf{x}_t = (\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0),\ \ t \geq 1.$$

Proof: We prove it by induction on t. For $t = 1$, by the definition of $\mathbf{G}_t^s$ and transformation from Lemma 4.5.2,

$$\begin{aligned}\overline{\mathbf{x}}_1 - \mathbf{x}_1 &= \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0), && Equation(4.9),\\ &= \boldsymbol{\theta}_0(\alpha_0 \cdot \mathbf{I} + (1-\alpha_0)\cdot\mathbf{P}_0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0),\\ &= \boldsymbol{\theta}_0\mathbf{G}_0^0(\overline{\mathbf{x}}_0 - \mathbf{x}_0).\end{aligned}$$

Suppose that it is true for $(\overline{\mathbf{x}}_t - \mathbf{x}_t)$, by Lemma 4.5.2, we have

$$\begin{aligned}
\overline{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{x}}_t - \mathbf{x}_t), \\
&= \boldsymbol{\theta}_t(\alpha_t \cdot \mathbf{I} - (1-\alpha_t) \cdot \mathbf{P}_t)(\overline{\mathbf{x}}_t - \mathbf{x}_t), && \text{Lemma 4.5.2,} \\
&= \boldsymbol{\theta}_t \mathbf{G}_t^0 (\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0). && (4.10)
\end{aligned}$$

Recall the definitions of $\mathbf{P}_t^{(s+1)} := \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}$, and $\mathbf{G}_t^s := \alpha_t \cdot \mathbf{I} + (1-\alpha_t)\mathbf{P}_t^s$,

$$\begin{aligned}
\mathbf{G}_t^{s+1} &= \alpha_t \cdot \mathbf{I} + (1-\alpha_t) \cdot \mathbf{P}_t^{(s+1)}, \\
&= \alpha_t \cdot \mathbf{I} + (1-\alpha_t) \cdot \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{P}_t^s\boldsymbol{\theta}_{t-s-1}, \\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\big(\alpha_t \cdot \mathbf{I} + (1-\alpha_t) \cdot \mathbf{P}_t^s\big)\boldsymbol{\theta}_{t-s-1}, \\
&= \boldsymbol{\theta}_{t-s-1}^{-1}\mathbf{G}_t^s\boldsymbol{\theta}_{t-s-1},
\end{aligned}$$

which results in the equality of the oblique projections. Furthermore,

$$\boldsymbol{\theta}_{t-s-1}\mathbf{G}_t^{(s+1)} = \mathbf{G}_t^s\boldsymbol{\theta}_{t-s-1}.$$

Applying the above to Equation (4.10) results in

$$\begin{aligned}
\overline{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t\mathbf{G}_t^0(\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0), \\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1})\mathbf{G}_t^1(\boldsymbol{\theta}_{t-2}\boldsymbol{\theta}_{t-3}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0), \\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2})\mathbf{G}_t^2(\boldsymbol{\theta}_{t-3}\boldsymbol{\theta}_{t-4}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0), \\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_t^t\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)(\overline{\mathbf{x}}_0 - \mathbf{x}_0).
\end{aligned}$$

■

Lemma 4.5.5 *Let*

$$\mathbf{F}_t := \mathbf{G}_{t-1}^{(t-1)}\mathbf{G}_{t-2}^{(t-2)}\cdots\mathbf{G}_0^0, \quad t \geq 1.$$

Then,

$$\mathbf{F}_t = \prod_{s=0}^{t-1} \alpha_s \cdot \mathbf{I} + (1 - \prod_{s=0}^{t-1} \alpha_s) \cdot \mathbf{P}_0.$$

Proof: We prove it by induction on t. Recall the definition of $\mathbf{G}_t^s := \alpha_t \cdot \mathbf{I} + (1 - \alpha_t) \cdot \mathbf{P}_t^s$. When $t = 1$,

$$\mathbf{F}_1 = \mathbf{G}_0^0 = \alpha_0 \cdot \mathbf{I} + (1 - \alpha_0) \cdot \mathbf{P}_0.$$

Suppose that it is true for t such that

$$\mathbf{F}_t = \prod_{s=0}^{t-1} \alpha_s \cdot \mathbf{I} + (1 - \prod_{s=0}^{t-1} \alpha_s) \cdot \mathbf{P}_0,$$

for $(t + 1)$,

$$\begin{aligned}
\mathbf{F}_{t+1} &= \mathbf{G}_t^t \mathbf{F}_t, \\
&= (\alpha_t \cdot \mathbf{I} + (1 - \alpha_t) \cdot \mathbf{P}_t^t) \mathbf{F}_t, \\
&= (\alpha_t \cdot \mathbf{I} + (1 - \alpha_t) \cdot \mathbf{P}_t^t)(\prod_{s=0}^{t-1} \alpha_s \cdot \mathbf{I} + (1 - \prod_{s=0}^{t-1} \alpha_s) \cdot \mathbf{P}_0), \\
&= \prod_{s=0}^{t} \alpha_s \cdot \mathbf{I} + \alpha_t (1 - \prod_{s=0}^{t-1} \alpha_s) \cdot \mathbf{P}_0 + (1 - \alpha_t) \prod_{s=0}^{t-1} \alpha_s \cdot \mathbf{P}_t^t + (1 - \alpha_t)(1 - \prod_{s=1}^{t-1} \alpha_s) \cdot \mathbf{P}_t^t \mathbf{P}_0.
\end{aligned}$$

Recall Lemma 4.5.3, if all $\boldsymbol{\theta}_t$ is orthogonal, then $\mathbf{P}_t^t = \mathbf{P}_0$, and $\mathbf{P}_t^t \mathbf{P}_0 = \mathbf{P}_0$. Hence,

$$\begin{aligned}
\mathbf{F}_{t+1} &= \prod_{s=0}^{t} \alpha_s \cdot \mathbf{I} + \alpha_t (1 - \prod_{s=0}^{t-1} \alpha_s) \cdot \mathbf{P}_0 + (1 - \alpha_t) \prod_{s=0}^{t-1} \alpha_s \cdot \mathbf{P}_0 + (1 - \alpha_t)(1 - \prod_{s=1}^{t-1} \alpha_s) \cdot \mathbf{P}_0, \\
&= \prod_{s=0}^{t} \alpha_s \cdot \mathbf{I} + \left(\alpha_t - \prod_{s=0}^{t} \alpha_s + \prod_{s=0}^{t-1} \alpha_s - \prod_{s=0}^{t} \alpha_s + 1 - \alpha_t - \prod_{s=0}^{t-1} \alpha_s + \prod_{s=0}^{t} \alpha_s \right) \cdot \mathbf{P}_0, \\
&= \prod_{s=0}^{t} \alpha_s \cdot \mathbf{I} + \left(1 - \prod_{s=0}^{t} \alpha_s \right) \cdot \mathbf{P}_0.
\end{aligned}$$

■

Theorem 4.2.1 *For any time step $t \geq 1$, assuming that each $\boldsymbol{\theta}_t$ is an orthogonal matrix, we*

have the following error computation:

$$\|\overline{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 = \prod_{s=0}^{t-1} \alpha_s^2 \cdot \|\mathbf{z}^{\perp}\|_2^2 + \|\mathbf{z}^{\|}\|_2^2,$$

where α_t is a time-varying parameter defined in relation to the control regularization c, and λ_t are auxiliary variables, as follows:

$$\alpha_t = \frac{c}{1 + \lambda_{t+1} + c}, \quad \lambda_T = 0, \quad \lambda_{T-1} = \frac{c}{1+c}, \quad \lambda_t = \frac{c(1+\lambda_{t+1})}{1 + c + \lambda_{t+1}}.$$

Proof: The input perturbation $\mathbf{z} = \overline{\mathbf{x}}_0 - \mathbf{x}_0$ can be decomposed as $\mathbf{z} = \mathbf{z}^{\|} + \mathbf{z}^{\perp}$, where $\mathbf{z}^{\|} \in Z_0^{\|}$ and $\mathbf{z}^{\perp} \in Z_0^{\perp}$, and $\mathbf{z}^{\|}$ and $\mathbf{z}^{\perp}$ are vectors such that

- $\mathbf{z}^{\|} \cdot \mathbf{z}^{\perp} = 0$ almost surely.
- $\mathbf{z}^{\|}$, $\mathbf{z}^{\perp}$ have uncorrelated components.
- $\mathbf{z}^{\|} \in Z^{\|}$, and $\mathbf{z}^{\perp} \in \mathcal{Z}^{\perp}$.

Since the layer transformations $\boldsymbol{\theta}_t$ are orthogonal matrices for all t, recall the dynamical system Equation (4.9) and Lemma 4.5.4,

$$\begin{aligned}
\|\overline{\mathbf{x}}_t - \mathbf{x}_t\|_2^2 &= \|\boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)\boldsymbol{\theta}_{t-1}(\mathbf{I} - \mathbf{K}_{t-1}) \cdots \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_0)\mathbf{z}\|_2^2, \\
&= \|(\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2} \cdots \boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2, \\
&= \|(\mathbf{G}_{t-1}^{t-1} \cdots \mathbf{G}_0^0)\mathbf{z}\|_2^2, \qquad (4.11)
\end{aligned}$$

For the term $\|(\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)\mathbf{z}\|_2^2$, recall Lemma 4.5.5,

$$
\begin{aligned}
&\|(\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)\mathbf{z}\|_2^2 \\
&= \|\Big(\prod_{s=0}^{t-1}\alpha_s\cdot\mathbf{I}+(1-\prod_{s=0}^{t-1}\alpha_s)\mathbf{P}_0\Big)\mathbf{z}\|_2^2, \\
&= \|\prod_{s=0}^{t-1}\alpha_s\cdot\mathbf{z}+(1-\prod_{s=0}^{t-1}\alpha_s)\cdot\mathbf{z}^{\|}\|_2^2, \\
&= (\prod_{s=0}^{t-1}\alpha_s)^2\cdot\|\mathbf{z}\|_2^2+(1-\prod_{s=0}^{t-1}\alpha_s)^2\cdot\|\mathbf{z}^{\|}\|_2^2+2(\prod_{s=0}^{t-1}\alpha_s)(1-\prod_{s=0}^{t-1}\alpha_s)(\mathbf{z})^{\top}\mathbf{z}^{\|}, \\
&= (\prod_{s=0}^{t-1}\alpha_s)^2\cdot(\|\mathbf{z}^{\|}\|_2^2+\|\mathbf{z}^{\perp}\|_2^2)+(1-\prod_{s=0}^{t-1}\alpha_0)^2\cdot\|\mathbf{z}^{\|}\|_2^2+2(\prod_{s=0}^{t-1}\alpha_s)(1-\prod_{s=0}^{t-1}\alpha_s)(\mathbf{z}^{\|}+\mathbf{z}^{\perp})^{\top}\mathbf{z}^{\|} \\
&= \prod_{s=0}^{t-1}\alpha_s^2\cdot\|\mathbf{z}^{\perp}\|_2^2+\Big(\prod_{s=0}^{t-1}\alpha_s^2+(1-\prod_{s=0}^{t-1}\alpha_s)^2+2(\prod_{s=0}^{t-1}\alpha_s)(1-\prod_{s=0}^{t-1}\alpha_s)\Big)\cdot\|\mathbf{z}^{\|}\|_2^2, \\
&= \prod_{s=0}^{t-1}\alpha_s^2\cdot\|\mathbf{z}^{\perp}\|_2^2+\|\mathbf{z}^{\|}\|_2^2.
\end{aligned}
$$

Recall the error computation in Equation (4.11),

$$\|\overline{\mathbf{x}}_{\epsilon,t}-\mathbf{x}_t\|_2^2=\prod_{s=0}^{t-1}\alpha_s^2\cdot\|\mathbf{z}^{\perp}\|_2^2+\|\mathbf{z}^{\|}\|_2^2.$$

■

Chapter 5

Achieving Asymptotic Fairness of Machine Learning Models via Optimal Control

This chapter investigates fairness issue in a non-stationary environment where the distribution of training data evolves with time, and the deployed model needs to be adjusted accordingly, the adjusted model will influence users' participation and future training data in turn. Consider a scenario wherein a deployed model is continuously fine-tuned based on the majority's input. Minority users may find that their interactions with the machine learning model are unsatisfying. As a result, these minority users tend to not engage with the model. The deployed model, which is already under-representing them, gets even less data from minority users, leading to even worse services to this group

With considerations of the interplay between a machine learning model and users' participation, we propose a model-based optimal control method to engage users from all demographic groups over a long time horizon. The optimal control formulation accounts for long-term planning, therefore it can achieve asymptotic fairness in a non-stationary setting. Different from existing reinforcement learning-based approaches, our model-based optimal control allows more efficient computation, leading to superior performance compared with existing baseline methods. Adopting an optimal control approach to achieve asymptotic fairness poses three main challenges: (1) lack of a fairness metric in a non-stationary environment, (2) complexity of solving the optimal control problem under complicated user dynamics, and (3) lack of a dynamic model describing users participation. To address these challenges, we make the following major

contributions:

1. We introduce the concept of asymptotic fairness to describe the maintenance of performance across all demographic groups over a long time horizon. Existing fairness definitions like equal opportunity [43] and demographic parity [104] focus on measuring performance disparities between different demographic groups at a single time step, thus they are not applicable to a non-stationary setting. Instead, our definition describes how the fairness evolves in a long term. As time approaches to infinity, user engagement dynamics converge towards a state wherein users from all demographic groups are fully participating. Our definition leverages a Markov decision process (MDP) and a deterministic dynamic system that describe the behavior of each individual user and of each demographic group, respectively.

2. We present an optimal control formulation to achieve the asymptotic fairness of a machine learning model. Solving the resulting optimal control problems is particularly challenging due to the complex dynamics of user engagements. To address this challenge, we employ an efficient implementation of Pontryagin's Maximum Principle (PMP) to solve the control problem. We further derive a Hamiltonian under mild assumptions to achieve better convergence.

3. We design a discrete dynamic system as a surrogate model to approximate the complex dynamics of user engagement. Different from reinforcement learning, a model-based control framework relies on a dynamical system to describe the objective function and constraints. Our surrogate model provides a simplified description for the dynamic behavior of users in a complex environment, making solving the model-based optimal control problem and achieving asymptotic fairness feasible.

Figure 5.1 shows the key idea of this work. In Figure 5.1 (a), a machine learning model is updated based on newly generated user data, resulting in increasing bias towards the majority group and disparate population density (calculated as the ratio of participating users and total

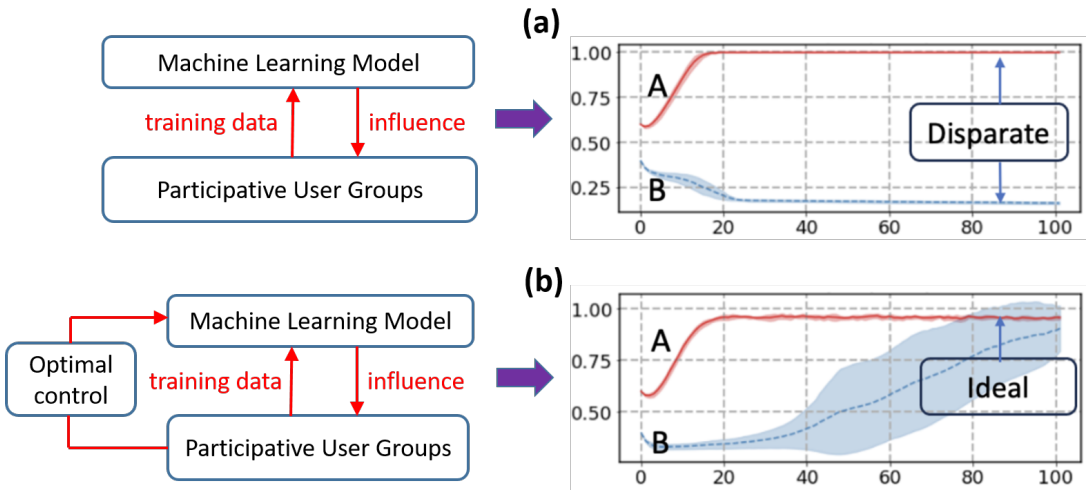


Figure 5.1: (a) illustrates the initial feature distributions for two distinct demographic groups. The variations in feature distribution among engaged users, resulting from population risk minimization and the proposed optimal control method, are presented in (b) and (c). (d) and (e), respectively, reveal the changes in the population densities of these users over time.

users) at terminal step. In Figure 5.1 (b), the proposed optimal control method initially biases the machine learning model towards the minority group to attract more users, and control the long-term behavior of the machine learning model, ultimately leading the population densities of both groups to asymptotically converging to 1.

Table 5.1 further highlights the key difference of our method with main existing baselines. Most existing methods fail to handle the long-term fairness of a machine learning model. Our method and reinforce learning-based methods are among the limited techniques that can provide long-term fair performance, but our method is computationally more efficient due to the employment of a model-based control.

5.1 Fairness in Non-Stationary Environment

In this section, we elaborate on the problem setting to fairness in a non-stationary environment, where user participation depends on the model's performance (Section 5.1.1). We then define asymptotic fairness as a condition to measure the performance of a machine learning model across all demographic groups in a long time horizon (Section 5.1.2). Table 5.2 presents

Table 5.1: Key differences between the proposed optimal control method and baseline approaches. ERM (empirical risk minimization) represents the naive optimization to generate predictive models. Minimax and DRO (distributional robust optimization) are fairness-concerned approaches to generate predictive models. Reinforcement learning (RL) method that take into consideration the underlying dynamics. The proposed optimal control method is more efficient compared to RL-based approaches.

	Fairness	Lone-term Effect	Efficiency
ERM	X	X	X
Minimax	✓	X	X
DRO	✓	X	X
RL	✓	✓	X
Proposed	✓	✓	✓

a list of used notations accompanied by their descriptions.

5.1.1 Problem Setting

We denote a sequence of models as $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$, where model $\boldsymbol{\theta}_t \in \mathbb{R}^m$ and m is the number of parameters. We consider the interplay of the machine learning model with its users consisting of K different demographic groups. Assume that the ith group has N^i users, and the total number of users is $N = \sum_{i=1}^{K} N^i$. A user can either have a status of participating or non-participating. and the n^{th} user is represented by a feature vector ($\mathbf{x}(n) \in \mathbb{R}^d$), a label ($\mathbf{y}(n) \in \mathbb{R}$), and a demographic membership ($\mathbf{z}(n) \in \mathbb{R}$).

We develop a Markov decision process (MDP) to simulate the dynamics of user engagement across different demographic groups as they interact with a machine learning model. This MDP is intuitively explained in Fig. 5.2 (a), and its detailed descriptions are given below.

- **States**: The state is described by a binary vector $\mathbf{s}_t \in \{0,1\}^N$ that indicates whether each user is participating or not at time step t. For instance, the n^{th} user at time t is participating if $\mathbf{s}_t(n) = 1$ and non-participating if $\mathbf{s}_t(n) = 0$.

- **Actions**: Actions are the outputs, $\mathbf{a}_t(n) = \boldsymbol{\theta}_t(\mathbf{x}(n))$, of the machine learning model. An action $\mathbf{a}_t$ consists of N binary elements. Moreover, only actions of participating users

Table 5.2: Notation Table

Symbol	Description
m	Number of model parameters
d	Dimension of user feature
T	Length of time horizon
K	Total number of demographic groups
N^i	Number of users from the i^{th} demographic group
N	Total number of users
$\mathbf{x}(n)$	Feature vector of the n^{th} user
$\mathbf{y}(n)$	Label of the n^{th} user
$\mathbf{z}(n)$	Demographic membership of the n^{th} user
$\boldsymbol{\theta}_t$	Predictive model at time step t
$\boldsymbol{\theta}$	Predictive model $\boldsymbol{\theta} := \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$
λ_t	Population density of participating users from the i^{th} group
$\boldsymbol{\lambda}_t$	Vector of population densities of all demographic groups
$\mathbf{p}_t$	Adjoint state at time step t
F^*	Population retention system (simulation environment)
F	surrogate dynamic system (estimation of the simulation environment)
$\mathbf{s}_t$	State of the simulation environment at time step t
$\mathbf{a}_t$	Action at time step t
$R(\cdot)$	Reward function acting on state $\mathbf{s}_t$
$\Psi(\cdot)$	Terminal objective function acting on terminal state
$\Phi(\cdot)$	Classification loss function (e.g. Cross-entropy)
$H(\cdot)$	Hamiltonian function
$V^{\boldsymbol{\theta}}(\cdot)$	Value function of model $\boldsymbol{\theta}$
γ	Discounting factor
κ^i, g^i, h^i	Functions used to define surrogate dynamic system F

(where $\mathbf{s}_t(n) = 1$) are leveraged by the transition probability that will be specified later.

- **Transition probability**: The transition probability characterizes the changes in the participating status of all users. The status of the n^{th} user at time step $t+1$ is conditioned on both the n^{th} action $\boldsymbol{\theta}_t(\mathbf{x}(n))$ and $\mathbf{s}_t(n)$. Following [105, 106], we assume that user n is more likely to be participating [i.e., $\mathbf{s}_{t+1}(n) = 1$] if the machine learning model $\boldsymbol{\theta}_t$ provides a correct prediction for this particular user. In contrast, a wrong prediction results in a higher possibility of user n stopping using the machine learning model [i.e., $\mathbf{s}_{t+1}(n) = 0$].

This is defined by the transition probability,

$$\mathbf{s}_{t+1} \sim \mathbf{MDP}(\cdot|\mathbf{s}_t, \{\boldsymbol{\theta}_t(\mathbf{x}(n))\}_{n=1}^N, \{\mathbf{y}(n)\}_{n=1}^N, \{\mathbf{z}(n)\}_{n=1}^N),$$
$$\text{where } \mathbf{s}_{t+1}(n) \sim \text{Bernoulli}\left(u\left(P\left[\boldsymbol{\theta}_t(\mathbf{x}_n) = \mathbf{y}(n)\right]\right)\right), \tag{5.1}$$

where $u(\cdot)$ is a utility function $u : [0,1] \to [0,1]$. For example, if $u(\cdot)$ is an identity mapping, then the probability of a participating user continuing to be engaged in the next time step equals the probability of model making a correct prediction. We denote Eq. (5.1) as **population retention system**.

- **Rewards**: At a time t, a reward $R(\mathbf{s}_t)$ is measured from the current state $\mathbf{s}_t$. Let λ_t^i be the population density of participating users from the i$^{\text{th}}$ demographic group,

$$\lambda_t^i = \frac{1}{N^i} \cdot \sum_{n=1}^{N} \mathbf{s}_t(n) \cdot \mathbb{1}_{\mathbf{z}(n)=i},$$

where $\mathbb{1}_{\mathbf{z}(n)=i}$ is a indicator function that returns 1 if $\mathbf{z}(n) = i$, meaning that the n$^{\text{th}}$ user belongs to the i$^{\text{th}}$ group. This density value is between 0 and 1. If λ_t^i increases, it means more users from the i$^{\text{th}}$ demographic group become participating. Conversely, a decrease in λ_t^i suggests users from that group are leaving or becoming non-participating. We compute rewards by measuring the population densities, one example could be the sum of population densities $R(\mathbf{s}_t) = \sum_{i=1}^K \lambda_t^i$, where λ_t^i is computed based on $\mathbf{s}_t$.

- **Initial states**: Given the initial population densities, the initial states are constructed by randomly sampling participating users from each demographic group. For example, consider two demographic groups, each with a total of 10 users. The initial population densities for these groups are 0.7 and 0.1, respectively. To construct the initial state $\mathbf{s}_0 \in \mathbb{R}^{20}$, we randomly select 8 elements to be 1 (representing 7 users from the first group and 1 user from the second group) and the remaining 12 elements to be 0 (representing 3 users from the first group and 9 users from the second group).

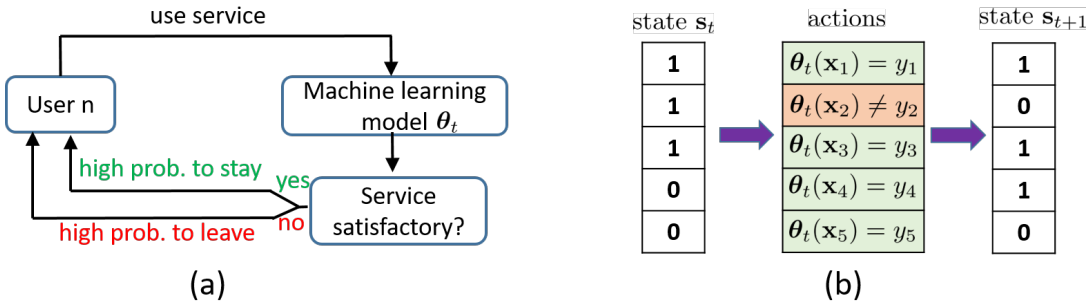


Figure 5.2: (a): The intuition of our designed MDP. A user, once receiving satisfactory service from a machine learning model, is more likely to stay participating. (b) An example of the MDP transition in Eq. (5.1). At time step t, the first, second, and third users are actively participating. However, due to an incorrect prediction made by the model $\boldsymbol{\theta}_t$, the second user becomes non-participating at time step $t+1$. On the other hand, the fourth user, who is non-participating at t, has a slight chance of becoming participating at time step $t+1$.

The setup of the MDP slightly deviates from the general case, where the agent (here the model) can take any action in the action space. In this framework, actions are produced by a predetermined, parameterized model, such as an classifier. Additionally, these actions can only be updated by data obtained from users who are currently participating (from feature vectors $\mathbf{x}(n)$, labels $\mathbf{y}(n)$ and demographic information $\mathbf{z}(n)$ if $\mathbf{s}_t(n) = 1$).

The proposed MDP consists of two stages.

- **Model generation:** Our goal is to create a sequence of T models, represented by $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$, from interacting with the population retention system in Eq. (5.1). At every time step, we use the feature vectors and labels of participating users, along with feedback on rewards, to improve model performance. However, if a user stops participating in the following time step, we no longer have access to their data.

- **Model evaluation:** To assess the performance of the generated models, we deploy them into the population retention system, initiating from a random starting point, and evaluating the reward based on the observed population densities $\lambda_0^i, \lambda_1^i, ..., \lambda_T^i$.

Figure 5.2 (b) provides an example of the proposed MDP. At time step t, the state $\mathbf{s}_t$ reveals that the first three users are actively participating. However, due to an erroneous prediction

by the predictive model $\boldsymbol{\theta}_t(\mathbf{x}(2)) \neq \mathbf{y}(2)$, the second user stops participation in the next time step, leading to $\mathbf{s}_{t+1}(2) = 0$. Regarding non-participating users at t, there is a small chance that they will engage with the model in the following time step, exemplified by the behavior of the fourth user.

5.1.2 The Definition for Asymptotic Fairness

The population retention system, defined in Eq. (5.1), simulates the user's stochastic activity of participating in a machine learning model. In this context, a machine learning model with minimum population risk across the entire population encourages increased user participation. Consequently, active users are more inclined to provide additional training data, which is invaluable for refining the model performance. With these newly contributed data, the machine learning model is further improved. This process progressively improves the predictive accuracy of the machine learning model, thereby reducing population risk even further in each succeeding time step.

Given the above interplay between users and the machine learning model, there exists a positive feedback loop. With sufficient iterations and time, this loop leads to a scenario where the population risks associated with all demographic groups tend towards 0. This occurs as the number of total users denoted as N^i grows significantly larger. Simultaneously, the population densities of all demographic groups approach 1, indicating high levels of engagement and participation across all demographics. This motivates us to define asymptotic fairness as follows:

Definition 5.1.1 (asymptotic fairness) *A sequence of models satisfies asymptotic fairness if the dynamics of population density it drives satisfy the following condition:*

$$\lambda_t^i \to 1 \text{ almost surely, as } t \to \infty \ \ \forall i \in \{1, 2, ..., K\}, \text{ s.t. Eq. (5.1).}$$

When the population densities of participating users from each demographic group converge

towards 1, it indicates that the underlying predictive model consistently performs fairly across all these groups.

Prior research has considered disparity amplification [51] to assess the representation disparity across all demographic groups at each individual time step. The proposed asymptotic fairness provides a more precise interpretation of fairness in a non-stationary environment. 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 defined in [51], 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.

5.2 Optimal Control for Asymptotic Fairness

In this section, we first present an optimal control formulation to achieve asymptotic fairness. Next, we provide an efficient solver based on Pontryagin's maximum principle (PMP) [78]. Finally, we derive a new form of the Hamiltonian under mild assumptions to achieve better convergence.

5.2.1 Optimal Control Formulation

The asymptotic fairness in Definition 5.1.1 requires that over an infinite period, the population density of every demographic group approaches 1. In a practical scenario where there's a finite time horizon, this definition is equivalent to maximizing the population densities of all demographic groups at the terminal time. This can be formulated as an optimal control problem with terminal reward by viewing the machine learning model parameters $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ as control variables.

The model-based optimal control formulation relies on a dynamical system to describe the objective function and constraints. We consider a discrete dynamic system to approximate the

complex dynamics of the MDP in Eq. (5.1):

$$\boldsymbol{\lambda}_{t+1} = \mathbf{f}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t), \text{ given } \boldsymbol{\lambda}_0. \tag{5.2}$$

Here the K-dimensional vector $\boldsymbol{\lambda}_t = [\lambda_t^1, \lambda_t^2, ..., \lambda_t^K]^T$ denote the population density of all user groups. In practice, $\mathbf{f}(\cdot)$ is optimized together with the machine learning model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ by interacting with the simulation environment. The detailed design for this discrete dynamic system is elaborated in Section 5.3. For now, let $\mathbf{f}$ be a continuous function with respect to $\boldsymbol{\lambda}_t$.

Remark 5.2.1 *While not necessarily the best modeling choice, this discrete dynamic system significantly simplifies the state space, reducing its complexity from extremely high to relatively manageable dimensions. Additionally, the state space representing population density is continuous, as opposed to the discrete state space of user participation status. This distinction makes an optimal control solution more feasible.*

Let $\Psi(\boldsymbol{\lambda}_T)$ denote a measurement of the terminal state. As an example, $\Psi(\boldsymbol{\lambda}_T)$ can be the sum of population densities $\Psi(\boldsymbol{\lambda}_T) = \sum_{i=1}^{K} \lambda_T^i$ at the terminal step. Then the model-based optimal control formulation to achieve asymptotic fairness can be formulated as follows:

$$\max_{\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}} \Psi(\boldsymbol{\lambda}_T) \text{ s.t. dynamic sysetm Eq. (5.2).} \tag{5.3}$$

This is a special case of a class of optimal control problems for discrete dynamical systems, in which we consider the control variables as the model parameters at all time steps [56].

5.2.2 Optimal Control Solution via Pontryagin's Maximum Principle

We describe PMP as a general solver for the optimal control problem defined in Eq. (5.3). The PMP [78] consists of two difference equations and a maximization condition. The PMP solves for a set of fixed control parameters for a given initial state. To begin with, we define the Hamiltonian as

$$H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t) := \mathbf{p}_{t+1}^{\top} \cdot \mathbf{f}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) - \mathcal{L}(\boldsymbol{\theta}_t, \boldsymbol{\lambda}_t), \tag{5.4}$$

where $\mathcal{L}(\boldsymbol{\theta}_t, \lambda_t^i)$ is a running loss at time t, which can be defined as the regularization term of model parameters. $\mathbf{p}_t$ is the adjoint state representing the derivative of the terminal cost with respect to $\boldsymbol{\lambda}_t$, $\mathbf{p}_t = \frac{\partial \Psi(\boldsymbol{\lambda}_T)}{\partial \boldsymbol{\lambda}_t}$.

The PMP consists of a two-point boundary value problem,

$$\boldsymbol{\lambda}_{t+1} = \nabla_p H(t, \boldsymbol{\lambda}_t^i, \mathbf{p}_t, \boldsymbol{\theta}_t), \quad \boldsymbol{\lambda}_0 \text{ given}, \tag{5.5}$$

$$\mathbf{p}_t = \nabla_\lambda H(t, \boldsymbol{\lambda}_t^i, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t), \quad \mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T)}{\partial \boldsymbol{\lambda}_T}, \tag{5.6}$$

plus a maximization condition of the Hamiltonian.

$$H(t, \boldsymbol{\lambda}_t^i, \mathbf{p}_t, \boldsymbol{\theta}_t) \geq H(t, \boldsymbol{\lambda}_t^i, \mathbf{p}_t, \boldsymbol{\theta}_t), \ \ \forall\, \boldsymbol{\theta}_t \text{ and } t. \tag{5.7}$$

The PMP establishes a necessary condition for the optimal control solution, the optimal solution maximizes the Hamiltonian, given the states and adjoint states. We consider the method of successive approximation [56,107,108] to solve the PMP. For a given dynamic system $M(\cdot)$, this algorithm iteratively runs Eq. (5.5), (5.6) to generate states $\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1, ..., \boldsymbol{\lambda}_T$ and adjoint states $\mathbf{p}_T$, $\mathbf{p}_{T-1}, ..., \mathbf{p}_0$, and maximizes the Hamiltonian as defined in (5.7) to generate the optimal control solution $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$.

5.2.3 Modified Pontryagin's Maximum Principle

In this subsection, we introduce a modified Hamiltonian that can achieve better convergence for solving the PMP. In addition, we also provide a detailed implementation that iteratively optimizes both the surrogate dynamic system and the control variables (e.g., parameters of the machine learning model).

Deterministic surrogate dynamical system: The proposed optimal control formulation utilizes a deterministic surrogate dynamical system to simplify the complex MDP defined in Eq. (5.1). Specifically, for a given state $\mathbf{s}_t$ that represents the participation status of all users at time t, this MDP produces the next state $\mathbf{s}_{t+1}$ with a certain probability. Among the various possible next states $\mathbf{s}_{t+1}$, let $\mathbf{s}_{t+1}^{\mathrm{ML}}$ denote the most likely one. Then the optimal design of the surrogate dynamical system is to deterministically predict $\mathbf{s}_{t+1}^{\mathrm{ML}}$ from the current state $\mathbf{s}_t$ using the predictive model $\boldsymbol{\theta}_t$.

$$\mathbf{s}_{t+1}^{\mathrm{ML}} = F(\mathbf{s}_t, \boldsymbol{\theta}_t). \tag{5.8}$$

The surrogate dynamical system is not used to describe the dynamic behavior of the MDP exactly. Instead, when utilizing this surrogate system, the optimal control aims to improve model performance based on the most probable dynamics of the users. For example, if the current model predicts incorrectly for an participating user $\mathbf{s}_t(n) = 1$, it is more likely that the user will not participate in the next state $[\mathbf{s}_{t+1}(n) = 0]$. Although there remains a small probability that the user might continue to participate $[\mathbf{s}_{t+1}(n) = 1]$, the optimal control formulation prioritizes the more likely scenario, predicting the user as non-participating in the subsequent state.

Non-decreasing value function: To begin with, let $\boldsymbol{\theta} := \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ represent a sequence of models, where $\boldsymbol{\theta}(\mathbf{x}_t) = \boldsymbol{\theta}_t(\mathbf{x}_t)$ generates deterministic predictions. We denote $V^{\boldsymbol{\theta}}$ as the expected accumulated rewards obtained by deploying the model $\boldsymbol{\theta}$ to the MDP,

$$V^{\boldsymbol{\theta}}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_{t+1} \sim \mathbf{MDP}(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\theta}(\mathbf{s}_t))} \Big[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t) | \mathbf{s}_0 = \mathbf{s} \Big],$$

where γ is a discounting factor, and $\mathbf{MDP}(\mathbf{s}_{t+1}|\mathbf{s}_t, \boldsymbol{\theta}(\mathbf{s}_t))$ is the MDP-based population retention system defined in Eq. (5.1).

Let F^* denote the stochastic process represented by the MDP defined in Eq. (5.1).

Theorem 5.2.1 *Let the value function satisfy L-Lipschitz continuity with a Lipschitz constant L. Suppose F^* is an element of the set $\mathcal{F}$, which denotes the space of estimated systems under consideration. When the optimality of the following objective is achieved,*

$$\boldsymbol{\theta}^{\text{new}}, F^{\text{new}} = \arg\max_{\boldsymbol{\theta}, F} V^{\boldsymbol{\theta}} - \Big(\gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}},\mathbf{MDP}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + \frac{2B\gamma\kappa}{1-\gamma}\Big)$$
$$\text{s.t. } \text{KL}(\boldsymbol{\theta}^{\text{old}}(\mathbf{s}),\boldsymbol{\theta}(\mathbf{s}))^{\frac{1}{2}} \leq \kappa,$$

where F is the surrogate dynamical system defined in Eq. (5.8)*, $\rho^{\boldsymbol{\theta}^{\text{old}},\mathbf{MDP}}$ represents the stationary state distribution of the MDP with model at previous iteration $\boldsymbol{\theta}^{\text{old}}$. $\boldsymbol{\theta}^{\text{new}}$ and F^{new} represent the optimal model and surrogate dynamical system for maximizing the objective function, respectively. $\|\mathbf{s}\| \leq B$. κ is an upper bound on the Kullback–Leibler divergence. Then we have a non-decreasing value function of the MDP from the resulting models,*

$$\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}},\mathbf{MDP}}\Big[V^{\boldsymbol{\theta}^{\text{old}}}(\mathbf{s})\Big] \leq \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}},\mathbf{MDP}}\Big[V^{\boldsymbol{\theta}^{\text{new}}}(\mathbf{s})\Big].$$

The detailed proof is provided in Section 5.5.1.

Theorem 5.2.1 suggests an objective function that optimizes both the value function and the surrogate dynamical system. The central message provided by this Theorem is on the constraint of model parameter updates. To ensure non-decreasing expected accumulated rewards for each model update, an KL-divergence constraint is applied on the predictions of the updated and previous models. This is consistent with existing RL algorithms [109, 110] as constraining the model parameter update from collected simulation data has been shown to improve algorithm convergence. The connection between Theorem 5.2.1 and the concept of asymptotic fairness is that it improves the convergence of the control solution towards an asymptotically fair state by incorporating a KL-divergence regularization term.

A modified Hamiltonian: In practice, the KL-divergence constraint can be approximated by a regularization term. Motivated by the theoretical insights from this Theorem, we design

a modified Hamiltonian of Eq. (5.4),

$$H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t) := \mathbf{p}_{t+1}^T \cdot \mathbf{f}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) - \mathrm{KL}(\boldsymbol{\theta}_t^{\mathrm{old}}, \boldsymbol{\theta}_t)^{\frac{1}{2}} - \frac{2B\gamma\kappa}{1-\gamma}. \tag{5.9}$$

Theorem 5.2.1 considers a surrogate dynamical system that deterministically predicts $\mathbf{s}_{t+1}^{\mathrm{ML}}$ as defined in Eq. (5.8). Its derivation does not rely on a specific form of F. Recall that the simplified surrogate dynamical system defined in Eq. (5.2) acts on population densities $\boldsymbol{\lambda}_t$. This is a special case of the general surrogate dynamical system considered in Eq. (5.8), since population densities can be computed exactly from state $\mathbf{s}_t$. Section 5.3 elaborates on a specific design for $\mathbf{f}$, which allows the theoretical results from Theorem 5.2.1 to be applied. Let $\boldsymbol{\lambda}_t = e(\mathbf{s}_t)$ compute population densities, then the simplified surrogate dynamical system $\mathbf{f}(\cdot)$ can be optimized from $\|\mathbf{f}(e(\mathbf{s}_t), \boldsymbol{\theta}(\mathbf{s})) - e(F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})))\|$.

Practical implementation: Algorithm 3 outlines the implementation steps of the modified PMP for solving the optimal control problem as defined in Eq. (5.3). Starting with the initial population densities for each demographic group, we employ forward propagation following Eq. (5.5) to calculate population densities at subsequent time points. Beginning with the terminal condition $\mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T)}{\partial \boldsymbol{\lambda}_T}$, backward propagation is then performed as per Eq. (5.6) to determine all adjoint states. The process involves iteratively updating a surrogate discrete dynamic system through its interaction with the simulation environment and maximizing the modified Hamiltonian defined in Eq. (5.9).

5.3 Surrogate Dynamic System Design

In this section, we provide a specific choice for the surrogate system outlined in Eq. (5.2). This design features a low-dimensional state representation, thereby improving computational efficiency. A discrete dynamic system describing the population dynamics can be constructed

Algorithm 3 Modified 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**
    $\boldsymbol{\lambda}_{t+1} = \nabla_p H(t, \boldsymbol{\lambda}_t, \mathbf{p}_t, \boldsymbol{\theta}_t)$, // Forward propagation of number densities (Eq. (5.5)).
  end for
  $\mathbf{p}_T = \frac{\partial \Psi(\boldsymbol{\lambda}_T)}{\partial \boldsymbol{\lambda}_T}$, // Set terminal condition.
  for $t = T - 1$ to 0 **do**
    $\mathbf{f}^{\text{new}} = \arg\min_{\mathbf{f}} \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}},\mathbf{MDP}} \left[\|\mathbf{f}(e(\mathbf{s}), \boldsymbol{\theta}(\mathbf{s})) - e(F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})))\| \right]$ // Update surrogate dynamic system.
    $\boldsymbol{\theta}_t^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathbf{p}_{t+1}^T \cdot \mathbf{f}^{\text{new}}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t) - \text{KL}(\boldsymbol{\theta}^{\text{old}}(\mathbf{s}), \boldsymbol{\theta}(\mathbf{s}))^{\frac{1}{2}} - \frac{2B\gamma\kappa}{1-\gamma}$, // Maximize Hamiltonian Eq. (5.9). $\boldsymbol{\lambda}_t = e(\mathbf{s}_t)$.
    $\mathbf{p}_t = \nabla_{\boldsymbol{\lambda}} H(t, \boldsymbol{\lambda}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t)$, // Backward propagation of adjoint state (Eq. (5.6)).
  end for
end for

as follows,

$$\boldsymbol{\lambda}_{t+1} = \mathbf{f}(\boldsymbol{\lambda}_t, \boldsymbol{\theta}_t), \quad \text{where} \quad f_i(\cdot) = g\left(\kappa^i\left(\lambda_t^i, \boldsymbol{\theta}_t\right)\right)\left(1 - \lambda_t^i\right) + h\left(\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)\right)\lambda_t^i, \tag{5.10}$$

where f_i is the ith element of $\mathbf{f}$, $\kappa^i(\cdot)$ measures the model performance on the ith demographic group, $g(\cdot)$ and $h(\cdot)$ compute the rates of incoming and retained users at each time step, respectively. As an example, when $\kappa^i(\cdot)$ measures the model accuracy, $g(\cdot)$ and $h(\cdot)$ are monotonically increasing functions with range from 0 to 1 (e.g., sigmoid function).

The discrete dynamic system yields a low-dimensional state representation, consists only of the population densities across all demographic groups. However, notice that $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ measures the model performance with population density λ_t^i, this requires data selection based on the population density λ_t^i. We consider the distributionally robust optimization (DRO), which facilitates a deterministic generation process of λ_t^i proportion of users who received optimal model performance. To begin with, let $d_{\mathcal{X}^2}(\mathcal{P}||\mathcal{Q})$ denote the $\mathcal{X}^2$-divergence between two probability distributions $\mathcal{P}$ and $\mathcal{Q}$ [1], $\mathcal{B}(\mathcal{P}, r) = \{\mathcal{Q} : d_{\mathcal{X}^2}(\mathcal{P}||\mathcal{Q}) \leq r\}$ denote the chi-squared ball around a probability distribution $\mathcal{P}$ of radius r. Let $\mathcal{P}^i$ be the feature distribution of users

[1] $\mathcal{X}^2$-divergence definition: $d_{\mathcal{X}^2}(\mathcal{P}||\mathcal{Q}) = \int (\frac{d\mathcal{P}}{d\mathcal{Q}} - 1)^2 d\mathcal{Q})$

from the ith demographic group. The performance measure $\kappa^i(\cdot)$ is defined as the worst-case distributional loss over all r-radius balls around $\mathcal{P}^i$,

$$\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) = \sup_{\mathcal{Q} \in \mathcal{B}(\mathcal{P}^i, r_t^i)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{Q}} \Phi(\boldsymbol{\theta}_t, \mathbf{x}, y), \;\; r_t^i = (1/\lambda_t^i - 1)^2. \tag{5.11}$$

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. The computation of the the worst-case distributional loss defined in Eq. (5.11) is a challenging task. Section 5.5.2 provides its dual form for the ease of computation.

Stability of asymptotically fair state: The definition of asymptotic fairness requires that the population densities of each demographic group approach and stabilize at 1 over an infinite period. The subsequent Proposition confirms that an equilibrium state signified by $\boldsymbol{\lambda}_t = \mathbf{1}$ is stable under a certain condition. This suggests that upon reaching the equilibrium state of $\mathbf{1}$, the population densities will remain in this state.

Proposition 5.3.1 *In the surrogate system as described by Eq.* (5.10)*, a equilibrium state with* $\boldsymbol{\lambda}_t = \mathbf{1}$ *is stable if the following condition holds,*

$$\max_{i\in[1,2,\ldots,K]} \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \eta^*}{\partial \lambda_t^i} \cdot \Big(1 - \frac{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big]}{\sqrt{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big]}}\Big) < 1,$$

where

$$\eta^* = \arg\inf_{\eta\in\mathbb{R}} \Big(C(\lambda_t^i) \cdot \big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta]_+^2\big]\big)^{\frac{1}{2}} + \eta\Big), \;\; C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}},$$

where $[a]_+ = a$ if $a \geq 0$ and $[a]_+ = 0$ when $a < 0$, in which case, the DRO is the expected loss of samples that have higher loss than the optimal η^*. The detailed proof is derived in Section 5.5.2.

The above proposition indicates that the stability of an asymptotically fair state depends on

the variance of losses $\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)$ experienced by individual users. Specifically, within any given demographic group, an asymptotically fair state achieves stability when the losses encountered by individual users are similar, indicating consistent performance of the predictive model for all users within the group. This makes sense as uniform losses among users lead to a more consistent DRO loss across varying proportions of users.

5.4 Numerical Experiments

In this section, we begin with detailing experimental settings. Following that, we conduct an empirical validation of our optimal control solution using a synthetic dataset. We also explore two realistic datasets commonly used in fairness research, as outlined.

5.4.1 Experimental Setting

Simulation environment: We implement two simulation environments to realize the population retention systems.

- F_1^*: In this system, when a user decides to stay or leave, their decision follows a Bernoulli distribution conditioned on the model performance of this user. For instance, a user has a higher chance of staying when the model prediction is correct, and low probability of staying engaged when a wrong model prediction is given.

- F_2^*: The second system takes a more complex approach to modeling user retention. A user decides to leave because the model has consecutively resulted three incorrect prediction on this particular user.

Algorithm 4 shows the implementation of the population retention system for evaluation.

Evaluation metrics: We consider 3 evaluation metrics,

- **Loss** measures the terminal population densities $\boldsymbol{\lambda}_T$ using the binary cross-entropy loss function (e.g. $\sum_{i=1}^{K} -\log(\lambda_T^i)$). This metric assesses the variance in population densi-

Algorithm 4 Implementation of Population Retention System for Evaluation.

Input: Dataset $\{\mathbf{x}(n), \mathbf{y}(n), \mathbf{z}(n)\}_{n=1}^{N}$,
A sequence of models $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$,
Initial population densities $\boldsymbol{\lambda}_0$.
Output: Population densities at all time step $\{\boldsymbol{\lambda}_t\}_{t=0}^{T}$.
for episode = 1 to max episodes **do**
  // Set up an initial state by randomly sampling user indices as participating users.
  Initialize $\mathbf{s}_0$.
  for t = 1 to max time steps **do**
    // Return the features of currently participating users.
    Return $\mathbf{x}(n)$ if $\mathbf{s}_t(n) = 1$.
    // Model prediction based on the collected features.
    Predict $\boldsymbol{\theta}_t(\mathbf{x}(n))$ if $\mathbf{s}_t(n) = 1$.
    // Environment update for active users based on model predictions.
    $\mathbf{s}_{t+1} \sim F^*(\cdot|\mathbf{s}_t, \{\boldsymbol{\theta}_t(\mathbf{x}(n))\}_{n=1}^{N}, \{\mathbf{y}(n)\}_{n=1}^{N}, \{\mathbf{z}(n)\}_{n=1}^{N})$.
    // Collect reward.
    $R(\mathbf{s}_t)$.
  end for
end for

ties from a vector of all values set to 1, thereby measuring the distance from achieving asymptotic fairness.

- **Density-2** indicates the population density of the minority demographic group at the terminal time step.

- **Disparity** is the absolute difference between population densities of majority and minority groups at the terminal time step.

Implementation of Baseline Methods: We delve into three categories of algorithms designed for achieving asymptotic fairness: **fairness-agnostic**, **fairness-aware**, and **dynamic-aware approaches**. While fairness-agnostic algorithms employ empirical risk minimization, fairness-aware techniques utilize demographic data to ensure balanced model performance across diverse demographic groups. Additionally, dynamic-aware approaches consider the inherent population dynamics, typically leading to enhanced performance compared to the other types.

- Fairness-agnostic: Empirical risk minimization (**ERM**) optimizes an average loss of all

observable data,

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \frac{1}{\sum_{n=1}^{N} \mathbf{s}_t(n)} \sum_{n=1}^{N} \Phi(\boldsymbol{\theta}_t, \mathbf{x}(n), \mathbf{y}(n)) \cdot \mathbf{s}_t(n),$$

where $\mathbf{s}_t \in \{0,1\}^N$ indicating the participating user indices. This method often results in undesirable outcomes in terms of fairness [51]. Specifically, when there exists a disparity between the population densities of the majority and minority demographic groups at a particular time step, ERM focuses on minimizing the average loss across all samples. Consequently, this produces a model that performs better for the majority group than for the minority group. This not only accentuates the imbalance between the two demographic groups in subsequent time steps but also exacerbates the inherent bias in the model derived in the following steps.

- Fairness-aware: Methods that prioritize fairness use demographic data to ensure equal model performance among various demographic groups. We detail two techniques from this group: Minimax optimization (**Minimax**), and distributional robust optimization (**DRO**).

  - **Minimax** focuses on optimizing the most unfavorable outcome for all groups by using the demographic information of each sample [111].

    $$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \max_{i=[1,2,\ldots,K]} \Big(\frac{1}{\sum_{n=1}^{N} \mathbf{s}_t(n) \cdot \mathbb{1}_{\mathbf{z}(n)=i}} \sum_{n=1}^{N} \Phi(\boldsymbol{\theta}_t, \mathbf{x}(n), \mathbf{y}(n)) \cdot \mathbf{s}_t(n) \cdot \mathbb{1}_{\mathbf{z}(n)=i} \Big),$$

    where $\mathbb{1}_{\mathbf{z}(n)=i}$ is a indicator function that is used to select samples belonging to the i^{th} demographic group.

  - **DRO** can be seen as a milder form of Minimax since it optimizes for the most unfavorable outcome over a specified proportion (represented by λ) of samples.

Notably, the loss from DRO is always greater than or equal to the loss from Minimax.

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

The DRO algorithm is implemented by the dual representation shown in Eq. (5.13).

- Dynamic-aware: This category takes into account the underlying population evolution for optimal decision-making. In general time-evolving environments, optimizing model performance at each time step often cannot lead to the optimal model subject to dynamic change. In the MDP defined in Eq. (5.1), the transition of users' behavior can lead to different decision-making algorithms. We implement various reinforcement learning (RL) algorithms to build dynamic-aware models.

  To begin with, we first define the probability of transitioning from an initial state S_0 to any state $\mathbf{s}$ under model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ and system f for 1 step (we use F^* to indicate the population retention system defined in Eq. (5.1) and f as its estimated surrogate dynamic system defined in Eq. (5.10)),

$$P(\mathbf{s}_0 \to \mathbf{s}, 1, \boldsymbol{\theta}, F) = \int_{\mathbf{a}} \boldsymbol{\theta}(\mathbf{a}|\mathbf{s}_0) f(\mathbf{s}|\mathbf{s}_0, \mathbf{a}) d\mathbf{a},$$

  where $\boldsymbol{\theta}$ indicates a sequence of models $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$. More generally, this transitioning probability admits a recursive form with any steps t, a t-step probability transition can be represented as first transitioning to some intermediate state $\mathbf{s}'$ after $t-1$ steps, then transitioning to $\mathbf{s}$ for one more step,

$$P(\mathbf{s}_0 \to \mathbf{s}, t, \boldsymbol{\theta}, F) = \int_{\mathbf{s}'} P(\mathbf{s}_0 \to \mathbf{s}', t-1, \boldsymbol{\theta}, F) \int_{\mathbf{a}} \boldsymbol{\theta}(\mathbf{a}|\mathbf{s}') f(\mathbf{s}|\mathbf{s}', \mathbf{a}) d\mathbf{a} d\mathbf{s}'.$$

  We define $\rho^{\boldsymbol{\theta},F}$ the stationary state distribution of the MDP under model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ and

system f,

$$\rho^{\boldsymbol{\theta},F} = \int_{\mathbf{s}_0} \mu(\mathbf{s}_0) \int_{\mathbf{s}} \sum_{t=0}^{\infty} \gamma^t \cdot P(\mathbf{s}_0 \to \mathbf{s}, t, \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}, F).$$

We detail three RL algorithms: Naive policy gradient (**PG**), trust region policy optimization (**TRPO**), and proximal policy optimization (**PPO**).

- **PG** [112] is implemented via Monte-Carlo sampling to generate the accumulated reward,

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}} \Big[\mathbb{E}_{\mathbf{a}\sim\boldsymbol{\theta}(\mathbf{a}|\mathbf{s})} \Big[G_t \nabla_{\boldsymbol{\theta}} \ln \boldsymbol{\theta}(\mathbf{a}|\mathbf{s}) \Big] \Big],$$

  where $\rho^{\boldsymbol{\theta},F^*}$ is the stationary state distribution resulting from model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ and the population retention system F^*, $G_t = \sum_{\tau=t}^{T} \gamma^{\tau-t} R(\mathbf{s}_t, A_t)$ represents the accumulated reward from time step t, which is generated from collecting sample trajectories using the current model $\{\boldsymbol{\theta}_\tau\}_{\tau=t}^{T-1}$. This policy gradient implementation is sample-inefficient since every parameter update requires re-collecting the sample trajectory (the stationary state distribution depends on the current model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$).

- **TRPO** [109] makes updates that improve the model parameters while ensuring the new model doesn't deviate too much from the old one, it can produce more stable and reliable learning compared to vanilla policy gradient methods.

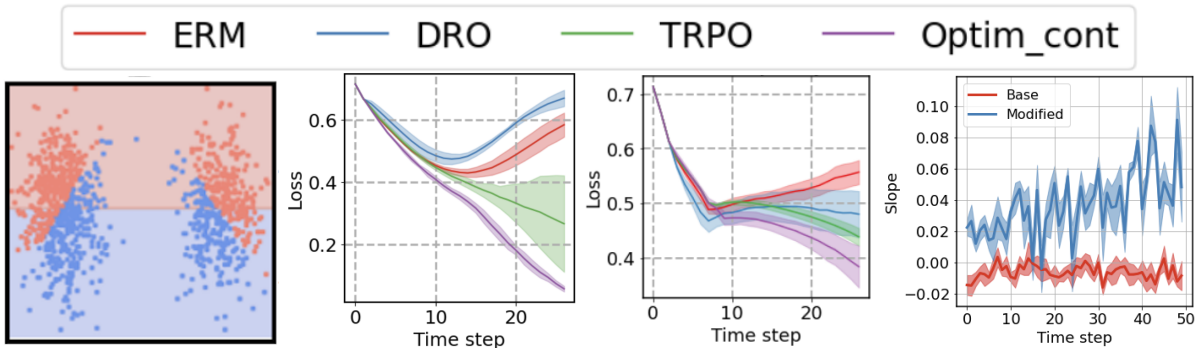
$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{q},F^*}} \Big[\mathbb{E}_{\mathbf{a}\sim\boldsymbol{q}(\mathbf{a}|\mathbf{s})} \Big[\frac{\boldsymbol{\theta}(\mathbf{a}|\mathbf{s})}{\boldsymbol{q}(\mathbf{a}|\mathbf{s})} \Big] G_t \Big],$$

  where $\boldsymbol{q} = \{\boldsymbol{q}_t\}_{t=0}^{T-1}$ is the sequence of models at previous update.

- **PPO** [110] aims to approximate the behavior of TRPO but in a more straightforward and computationally efficient manner.

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\substack{\mathbf{s}\sim\rho^{\boldsymbol{q},F^*} \\ \mathbf{a}\sim\boldsymbol{q}(\mathbf{a}|\mathbf{s})}} \Big[\min \Big(\frac{\boldsymbol{\theta}(\mathbf{a}|\mathbf{s})}{\boldsymbol{q}(\mathbf{a}|\mathbf{s})}, \mathrm{clip}(\frac{\boldsymbol{\theta}(\mathbf{a}|\mathbf{s})}{\mathrm{q}(\mathbf{a}|\mathbf{s})}, 1-\epsilon, 1+\epsilon) \Big) G_t \Big],$$

  where clip$(\cdot)$ is the clip function. PPO simplifies and improves upon the TRPO



(a): Synthetic data (b): Loss in F_1^* (c): Loss in F_2^* (d): Slopes

Figure 5.3: (a) shows the feature distribution of synthetic dataset. (b) and (c) plot the binary cross-entropy losses of population densities resulting from ERM, DRO, TRPO, and the optimal control method in simulation environments F_1^* and F_2^* respectively. (d) plots model slopes resulting from the optimal control method in the base and modified environments.

method, offering a balance between ease of implementation and sample efficiency.

5.4.2 Modeling with Synthetic Dataset

In this study, we employ both F_1^* and F_2^*, with a synthetic binary classification dataset. As depicted in Fig. 5.3 (a), the synthetic dataset is composed of two Gaussian blobs, each centered at different locations of a 2-dimensional space, which 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 0.6 and 0.4 (e.g. majority demographic group has a larger population density compared with the minority demographic group at the initial step). All experiments are repeated with five random seeds.

In the simulation environment F_1^*, user churn is sensitive to model accuracy, as a single incorrect prediction can lead to churn with high probability. This sensitivity is depicted by the sharp increase in the loss trajectory when using ERM, as seen in Fig. 5.3 (b). The DRO method, which is static and fairness-aware, fails to correct this undesired trend. In contrast, TRPO accounts for population dynamics and markedly improves upon the static methods, consistently boosting the population densities of both groups. The optimal control method,

Table 5.3: Synthetic Dataset: Terminal State with Initial $\lambda_0^1 = 0.6$ and $\lambda_0^2 = 0.4$.

Environment F_1^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.323	0.224	0.275	0.481	0.645	0.405	0.920
Disparity ↓	0.677	0.776	0.715	0.459	0.335	0.555	0.05
Loss ↓	0.56	0.76	0.66	0.40	0.23	0.47	0.06
Environment F_2^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.316	0.215	0.348	0.312	0.327	0.314	0.471
Disparity ↓	0.684	0.785	0.652	0.688	0.673	0.686	0.529
Loss ↓	0.556	0.775	0.478	0.555	0.434	0.553	0.402

however, excels by optimally moderating the model across demographics, thereby substantially increasing minority densities with minimal impact on the majority. This is evaluated by the loss metrics in Fig. 5.3 (b), where the optimal control method consistently shows lower losses at every step when compared to other methods. Experimental results from simulation environment F_2^*, in Fig. 5.3 (c), show comparable trends, with smoother transitions. This is due to the slower variation in user churn behavior specific to the environment F_2^* (e.g. 3 consecutive wrong prediction causes a user churn instead of 1).

Here we aim to demonstrate that 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., $t < 50$). We consider a linear classifier where a positive (resp. negative) slope indicates a model favoring the majority demographic (resp. minority) group. Fig. 5.3 (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

Table 5.4: Adult Income: Terminal State with Initial $\lambda_0^1 = 0.6$ and $\lambda_0^2 = 0.4$.

Environment F_1^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.292	0.252	0.291	0.284	0.285	0.285	0.318
Disparity ↓	0.708	0.748	0.709	0.716	0.715	0.715	0.682
Loss ↓	0.615	0.690	0.618	0.630	0.627	0.627	0.573
Environment F_2^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.363	0.315	0.353	0.334	0.334	0.334	0.397
Disparity ↓	0.637	0.685	0.647	0.666	0.666	0.666	0.603
Loss ↓	0.507	0.579	0.521	0.549	0.548	0.549	0.462

adjustment cannot be accomplished by the existing baselines.

5.4.3 Modeling with Adult Income and COMPAS Recidivism Racial Bias.

We explore two real-world datasets: the Adult Income dataset (**Adult**) [113] and the **COMPAS** dataset [114]. The Adult dataset provides information on individual annual incomes, influenced by factors like gender, race, age, and education. COMPAS, on the other hand, is a commercial tool used in the legal system to predict a criminal defendant's likelihood of reoffending. In both datasets, gender attributes are used to differentiate demographic groups. To simulate population dynamics in these static datasets, we follow the same simulation configurations as the experiment with the synthetic dataset in Section 5.4.2. For each demographic group, we randomly select $N^i = 1000$ samples and set initial population densities at $\lambda_0^1 = 0.6$ for the majority group and $\lambda_0^2 = 0.4$ for the minority group.

The outcomes of these simulations, specifically for the Adult dataset, are summarized in Table 5.4, in which the minority population density at the terminal state (Density-2, higher is better), the disparity between two population densities (e.g. $|\lambda_T^1 - \lambda_T^2|$, lower is better), and the loss measures (Loss, lower is better) are presented. These results show that our proposed optimal control method (Optim) outperforms other baseline methods. In environments F_1^* and

Table 5.5: COMPAS: Terminal State with Initial $\lambda_0^1 = 0.6$ and $\lambda_0^2 = 0.4$.

Environment F_1^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.271	0.256	0.184	0.263	0.264	0.257	0.274
Disparity ↓	0.522	0.529	0.153	0.435	0.489	0.442	0.516
Loss ↓	0.770	0.803	1.391	0.848	0.808	0.858	0.764
Environment F_2^*							
	Fair-agnostic	Fair-aware		Dynamic-aware			
	ERM	Minimax	DRO	PG	TRPO	PPO	Optim
Density-2 ↑	0.316	0.298	0.075	0.317	0.317	0.317	0.317
Disparity ↓	0.684	0.702	0.925	0.863	0.683	0.683	0.683
Loss ↓	0.577	0.605	1.296	0.594	0.574	0.574	0.574

F_2^*, it achieves terminal states with $\lambda_T^1 = 1.0$ and $\lambda_T^2 = 0.318$, and $\lambda_T^1 = 1.0$ and $\lambda_T^2 = 0.81$, respectively.

The results from the COMPAS dataset are detailed in Table 5.5, where minority population density, disparity, and loss measures are shown. In this scenario, the optimal control method shows comparable performance to RL-based algorithms. Specifically, TRPO and PPO reach a terminal state with a loss value of 0.574, which achieves the same level of performance compared to the proposed optimal control method. This similarity in performance is attributed to the lesser disparity in representation between different gender attributes within the COMPAS dataset.

5.5 Detailed Theoretical Proofs

5.5.1 Monotone Improvement

In this section, we provide derivation for Theorem 5.2.1. We consider an infinity horizon discounted reward setting, where the reward function $R(\mathbf{s}_t)$ is defined over a state. As defined in Section 5.1.1, the state $\mathbf{s}_t$ includes indices of participating and non-participating users, the reward function can be defined as measuring the population density of participating users, for

instance,

$$R(\mathbf{s}_t) = \sum_{i=1}^{N} \lambda_t^i \text{ where } \lambda_t^i = \frac{1}{N^i} \cdot \sum_{n=1}^{N} \mathbf{s}_t(n) \cdot \mathbb{1}_{\mathbf{z}(n)=i},$$

where this reward function measures the sum of all population densities at time step t.

Let F^* denote the stochastic process represented by the MDP defined in Eq. (5.1), $\tilde{F}(\cdot)$ and $\hat{F}(\cdot)$ denote two stochastic processes that estimate $F^*(\cdot)$. $\boldsymbol{\theta} := \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ represents a sequence of models, where $\boldsymbol{\theta}(\mathbf{x}_t) = \boldsymbol{\theta}_t(\mathbf{x}_t)$ generates deterministic predictions. We denote $V^{\boldsymbol{\theta},F^*}$ as the value function obtained by deploying model $\boldsymbol{\theta}$ to the stochastic process $F^*(\cdot)$,

$$V^{\boldsymbol{\theta},F^*}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_{t+1}\sim F^*(\mathbf{s}_{t+1}|\mathbf{s}_t,\boldsymbol{\theta}(\mathbf{s}_t))}\Big[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t)|\mathbf{s}_0 = \mathbf{s}\Big],$$

where γ is a discounting factor. In addition, for the estimated stochastic processes $\tilde{F}(\cdot)$ and $\hat{F}(\cdot)$,

$$V^{\boldsymbol{\theta},\tilde{F}}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_{t+1}\sim \tilde{F}(\mathbf{s}_{t+1}|\mathbf{s}_t,\boldsymbol{\theta}(\mathbf{s}_t))}\Big[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t)|\mathbf{s}_0 = \mathbf{s}\Big],$$

and

$$V^{\boldsymbol{\theta},\hat{F}}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_{t+1}\sim \hat{F}(\mathbf{s}_{t+1}|\mathbf{s}_t,\boldsymbol{\theta}(\mathbf{s}_t))}\Big[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t)|\mathbf{s}_0 = \mathbf{s}\Big].$$

The proof is structured as follows,

- We first calculate the difference between the value functions of two distinct stochastic processes, $V^{\boldsymbol{\theta},\tilde{F}} - V^{\boldsymbol{\theta},\hat{F}}$ (See Lemma 5.5.1).
- We assume that the value function satisfies an L-Lipschitz condition to a certain norm, and determine an upper bound for the difference between the value functions resulting from the population retention system and an estimated dynamical system (e.g. surrogate dynamic system) (See Proposition 5.5.1).
- We discuss the challenge of optimizing the aforementioned upper bound. To address this, we further refine this upper bound. (See Proposition 5.5.2).

To begin with, we define the probability of transitioning from an initial state $\mathbf{s}_0$ to any state

$\mathbf{s}$ under the predictive model $\boldsymbol{\theta}$ and dynamical system $F(\cdot)$ for 1 step,

$$P(\mathbf{s}_0 \rightarrow \mathbf{s}, 1, \boldsymbol{\theta}, F^*) = F^*(\mathbf{s}|\mathbf{s}_0, \boldsymbol{\theta}_0(\mathbf{s}_0)),$$

this transitioning probability admits a recursive form with any steps t. A t-step probability transition can be represented as first transitioning to some intermediate state $\mathbf{s}'$ after $t-1$ steps, then transitioning to $\mathbf{s}$ for one more step,

$$P(\mathbf{s}_0 \rightarrow \mathbf{s}, t, \boldsymbol{\theta}, F^*) = \int_{\mathbf{s}'} P(\mathbf{s}_0 \rightarrow \mathbf{s}', t-1, \boldsymbol{\theta}, F^*) \cdot F^*(\mathbf{s}|\mathbf{s}', \boldsymbol{\theta}(\mathbf{s}'))d\mathbf{s}'.$$

Moreover, we define $\rho^{\boldsymbol{\theta}, F^*}$ as the stationary state distribution,

$$\rho^{\boldsymbol{\theta}, F^*} = \int_{\mathbf{s}_0} \mu(\mathbf{s}_0) \int_{\mathbf{s}} \sum_{t=0}^{\infty} \gamma^t P(\mathbf{s}_0 \rightarrow \mathbf{s}, t, \boldsymbol{\theta}, F^*), \tag{5.12}$$

where γ is the discounting factor, and we assume that the initial state $\mathbf{s}_0$ follows uniform distribution over the state space (e.g. $\mu(\mathbf{s}_0)$ has equal probability for all possible $\mathbf{s}_0$).

For any predictive model $\{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$, the following lemma quantifies the discrepancy between $V^{\boldsymbol{\theta}, \tilde{F}} - V^{\boldsymbol{\theta}, \hat{F}}$.

Lemma 5.5.1 *For any predictive model $\boldsymbol{\theta}$ and two distinct dynamical systems, $\tilde{F}$ and $\hat{F}$, the following holds true,*

$$V^{\boldsymbol{\theta}, \tilde{F}} - V^{\boldsymbol{\theta}, \hat{F}} = \gamma \cdot \mathbb{E}_{\mathbf{s} \sim \rho^{\boldsymbol{\theta}, \tilde{F}}} \Big[\mathbb{E}_{\mathbf{s}' \sim \tilde{F}(\mathbf{s}'|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}') \big] - \mathbb{E}_{\hat{\mathbf{s}} \sim \hat{F}(\hat{\mathbf{s}}|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta}, \hat{F}}(\hat{\mathbf{s}}) \big] \Big].$$

Proof: We denote $w_j(\mathbf{s}_0)$ as the discounted rewards computed from a dynamical system $\tilde{F}$

for the first j steps and another dynamical system $\hat{F}$ starting from the $(\mathrm{j}+1)^{\mathrm{th}}$ step.

$$
\begin{aligned}
w_j(\mathbf{s}_0) &= \sum_{t=0}^{j} \gamma^t \int_{\mathbf{s}} P(\mathbf{s}_0 \to \mathbf{s}, t, \boldsymbol{\theta}, \tilde{F}) \cdot R(\mathbf{s}) d\mathbf{s} \\
&+ \gamma^{j+1} \int_{\mathbf{s}} P(\mathbf{s}_0 \to \mathbf{s}, j, \boldsymbol{\theta}, \tilde{F}) \int_{\mathbf{s}'} \hat{F}(\mathbf{s}'|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})) \cdot V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}') d\mathbf{s}' d\mathbf{s}, \\
= \sum_{t=0}^{j} \gamma^t \Big(\mathbb{E}_{\mathbf{s} \sim P(\mathbf{s}_0 \to \mathbf{s}, t, \boldsymbol{\theta}, \tilde{F})} \big[R(\mathbf{s}) \big] \Big) &+ \gamma^{j+1} \Big(\mathbb{E}_{\mathbf{s} \sim P(\mathbf{s}_0 \to \mathbf{s}, j, \boldsymbol{\theta}, \tilde{F})} \big[\mathbb{E}_{\hat{\mathbf{s}} \sim \hat{F}(\hat{\mathbf{s}}|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta}, \hat{F}}(\hat{\mathbf{s}}) \big] \big] \Big).
\end{aligned}
$$

Moreover, $w_{j+1}(\mathbf{s}_0)$ at time step $t+1$ can be shown similarly,

$$
\begin{aligned}
&w_{j+1}(\mathbf{s}_0) \\
&= \sum_{t=0}^{j} \gamma^t \Big(\mathbb{E}_{\mathbf{s} \sim P(\mathbf{s}_0 \to \mathbf{s}, t, \boldsymbol{\theta}, \tilde{F})} [R(\mathbf{s})] \Big) + \gamma^{j+1} \Big(\mathbb{E}_{\mathbf{s} \sim P(\mathbf{s}_0 \to \mathbf{s}, j, \boldsymbol{\theta}, \tilde{F})} \big[\mathbb{E}_{\mathbf{s}' \sim F(\mathbf{s}'|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}') \big] \big] \Big).
\end{aligned}
$$

Notice that the difference between $w_j(\mathbf{s}_0)$ and $w_{j+1}(\mathbf{s}_0)$ lies in the transitioning dynamical system from state $\mathbf{s}_j$ to $\mathbf{s}_{j+1}$ ($w_j(\mathbf{s}_0)$ relies on $\hat{F}$ and $w_{j+1}(\mathbf{s}_0)$ relies on $\tilde{F}$). From the definitions of $w_j(\mathbf{s}_0)$ and $w_{j+1}(\mathbf{s}_0)$, the discrepancy between the value functions $V^{\boldsymbol{\theta}, \tilde{F}}(\mathbf{s}_0)$ and $V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}_0)$ can be reformulated in terms of the definition of $w_j(\mathbf{s}_0)$,

$$
V^{\boldsymbol{\theta}, \tilde{F}}(\mathbf{s}_0) - V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}_0) = \sum_{j=0}^{\infty} w_{j+1}(\mathbf{s}_0) - w_j(\mathbf{s}_0) = w_\infty(\mathbf{s}_0) - w_0(\mathbf{s}_0),
$$

since $V^{\boldsymbol{\theta}, \tilde{F}}(\mathbf{s}_0) = w_\infty(\mathbf{s}_0)$, and $V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}_0) = w_0(\mathbf{s}_0)$.

For a given j, each term $w_{j+1}(\mathbf{s}_0) - w_j(\mathbf{s}_0)$ can be computed based on their definitions,

$$
\begin{aligned}
&w_{j+1}(\mathbf{s}_0) - w_j(\mathbf{s}_0) \\
&= \gamma^{j+1} \Bigg(\mathbb{E}_{\mathbf{s} \sim P(\mathbf{s}_0 \to \mathbf{s}, j, \boldsymbol{\theta}, \tilde{F})} \Big[\mathbb{E}_{\mathbf{s}' \sim \tilde{F}(\mathbf{s}'|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} [V^{\boldsymbol{\theta}, \hat{F}}(\mathbf{s}')] - \mathbb{E}_{\hat{\mathbf{s}} \sim \hat{F}(\hat{\mathbf{s}}|\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))} [V^{\boldsymbol{\theta}, \hat{F}}(\hat{\mathbf{s}})] \Big] \Bigg).
\end{aligned}
$$

Recall that $\rho^{\boldsymbol{\theta}, \tilde{F}} = \int_{\mathbf{s}_0} \mu(\mathbf{s}_0) \int_S \sum_{t=0}^{\infty} \gamma^t P(\mathbf{s}_0 \to \mathbf{s}, t, \boldsymbol{\theta}, \tilde{F})$, the expected value of $V^{\boldsymbol{\theta}, \tilde{F}}(\mathbf{s}_0) -$

$V^{\boldsymbol{\theta},\hat{F}}(\mathbf{s}_0)$ with respect to $\mathbf{s}_0$ can be computed as follows,

$$
\begin{aligned}
&V^{\boldsymbol{\theta},\tilde{F}} - V^{\boldsymbol{\theta},\hat{F}} \\
&= \int_{\mathbf{s}_0} \mu(\mathbf{s}_0)\big[V^{\boldsymbol{\theta},\tilde{F}}(\mathbf{s}_0) - V^{\boldsymbol{\theta},\hat{F}}(\mathbf{s}_0)\big], \\
&= \sum_{j=0}^{\infty} \gamma^{j+1} \cdot \int_{\mathbf{s}_0} \mu(\mathbf{s}_0) \cdot \mathbb{E}_{\mathbf{s}\sim P(\mathbf{s}_0\to\mathbf{s},j,\boldsymbol{\theta},\tilde{F})}\Big[\mathbb{E}_{\substack{\mathbf{s}'\sim\tilde{F}(\mathbf{s}'|\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) \\ \hat{\mathbf{s}}\sim\hat{F}(\hat{\mathbf{s}}|\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))}}[V^{\boldsymbol{\theta},\hat{F}}(\mathbf{s}') - V^{\boldsymbol{\theta},\hat{F}}(\hat{\mathbf{s}})]\Big], \\
&= \gamma \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F}}\Big[\mathbb{E}_{\mathbf{s}'\sim\tilde{F}(\mathbf{s}'|\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))}\big[V^{\boldsymbol{\theta},\hat{F}}(\mathbf{s}')\big] - \mathbb{E}_{\hat{\mathbf{s}}\sim\hat{F}(\hat{\mathbf{s}}|\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))}\big[V^{\boldsymbol{\theta},\hat{F}}(\hat{\mathbf{s}})\big]\Big].
\end{aligned}
$$

■

For a given state $\mathbf{s}_t$ that represents the participation status of all users at time t, F^* produces the next state $\mathbf{s}_{t+1}$ with a certain probability. Among the various possible next states $\mathbf{s}_{t+1}$, let $\mathbf{s}_{t+1}^{\mathrm{ML}}$ denote the most likely one and $F(\mathbf{s}_t, \boldsymbol{\theta}_t)$ denote the surrogate dynamical system that deterministically predict $\mathbf{s}_{t+1}^{\mathrm{ML}}$ from the current state $\mathbf{s}_t$ using the predictive model $\boldsymbol{\theta}_t$. Let $V^{\boldsymbol{\theta},F}(\mathbf{s})$ denote the value function of the surrogate dynamical system F and model $\boldsymbol{\theta}$,

$$
V^{\boldsymbol{\theta},F}(\mathbf{s}) = \mathbb{E}_{\mathbf{s}_{t+1}\sim F(\mathbf{s}_{t+1}|\mathbf{s}_t,\boldsymbol{\theta}(\mathbf{s}_t))}\Big[\sum_{t=0}^{\infty}\gamma^t R(\mathbf{s}_t)|\mathbf{s}_0 = \mathbf{s}\Big],
$$

Proposition 5.5.1 *Suppose that the value function $V^{\boldsymbol{\theta},F^*}$ on any stochastic process is L-Lipschitz with respect to some norm $\|\cdot\|$ in the sense that*

$$
|V^{\boldsymbol{\theta},F^*}(\mathbf{s}) - V^{\boldsymbol{\theta},F^*}(\mathbf{s}')| \le L \cdot \|\mathbf{s} - \mathbf{s}'\|, \quad \forall \mathbf{s}, \mathbf{s}'.
$$

and assume that the underlying dynamical system is deterministic, then the following establishes an upper bound for the discrepancy between the value functions $V^{\boldsymbol{\theta},F}$ of an estimated dynamical system F and the value function of the true environment $V^{\boldsymbol{\theta},F^*}$,

$$
|V^{\boldsymbol{\theta},F} - V^{\boldsymbol{\theta},F^*}| \le \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big].
$$

Proof: According to Lemma 5.5.1,

$$
\begin{aligned}
&|V^{\boldsymbol{\theta},F^*} - V^{\boldsymbol{\theta},F}| \\
&= \gamma \cdot \Big| \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}} \Big[\mathbb{E}_{\mathbf{s}'\sim F^*(\mathbf{s}'|\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta},F}(\mathbf{s}') \big] - \mathbb{E}_{\hat{S}\sim F(\hat{\mathbf{s}}|\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))} \big[V^{\boldsymbol{\theta},F}(\hat{\mathbf{s}}) \big] \Big] \Big|, \\
&= \gamma \cdot \Big| \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}} \Big[V^{\boldsymbol{\theta},F}(F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))) - V^{\boldsymbol{\theta},F}(F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))) \Big] \Big|, \quad \text{(Deterministic)} \\
&\leq \gamma \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}} \Big[|V^{\boldsymbol{\theta},F}(F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))) - V^{\boldsymbol{\theta},F}(F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})))| \Big], \\
&\leq \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}} \Big[\|F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\| \Big]. \quad \text{(L} - \text{Lipschitz)}
\end{aligned}
$$

■

Given two transition kernels $\mathbf{P}$ and $\mathbf{P}'$, the accumulated state transitions resulting from $\mathbf{P}$ and $\mathbf{P}'$ are $\sum_{t=0}^{\infty}(\gamma\mathbf{P})^t = (I-\gamma\mathbf{P})^{-1}$, and $\sum_{t=0}^{\infty}(\gamma\mathbf{P}')^t = (I-\gamma\mathbf{P}')^{-1}$, respectively. The subsequent Corollary establishes an upper bound for the difference between two state distributions that emerge from these kernels.

Corollary 5.5.1 *Let μ be a distribution over the state space, $d = (1-\gamma)(I-\gamma\mathbf{P})^{-1}\mu$, and $d' = (1-\gamma)(I-\gamma\mathbf{P}')^{-1}\mu$ denote the discounted distribution starting from μ induced by the transitions $\mathbf{P}$ and $\mathbf{P}'$. Then,*

$$|d - d'|_1 \leq \frac{\gamma}{1-\gamma}|(\mathbf{P}-\mathbf{P}')d'|_1.$$

Proof:

$$
\begin{aligned}
|d - d'|_1 &= (1-\gamma)\cdot|(\mathbf{I}-\gamma\mathbf{P})^{-1}\mu - (\mathbf{I}-\gamma\mathbf{P}')^{-1}\mu|_1, \\
&= (1-\gamma)\cdot|\Big((\mathbf{I}-\gamma\mathbf{P})^{-1}\Big((\mathbf{I}-\gamma\mathbf{P}') - (\mathbf{I}-\gamma\mathbf{P})\Big)(\mathbf{I}-\gamma\mathbf{P}')^{-1}\mu\Big)|_1, \\
&= (1-\gamma)\cdot|\Big((\mathbf{I}-\gamma\mathbf{P})^{-1}(\gamma\mathbf{P}-\gamma\mathbf{P}')(\mathbf{I}-\gamma\mathbf{P}')^{-1}\mu\Big)|_1, \\
&\leq |\gamma(\mathbf{P}-\mathbf{P}')(\mathbf{I}-\gamma\mathbf{P}')^{-1}\mu|_1, \\
&= \frac{\gamma}{1-\gamma}|(\mathbf{P}-\mathbf{P}')d'|_1,
\end{aligned}
$$

where $(1-\gamma)\cdot|(\mathbf{I}-\gamma\mathbf{P})^{-1}|_1 \leq 1$. ■

Consider two sequences of predictive models, denoted as $\boldsymbol{\theta} = \{\boldsymbol{\theta}_t\}_{t=0}^{T-1}$ and $\boldsymbol{\theta}' = \{\boldsymbol{\theta}'_t\}_{t=0}^{T-1}$. The stationary state distributions resulting from $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ are $\rho^{\boldsymbol{\theta},F^*}$ and $\rho^{\boldsymbol{\theta}',F^*}$. The subsequent Corollary establishes an upper bound for the stationary state distributions that emerge from $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$.

Corollary 5.5.2 *The following holds true for $\rho^{\boldsymbol{\theta},F^*}$ and $\rho^{\boldsymbol{\theta}',F^*}$,*

$$|\rho^{\boldsymbol{\theta},F^*} - \rho^{\boldsymbol{\theta}',F^*}|_1 \leq \frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[\mathrm{KL}(\boldsymbol{\theta}(\mathbf{s}),\boldsymbol{\theta}'(\mathbf{s}))^{\frac{1}{2}}\Big].$$

Proof: Recall Corollary 5.5.1, given two state distributions $\rho^{\boldsymbol{\theta},F^*}$ and $\rho^{\boldsymbol{\theta}',F^*}$,

$$|\rho^{\boldsymbol{\theta},F^*} - \rho^{\boldsymbol{\theta}',F^*}|_1 \leq \frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[|P_{F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))} - P_{F^*(S,\boldsymbol{\theta}'(\mathbf{s}))}|_1\Big],$$

where $P_{F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))}$ represents the probability distribution of $F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))$. Since $P_{F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))}$ is a mapping from a state action pair to a probability,

$$\frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[|P_{F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))} - P_{F^*(S,\boldsymbol{\theta}'(\mathbf{s}))}|_1\Big] \leq \frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[|P_{\boldsymbol{\theta}(\mathbf{s})} - P_{\boldsymbol{\theta}'(\mathbf{s})}|_1\Big],$$

where $P_{\boldsymbol{\theta}(\mathbf{s})}$ represents the probability distribution of $\boldsymbol{\theta}(\mathbf{s})$. Based on Pinkser's inequality,

$$\frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[|P_{\boldsymbol{\theta}(\mathbf{s})} - P_{\boldsymbol{\theta}'(\mathbf{s})}|_1\Big] \leq \frac{\gamma}{1-\gamma}\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}',F^*}}\Big[\mathrm{KL}(\boldsymbol{\theta}(\mathbf{s}),\boldsymbol{\theta}'(\mathbf{s}))^{\frac{1}{2}}\Big],$$

where $\mathrm{KL}(\cdot,\cdot)$ represents the Kullback–Leibler divergence of two distributions. ■

Recall in Proposition 5.5.1, $|V^{\boldsymbol{\theta},F} - V^{\boldsymbol{\theta},F^*}| \leq \gamma\cdot L\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big]$. In the stationary state distribution $\rho^{\boldsymbol{\theta},F^*}$, the upper bound has explicit dependence on the model parameter $\boldsymbol{\theta}$. However, this complex dependency on $\boldsymbol{\theta}$ complicates the process of incorporating this upper bound into any objective function for the purpose of optimizing model parameters. The following Proposition further refines this upper bound to convert the explicit dependence on $\boldsymbol{\theta}$ to a reference model $\boldsymbol{\theta}^{ref}$.

Proposition 5.5.2 *Assume that the dynamical system is deterministic. Consider the value function $V^{\boldsymbol{\theta},F}$ for the estimated dynamical model F, which is L-Lipschitz. Also, assume that the state space is uniformly bounded by B. Under these conditions, we can determine an upper bound for the difference between the value functions $V^{\boldsymbol{\theta},F}$ of the estimated dynamical system F and the value function corresponding to the actual environment F^*.*

$$|V^{\boldsymbol{\theta},F} - V^{\boldsymbol{\theta},F^*}| \leq \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{ref}},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + 2B\kappa\frac{\gamma}{1-\gamma},$$

where κ is an upper bound on the Kullback–Leibler divergence between $\boldsymbol{\theta}^{\text{ref}}$ and $\boldsymbol{\theta}$.

Proof: For any distributions ρ and ρ' and function $f(\cdot)$, we have

$$\begin{aligned}\mathbb{E}_{\mathbf{s}\sim\rho}F(\mathbf{s}) &= \mathbb{E}_{\mathbf{s}\sim\rho'}F(\mathbf{s}) + < \rho - \rho', f > \\ &\leq \mathbb{E}_{\mathbf{s}\sim\rho'}F(\mathbf{s}) + \|\rho - \rho'\|_1 \cdot \|f\|_\infty.\end{aligned}$$

Recall Proposition 5.5.1 and apply this inequality,

$$\begin{aligned}&|V^{\boldsymbol{\theta},F} - V^{\boldsymbol{\theta},F^*}| \\ &\leq \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big], \\ &\leq \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{ref}},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + \|\rho^{\boldsymbol{\theta},F^*} - \rho^{\boldsymbol{\theta}^{\text{ref}},F^*}\|_1 \cdot \|F\|_\infty.\end{aligned}$$

Recall Corollary 5.5.2,

$$\|\rho^{\boldsymbol{\theta},F^*} - \rho^{\boldsymbol{\theta}^{\text{ref}},F^*}\|_1 \leq \frac{\gamma}{1-\gamma} \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{ref}},F^*}}\Big[\text{KL}(\boldsymbol{\theta}(\mathbf{s}),\boldsymbol{\theta}^{\text{ref}}(\mathbf{s}))^{\frac{1}{2}}\Big] \leq \frac{\gamma}{1-\gamma} \cdot \kappa,$$

where $\text{KL}(\boldsymbol{\theta}(\mathbf{s}),\boldsymbol{\theta}^{\text{ref}}(\mathbf{s}))^{\frac{1}{2}} \leq \kappa$. Since the state space is uniformly bounded by B,

$$\max_S \|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\| \leq \max_S \|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\| + \max_S \|F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\| \leq 2B.$$

Therefore,

$$|V^{\boldsymbol{\theta},F} - V^{\boldsymbol{\theta},F^*}| \leq \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{ref}},F^*}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + 2B\cdot\kappa\cdot\frac{\gamma}{1-\gamma}.$$

■

Theorem 5.2.1 *Let the value function satisfy L-Lipschitz continuity with a Lipschitz constant L. Suppose F^* is an element of the set $\mathcal{F}$, which denotes the space of estimated systems under consideration. When the optimality of the following objective is achieved,*

$$\boldsymbol{\theta}^{\text{new}}, F^{\text{new}} = \arg\max_{\boldsymbol{\theta},F} V^{\boldsymbol{\theta}} - \Big(\gamma\cdot L\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}},\mathbf{MDP}}}\Big[\|F(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + \frac{2B\gamma\kappa}{1-\gamma}\Big)$$
$$\text{s.t. } \text{KL}(\boldsymbol{\theta}^{\text{old}}(\mathbf{s}),\boldsymbol{\theta}(\mathbf{s}))^{\frac{1}{2}} \leq \kappa,$$

where F is the surrogate dynamical system defined in Eq. (5.8)*, $\rho^{\boldsymbol{\theta}^{\text{old}},\mathbf{MDP}}$ represents the stationary state distribution of the MDP with model at previous iteration $\boldsymbol{\theta}^{\text{old}}$. $\boldsymbol{\theta}^{\text{new}}$ and F^{new} represent the optimal model and surrogate dynamical system for maximizing the objective function, respectively. $\|\mathbf{s}\| \leq B$. κ is an upper bound on the Kullback–Leibler divergence. Then we have a non-decreasing value function of the MDP from the resulting models,*

$$\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}},\mathbf{MDP}}}\Big[V^{\boldsymbol{\theta}^{\text{old}}}(\mathbf{s})\Big] \leq \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}},\mathbf{MDP}}}\Big[V^{\boldsymbol{\theta}^{\text{new}}}(\mathbf{s})\Big].$$

Proof: Recall Proposition 5.5.2, at the current iteration,

$$V^{\boldsymbol{\theta}^{\text{new}},F^{\text{new}}} - \Big(\gamma\cdot L\cdot\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}},F^*}}\Big[\|F^{\text{new}}(\mathbf{s},\boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s},\boldsymbol{\theta}(\mathbf{s}))\|\Big] + \frac{2B\gamma\kappa}{1-\gamma}\Big) \leq V^{\boldsymbol{\theta}^{\text{new}},F^*},$$

since the second term measures the difference between the estimated system and the population retention system F^*, $V^{\boldsymbol{\theta}^{\text{new}},F^*}$ leads to 0 difference.

Since $\boldsymbol{\theta}^{\text{new}}$ and F^{new} attain the optimal for the following objective function,

$$\boldsymbol{\theta}^{\text{new}}, F^{\text{new}} = \arg\max_{\boldsymbol{\theta}, M} V^{\boldsymbol{\theta}, F} - \Big(\gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}, F^*}}\Big[\|F(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))\|\Big] + \frac{2B\gamma\kappa}{1-\gamma}\Big)$$
$$\text{s.t. } \text{KL}(\boldsymbol{\theta}_t^{\text{old}}(\mathbf{s}), \boldsymbol{\theta}_t(\mathbf{s}))^{\frac{1}{2}} \leq \kappa,$$

$$\begin{aligned}
&V^{\boldsymbol{\theta}^{\text{new}}, F^{\text{new}}} - \Big(\gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}, F^*}}\Big[\|F^{\text{new}}(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))\|\Big] + \frac{2B\gamma\kappa}{1-\gamma}\Big) \\
&\geq V^{\boldsymbol{\theta}^{\text{new}}, F^{\text{new}}} - \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{new}}, f^*}}\Big[\|F^{\text{new}}(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))\|\Big], \\
&\geq V^{\boldsymbol{\theta}^{\text{old}}, F^*} - \gamma \cdot L \cdot \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}, F^*}}\Big[\|F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s})) - F^*(\mathbf{s}, \boldsymbol{\theta}(\mathbf{s}))\|\Big], \\
&= V^{\boldsymbol{\theta}^{\text{old}}, F^*},
\end{aligned}$$

the second term equals to 0 since the oracle dynamical system F^* is considered. Therefore,

$$\mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}}, \mathbf{MDP}}\Big[V^{\boldsymbol{\theta}^{\text{old}}}(\mathbf{s})\Big] \leq \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\theta}^{\text{old}}}, \mathbf{MDP}}\Big[V^{\boldsymbol{\theta}^{\text{new}}}(\mathbf{s})\Big].$$

■

5.5.2 Analysis for the Surrogate System

Reformulation for Distributional Robust Optimization Loss In general, computing the worst-case distributional loss over a set of distributions is a challenging task. Fortunately, the maximization problem in Eq. (5.11) can be reformulated into its dual form [115]. 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:

$$\sup_{\mathcal{Q}\in\mathcal{B}(\mathcal{P}^i, r_t^i)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{Q}}\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) = \inf_{\eta\in\mathbb{R}}\Big(C(\lambda_t^i)\cdot\big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta]_+^2\big]\big)^{\frac{1}{2}} + \eta\Big),$$
$$\text{where } C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}}, \tag{5.13}$$

where $[x]_+ = x$ if $x \geq 0$ and 0 otherwise. This dual form provides an intuitive interpretation of the DRO loss. At each time step t, given $\boldsymbol{\theta}_t$ and λ_t^i, the DRO 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.

Stability Analysis for the Surrogate System

Proposition 5.3.1 *In the surrogate system as described by Eq.* (5.10)*, a equilibrium state with* $\boldsymbol{\lambda}_t = \mathbf{1}$ *is stable if the following condition holds,*

$$\max_{i \in [1,2,\ldots,K]} \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \eta^*}{\partial \lambda_t^i} \cdot \Big(1 - \frac{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big]}{\sqrt{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big]}}\Big) < 1,$$

where

$$\eta^* = \arg\inf_{\eta \in \mathbb{R}} \Big(C(\lambda_t^i) \cdot \big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta]_+^2\big]\big)^{\frac{1}{2}} + \eta\Big), \;\; C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}},$$

Proof: Recall the surrogate dynamic system in Eq. (5.10), for the i$^{\text{th}}$ demographic group,

$$\begin{aligned}
\lambda_{t+1}^i &= g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) \cdot (1 - \lambda_t^i) + h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) \cdot \lambda_t^i, \\
&= g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) + (h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) - g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)) \cdot \lambda_t^i.
\end{aligned}$$

We take the derivative of λ_{t+1}^i with respect to λ_t^i,

$$\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i} = \frac{\partial g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} + \Big[\frac{\partial h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} - \frac{\partial g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}\Big] \cdot \lambda_t^i + h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) - g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t).$$

When $\lambda_t^i = 1$, suppose that both birth rate and survival rate functions reach their maximum value of 1, this can simplify the above expression as follows,

$$\begin{aligned}
\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i} &= \frac{\partial h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} + h \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) - g \circ \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t), \\
&= \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i}.
\end{aligned}$$

Recall the definition of $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ as defined in Eq. (5.13),

$$\begin{aligned}\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t) &= \inf_{\eta \in \mathbb{R}} \Big(C(\lambda_t^i) \cdot \big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta]_+^2\big]\big)^{\frac{1}{2}} + \eta \Big), \;\; C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}}, \\ &= C(\lambda_t^i) \cdot \big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+^2\big]\big)^{\frac{1}{2}} + \eta^*, \;\; C(\lambda_t^i) = (2(1/\lambda_t^i - 1)^2 + 1)^{\frac{1}{2}},\end{aligned}$$

in which we use η^* as the optimal η that leads to the infimum, in which case, η^* is dependent on λ_t^i given $\boldsymbol{\theta}_t$. The derivative of $\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)$ with respect to λ_t^i can be derived as follows,

$$\begin{aligned}&\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} \\ &= \frac{\partial C(\lambda_t^i)}{\partial \lambda_t^i} \cdot \Big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+^2\big]\Big)^{\frac{1}{2}} + C(\lambda_t^i) \cdot \frac{\partial \Big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+^2\big]\Big)^{\frac{1}{2}}}{\partial \lambda_t^i} + \frac{\partial \eta^*}{\partial \lambda_t^i},\end{aligned}$$

where

$$\frac{\partial C(\lambda_t^i)}{\partial \lambda_t^i} = \frac{1}{2}(2(1/\lambda_t^i - 1)^2 + 1)^{-\frac{1}{2}} \cdot 4(1/\lambda_t^i - 1) \cdot (-(\lambda_t^i)^{-2}),$$

when $\lambda_t^i = 1$, $\frac{\partial C(\lambda_t^i)}{\partial \lambda_t^i} = 0$, and $C(\lambda_t^i) = 1$. Therefore,

$$\begin{aligned}\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} &= \frac{\partial \Big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+^2\big]\Big)^{\frac{1}{2}}}{\partial \lambda_t^i} + \frac{\partial \eta^*}{\partial \lambda_t^i}, \\ &= \frac{1}{2}\Big(\mathbb{E}_{\mathcal{P}^i}\big[[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+^2\big]\Big)^{-\frac{1}{2}} \cdot \mathbb{E}_{\mathcal{P}^i}\big[2 \cdot [\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y) - \eta^*]_+\big](-\frac{\partial \eta^*}{\partial \lambda_t^i}) + (\frac{\partial \eta^*}{\partial \lambda_t^i}),\end{aligned}$$

notice that when $\lambda_t^i = 1$, η^* approaches 0 as this leads to population risk in the DRO formulation. Therefore,

$$\frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} = \Big(\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big]\Big)^{-\frac{1}{2}} \cdot \mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big] \cdot (-\frac{\partial \eta^*}{\partial \lambda_t^i}) + (\frac{\partial \eta^*}{\partial \lambda_t^i}).$$

Therefore,

$$\begin{aligned}
\frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i} &= \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)}{\partial \lambda_t^i} \\
&= \frac{\partial h}{\partial \kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \Big(\Big(\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big] \Big)^{-\frac{1}{2}} \cdot \mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big] \cdot (-\frac{\partial \eta^*}{\partial \lambda_t^i}) + (\frac{\partial \eta^*}{\partial \lambda_t^i}) \Big), \\
&= \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \eta^*}{\partial \lambda_t^i} \cdot \Big(1 - \frac{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big]}{\sqrt{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big]}} \Big).
\end{aligned}$$

For a difference equation, an equilibrium state is stable if the maximum eigenvalue of the Jacobian matrix evaluated at this state is less than 1. Since the Jacobian matrix of the surrogate dynamic system is a diagonal matrix, its eigenvalues are the diagonal elements.

$$\max_{i \in [1,2,\ldots,K]} \frac{\partial \lambda_{t+1}^i}{\partial \lambda_t^i}\Big|_{\lambda_t^i = 1} < 1,$$

which leads to the following condition,

$$\max_{i \in [1,2,\ldots,K]} \frac{\partial h}{\kappa^i(\lambda_t^i, \boldsymbol{\theta}_t)} \cdot \frac{\partial \eta^*}{\partial \lambda_t^i} \cdot \Big(1 - \frac{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)\big]}{\sqrt{\mathbb{E}_{\mathcal{P}^i}\big[\Phi(\boldsymbol{\theta}_t, \mathbf{x}, y)^2\big]}} \Big) < 1.$$

■

Chapter 6

Conclusion

6.1 Summary

This dissertation has made significant advances in addressing the critical challenges of robustness and fairness in deep neural networks through the innovative application of optimal control theory. Chapter 3 presents a novel self-healing framework, it not only improves the robustness of neural networks against perturbations but also connects a closed-loop control approach for error correction during inference. Chapter 4 extends this methodology through a PID control framework, which further enables the practical applicability of these solutions to large-scale models, ensuring faster and more efficient online inference. Most importantly, by addressing the often-neglected issue of fairness in machine learning, chapter 5 introduces a novel algorithm for achieving asymptotic fairness across demographic groups, thereby mitigating the negative feedback loops that exacerbate demographic disparities. The comprehensive approach of this dissertation, spanning from theoretical foundations to practical applications, sets a new benchmark for future research in making machine learning models not only more robust and accurate but also fair. This work not only contributes valuable insights to the field of machine learning but also paves the way for the development of more robust, fair, and trustworthy AI systems.

6.2 Contributions

This dissertation presents three significant contributions: the development of self-healing robust neural networks through closed-loop control, the implementation of PID control for self-healing mechanisms, and the introduction of the concept of asymptotic fairness. More specifically,

- A self-healing framework designed to improve the robustness of a neural network against a wide range of unexpected perturbations through a closed-loop control formulation. The effective implementation of this approach leverages the method of successive approximations, facilitating its application across neural networks of varying depths and sizes. Additionally, the framework includes a comprehensive error analysis in its broadest application.

- A novel PID control framework has been developed to facilitate self-healing capability, improving the robustness of Large Language Models during online inference processes. This framework is efficiently implemented, drawing upon principles of linearity and orthogonality, which permits the application of PID control mechanisms across extensive language models with more than 1 billion model parameters.

- The concept of asymptotic fairness describes the maintenance of performance across all demographic groups over an extended period. This goal of ensuring asymptotic fairness is framed as an optimal control problem. The proposed optimal control formulation yields models that exhibit consistently non-diminishing performance with each update, prompting a new interpretation of the Hamiltonian in Pontryagin's Minimum Principle.

6.3 Future Works

Future work stemming from this dissertation could extend in several promising directions to further improve the robustness and fairness of deep neural networks. One area involves

exploring the integration of advanced machine learning techniques with optimal control theory to develop more sophisticated self-healing mechanisms that can adapt to a broader range of perturbations and biases. This could include the application of reinforcement learning for dynamic adjustment of control parameters in real time. Additionally, extending the PID control framework to incorporate learning-based controllers could offer improved performance and adaptability to complex and evolving data distributions. Another avenue for future research is the development of more comprehensive measures and models to quantify and address fairness in machine learning systems, especially in highly dynamic environments. This could involve creating simulation environments to test the effectiveness of asymptotic fairness strategies under various conditions and identifying new challenges that arise from evolving data and user interaction patterns. Further, Investigating the scalability of these approaches to support increasingly large and complex neural network architectures is also crucial, as is their applicability across different domains beyond those explored in this dissertation. Lastly, addressing the ethical implications of automated decision-making and control in machine learning systems, particularly in sensitive applications, is essential. Developing frameworks that not only improve robustness and fairness but also ensure privacy will be key to the responsible deployment of these advanced models.

Bibliography

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples*, *arXiv preprint arXiv:1412.6572* (2014).

[2] F. Croce and M. Hein, *Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks*, in *International conference on machine learning*, pp. 2206–2216, PMLR, 2020.

[3] J. Bughin, J. Seong, J. Manyika, M. Chui, and R. Joshi, *Notes from the ai frontier: Modeling the impact of ai on the world economy*, *McKinsey Global Institute* **4** (2018), no. 1.

[4] E. Commission *et. al.*, *Proposal for a regulation laying down harmonised rules on artificial intelligence*, *Brussels* **21** (2021) 2021.

[5] J. Buolamwini and T. Gebru, *Gender shades: Intersectional accuracy disparities in commercial gender classification*, in *Conference on fairness, accountability and transparency*, pp. 77–91, PMLR, 2018.

[6] J. Dastin, *Amazon scraps secret ai recruiting tool that showed bias against women*, in *Ethics of data and analytics*, pp. 296–299. Auerbach Publications, 2022.

[7] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, *Membership inference attacks against machine learning models*, in *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18, IEEE, 2017.

[8] W. N. Price and I. G. Cohen, *Privacy in the age of medical big data*, *Nature medicine* **25** (2019), no. 1 37–43.

[9] G. D. P. R. GDPR, *General data protection regulation*, *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC* (2016).

[10] N. Burkart and M. F. Huber, *A survey on the explainability of supervised machine learning*, *Journal of Artificial Intelligence Research* **70** (2021) 245–317.

[11] A. Bibal, M. Lognoul, A. De Streel, and B. Frénay, *Legal requirements on explainability in machine learning*, *Artificial Intelligence and Law* **29** (2021) 149–169.

[12] C. Kohler, *The eu cybersecurity act and european standards: an introduction to the role of european standardization*, *International Cybersecurity Law Review* **1** (2020), no. 1 7–12.

[13] D. Marculescu, D. Stamoulis, and E. Cai, *Hardware-aware machine learning: Modeling and optimization*, in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, IEEE, 2018.

[14] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks*, *arXiv preprint arXiv:1312.6199* (2013).

[15] A. Nguyen, J. Yosinski, and J. Clune, *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436, 2015.

[16] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, *A survey of deep learning techniques for autonomous driving*, *Journal of Field Robotics* **37** (2020), no. 3 362–386.

[17] A. S. Lundervold and A. Lundervold, *An overview of deep learning in medical imaging focusing on mri*, *Zeitschrift für Medizinische Physik* **29** (2019), no. 2 102–127.

[18] J. Yang, K. Zhou, Y. Li, and Z. Liu, *Generalized out-of-distribution detection: A survey*, *International Journal of Computer Vision* (2024) 1–28.

[19] L. Yu, Y. Wang, and X.-S. Gao, *Adversarial parameter attack on deep neural networks*, in *International Conference on Machine Learning*, pp. 40354–40372, PMLR, 2023.

[20] B. Li, Q. He, G. Cui, X. Xia, F. Chen, H. Jin, and Y. Yang, *Read: Robustness-oriented edge application deployment in edge computing environment*, *IEEE Transactions on Services Computing* **15** (2020), no. 3 1746–1759.

[21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et. al.*, *Learning transferable visual models from natural language supervision*, in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.

[22] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, *Adversarial attacks and defences: A survey*, *arXiv preprint arXiv:1810.00069* (2018).

[23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, *Towards deep learning models resistant to adversarial attacks*, *arXiv preprint arXiv:1706.06083* (2017).

[24] S. Luan, Z. Gu, L. B. Freidovich, L. Jiang, and Q. Zhao, *Out-of-distribution detection for deep neural networks with isolation forest and local outlier factor*, *IEEE Access* **9** (2021) 132980–132989.

[25] M. Woodard, S. S. Sarvestani, and A. R. Hurson, *A survey of research on data corruption in cyber–physical critical infrastructure systems*, *Advances in Computers* **98** (2015) 59–87.

[26] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, *Deep learning with limited numerical precision*, in *International conference on machine learning*, pp. 1737–1746, PMLR, 2015.

[27] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, *et. al.*, *Mixed precision training*, *arXiv preprint arXiv:1710.03740* (2017).

[28] P. S. Zuchowski, P. A. Habitz, J. D. Hayes, and J. H. Oppold, *Process and environmental variation impacts on asic timing*, in *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, pp. 336–342, IEEE, 2004.

[29] F. Dabiri and M. Potkonjak, *Hardware aging-based software metering*, in *2009 Design, Automation & Test in Europe Conference & Exhibition*, pp. 460–465, IEEE, 2009.

[30] Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, S. Zhao, H. Larochelle, D. Englund, *et. al.*, *Deep learning with coherent nanophotonic circuits*, *Nature photonics* **11** (2017), no. 7 441–446.

[31] A. Athalye, N. Carlini, and D. Wagner, *Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples*, *arXiv preprint arXiv:1802.00420* (2018).

[32] Z. Gan, Y.-C. Chen, L. Li, C. Zhu, Y. Cheng, and J. Liu, *Large-scale adversarial training for vision-and-language representation learning*, *Advances in Neural Information Processing Systems* **33** (2020) 6616–6628.

[33] D. Zhang, T. Zhang, Y. Lu, Z. Zhu, and B. Dong, *You only propagate once: Accelerating adversarial training via maximal principle*, in *Advances in Neural Information Processing Systems*, pp. 227–238, 2019.

[34] K. Bernhard and J. Vygen, *Combinatorial optimization: Theory and algorithms*, *Springer, Third Edition, 2005.* (2008).

[35] T. He, J. Liu, K. Cho, M. Ott, B. Liu, J. Glass, and F. Peng, *Analyzing the forgetting problem in pretrain-finetuning of open-domain dialogue response models*, in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 1121–1133, 2021.

[36] F. Tramer and D. Boneh, *Adversarial training and robustness for multiple perturbations*, *Advances in neural information processing systems* **32** (2019).

[37] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, *Pixeldefend: Leveraging generative models to understand and defend against adversarial examples*, *arXiv preprint arXiv:1710.10766* (2017).

[38] P. Samangouei, M. Kabkab, and R. Chellappa, *Defense-gan: Protecting classifiers against adversarial attacks using generative models*, *arXiv preprint arXiv:1805.06605* (2018).

[39] R. Binns, *Fairness in machine learning: Lessons from political philosophy*, in *Conference on fairness, accountability and transparency*, pp. 149–159, PMLR, 2018.

[40] A. Rajkomar, M. Hardt, M. D. Howell, G. Corrado, and M. H. Chin, *Ensuring fairness in machine learning to advance health equity*, *Annals of internal medicine* **169** (2018), no. 12 866–872.

[41] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, *A survey on bias and fairness in machine learning*, *ACM computing surveys (CSUR)* **54** (2021), no. 6 1–35.

[42] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, *Fairness through awareness*, in *Proceedings of the 3rd innovations in theoretical computer science conference*, pp. 214–226, 2012.

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

[44] T. Gajdos and J.-M. Tallon, *Fairness under uncertainty*, *Economics Bulletin* **4** (2002), no. 18 1–7.

[45] S. Barocas, M. Hardt, and A. Narayanan, *Fairness and machine learning: Limitations and opportunities*. MIT press, 2023.

[46] Z. C. Lipton, *The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.*, *Queue* **16** (2018), no. 3 31–57.

[47] D. Madras, E. Creager, T. Pitassi, and R. Zemel, *Learning adversarially fair and transferable representations*, in *International Conference on Machine Learning*, pp. 3384–3393, PMLR, 2018.

[48] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, *et. al.*, *Lora: Low-rank adaptation of large language models*, in *International Conference on Learning Representations*, 2021.

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

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

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

[52] J. Yang, A. A. Soltan, D. W. Eyre, and D. A. Clifton, *Algorithmic fairness and bias mitigation for clinical machine learning with deep reinforcement learning*, *Nature Machine Intelligence* **5** (2023), no. 8 884–894.

[53] T. Yin, R. Raab, M. Liu, and Y. Liu, *Long-term fairness with unknown dynamics*, *Advances in Neural Information Processing Systems* **36** (2024).

[54] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, *Neural ordinary differential equations*, in *Advances in neural information processing systems*, pp. 6571–6583, 2018.

[55] E. Haber and L. Ruthotto, *Stable architectures for deep neural networks*, *Inverse Problems* **34** (2017), no. 1 014004.

[56] Q. Li, L. Chen, C. Tai, and E. Weinan, *Maximum principle based algorithms for deep learning*, *Journal of Machine Learning Research* **18** (2018) 1–29.

[57] C. Finlay, J.-H. Jacobsen, L. Nurbekyan, and A. Oberman, *How to train your neural ode: the world of jacobian and kinetic regularization*, in *International conference on machine learning*, pp. 3154–3164, PMLR, 2020.

[58] M. Habiba and B. A. Pearlmutter, *Neural ordinary differential equation based recurrent neural network model*, in *2020 31st Irish signals and systems conference (ISSC)*, pp. 1–6, IEEE, 2020.

[59] Q. Li, L. Chen, C. Tai, and E. Weinan, *Maximum principle based algorithms for deep learning*, *The Journal of Machine Learning Research* **18** (2017), no. 1 5998–6026.

[60] A. Tang, F. Hsiao, D. Murphy, I.-N. Ku, J. Liu, S. D'Souza, N.-Y. Wang, H. Wu, Y.-H. Wang, M. Tang, *et. al.*, *A low-overhead self-healing embedded system for ensuring high yield and long-term sustainability of 60ghz 4gb/s radio-on-a-chip*, in *IEEE International Solid-State Circuits Conference*, pp. 316–318, 2012.

[61] J. Lee, S. Bhagavatula, S. Bhunia, K. Roy, and B. Jung, *Self-healing design in deep scaled CMOS technologies*, *Journal of Circuits, Systems, and Computers* **21** (2012), no. 06 1240011.

[62] A. Goyal, M. Swaminathan, A. Chatterjee, D. C. Howard, and J. D. Cressler, *A new self-healing methodology for RF amplifier circuits based on oscillation principles*, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **20** (2011), no. 10 1835–1848.

[63] J. Y.-C. Liu, A. Tang, N.-Y. Wang, Q. J. Gu, R. Berenguer, H.-H. Hsieh, P.-Y. Wu, C. Jou, and M.-C. F. Chang, *A V-band self-healing power amplifier with adaptive feedback bias control in 65 nm cmos*, in *IEEE Radio Frequency Integrated Circuits Symposium*, pp. 1–4, 2011.

[64] C. Chien, A. Tang, F. Hsiao, and M.-C. F. Chang, *Dual-control self-healing architecture for high-performance radio SoCs*, *IEEE Design & Test of Computers* **29** (2012), no. 6 40–51.

[65] G. Keskin, J. Proesel, and L. Pileggi, *Statistical modeling and post manufacturing configuration for scaled analog CMOS*, in *IEEE Custom Integrated Circuits Conference*, pp. 1–4, 2010.

[66] B. Sadhu, M. A. Ferriss, A. S. Natarajan, S. Yaldiz, J.-O. Plouchart, A. V. Rylyakov, A. Valdes-Garcia, B. D. Parker, A. Babakhani, S. Reynolds, *et. al.*, *A linearized, low-phase-noise vco-based 25-ghz pll with autonomic biasing*, *IEEE Journal of Solid-State Circuits* **48** (2013), no. 5 1138–1150.

[67] S. Sun, F. Wang, S. Yaldiz, X. Li, L. Pileggi, A. Natarajan, M. Ferriss, J.-O. Plouchart, B. Sadhu, B. Parker, *et. al.*, *Indirect performance sensing for on-chip self-healing of analog and RF circuits*, *IEEE Transactions on Circuits and Systems I: Regular Papers* **61** (2014), no. 8 2243–2252.

[68] J. C. Zhang and M. Styblinski, *Yield and variability optimization of integrated circuits*. Springer Science & Business Media, 2013.

[69] M. Wang, F. Yang, C. Yan, X. Zeng, and X. Hu, *Efficient bayesian yield optimization approach for analog and sram circuits*, in *Design Automation Conference*, pp. 1–6, 2017.

[70] X. Li, P. Gopalakrishnan, Y. Xu, and L. T. Pileggi, *Robust analog/RF circuit design with projection-based performance modeling*, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **26** (2006), no. 1 2–15.

[71] C. Cui, K. Liu, and Z. Zhang, *Chance-constrained and yield-aware optimization of photonic ICs with non-gaussian correlated process variations*, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **39** (2020), no. 12 4958–4970.

[72] R. P. Wool, *Self-healing materials: a review*, *Soft Matter* **4** (2008), no. 3 400–418.

[73] G. Wypych, *Self-healing materials: principles and technology*. Elsevier, 2022.

[74] I. Rajapakse and M. Groudine, *On emerging nuclear order*, *Journal of Cell Biology* **192** (2011), no. 5 711–721.

[75] C. Fefferman, S. Mitter, and H. Narayanan, *Testing the manifold hypothesis*, *Journal of the American Mathematical Society* **29** (2016), no. 4 983–1049.

[76] J. Schmidhuber, *Deep learning in neural networks: An overview*, *Neural networks* **61** (2015) 85–117.

[77] R. Bellman, *On the theory of dynamic programming*, *Proceedings of the National Academy of Sciences of the United States of America* **38** (1952), no. 8 716.

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

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

[80] F. Chernousko and A. Lyubushin, *Method of successive approximations for solution of optimal control problems*, *Optimal Control Applications and Methods* **3** (1982), no. 2 101–114.

[81] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, *Defense against adversarial attacks using high-level representation guided denoiser*, in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, 2018.

[82] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, *Theoretically principled trade-off between robustness and accuracy*, in *International Conference on Machine Learning*, pp. 7472–7482, PMLR, 2019.

[83] F. Croce and M. Hein, *Minimally distorted adversarial examples with a fast adaptive boundary attack*, in *International Conference on Machine Learning*, pp. 2196–2205, PMLR, 2020.

[84] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, *Square attack: a query-efficient black-box adversarial attack via random search*, in *European Conference on Computer Vision*, pp. 484–501, Springer, 2020.

[85] J. Long, E. Shelhamer, and T. Darrell, *Fully convolutional networks for semantic segmentation*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[86] V. Badrinarayanan, A. Kendall, and R. Cipolla, *Segnet: A deep convolutional encoder-decoder architecture for image segmentation*, *IEEE transactions on pattern analysis and machine intelligence* **39** (2017), no. 12 2481–2495.

[87] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *Ssd: Single shot multibox detector*, in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[88] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, *The pascal visual object classes (voc) challenge*, *International journal of computer vision* **88** (2010), no. 2 303–338.

[89] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, *Advances in neural information processing systems* **30** (2017).

[90] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti, *et. al.*, *Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model*, *arXiv preprint arXiv:2201.11990* (2022).

[91] E, *A proposal on machine learning via dynamical systems*, *Communications in Mathematics and Statistics* **5** (2017), no. 1 1–11.

[92] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, *SIAM review* **51** (2009), no. 3 455–500.

[93] S. Bowman, G. Angeli, C. Potts, and C. D. Manning, *A large annotated corpus for learning natural language inference*, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, 2015.

[94] A. Williams, N. Nangia, and S. Bowman, *A broad-coverage challenge corpus for sentence understanding through inference*, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, 2018.

[95] J. Y. Yoo and Y. Qi, *Towards improving adversarial training of nlp models*, in *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 945–956, 2021.

[96] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, *Word-level textual adversarial attacking as combinatorial optimization*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6066–6080, 2020.

[97] J. Li, S. Ji, T. Du, B. Li, and T. Wang, *Textbugger: Generating adversarial text against real-world applications*, in *26th Annual Network and Distributed System Security Symposium*, 2019.

[98] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, *Is bert really robust? a strong baseline for natural language attack on text classification and entailment*, in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 8018–8025, 2020.

[99] Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, and D. Kiela, *Adversarial nli: A new benchmark for natural language understanding*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, 2020.

[100] C. Zhu, Y. Cheng, Z. Gan, S. Sun, T. Goldstein, and J. Liu, *Freelb: Enhanced adversarial training for natural language understanding*, in *International Conference on Learning Representations*, 2019.

[101] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*, *arXiv preprint arXiv:1910.01108* (2019).

[102] J. D. M.-W. C. Kenton and L. K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, in *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[103] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Roberta: A robustly optimized bert pretraining approach*, *arXiv preprint arXiv:1907.11692* (2019).

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

[105] C. Riedl, F. Köbler, S. Goswami, and H. Krcmar, *Tweeting to feel connected: A model for social connectedness in online social networks*, *International Journal of Human-Computer Interaction* **29** (2013), no. 10 670–687.

[106] P. Huang, N. H. Lurie, and S. Mitra, *Searching for experience on the web: An empirical examination of consumer behavior for search and experience goods*, *Journal of marketing* **73** (2009), no. 2 55–69.

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

[108] Z. Chen, Q. Li, and Z. Zhang, *Self-healing robust neural networks via closed-loop control*, *Journal of Machine Learning Research* **23** (2022), no. 319 1–54.

[109] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, *Trust region policy optimization*, in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

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

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

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

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

[114] M. Barenstein, *Propublica's compas data revisited*, *arXiv preprint arXiv:1906.04711* (2019).

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