Journal of Machine Learning Research 23 (2022) 1-54 Submitted 5/22; Revised 9/22; Published 10/22

Self-Healing Robust Neural Networks via Closed-Loop Control

Zhuotong Chen ZTCHEN@UCSB.EDU
Department of Electrical and Computer Engineering
University of California at Santa Barbara,
Santa Barbara, CA, USA

Qianxiao Li QIANXIAO@NUS.EDU.SG
Department of Mathematics,
National University of Singapore
Singapore, 119076

Zheng Zhang ZHENGZHANG@ECE.UCSB.EDU
Department of Electrical and Computer Engineering
University of California at Santa Barbara,
Santa Barbara, CA, USA

Editor: Manuel Gomez-Rodriguez

Abstract

Despite the wide applications of neural networks, there have been increasing concerns about their vulnerability issue. While numerous attack and defense techniques have been developed, this work investigates the robustness issue from a new angle: can we design a self-healing neural network that can automatically detect and fix the vulnerability issue by itself? A typical self-healing mechanism is the immune system of a human body. This biology-inspired idea has been used in many engineering designs but has rarely been investigated in deep learning. This paper considers the post-training self-healing of a neural network, and proposes a closed-loop control formulation to automatically detect and fix the errors caused by various attacks or perturbations. We provide a margin-based analysis to explain how this formulation can improve the robustness of a classifier. To speed up the inference, we convert the optimal control problem to Pontryagon's Maximum Principle and solve it via the method of successive approximation. Lastly, we present an error estimation of the proposed framework for neural networks with nonlinear activation functions. We validate the performance of several network architectures against various perturbations. Since the self-healing method does not need *a-priori* information about data perturbations or attacks, it can handle a broad class of unforeseen perturbations. [1]

Keywords: Closed-loop Control, Neural Network Robustness, Optimal Control, Self-Healing, Pontryagin's Maximum Principle

1. Introduction

Despite their success in massive engineering applications, deep neural networks are found to be vulnerable to perturbations of input data. It has been shown that an imperceptible

1. A Pytorch implementation can be found in:`https://github.com/zhuotongchen/Self-Healing-Robust-Neural-Networks-via-Closed-Loop-Control.git`

perturbation of an input image can cause misclassification in a well-trained neural network (Szegedy et al., 2013; Goodfellow et al., 2014). Even though deep neural networks have achieved state-of-the-art performance in many applications, such as computer vision, natural language processing, and recommendation systems, the vulnerability of neural networks has raised security concerns in safety-critical applications.

Many defense methods have been proposed to circumvent this issue, and we summarize the existing methods into two main types: training-based defense which focuses on the classifier itself, and data-based defense that exploits the underlying data information. Arguably, the most representative training-based defense is adversarial training (Madry et al., 2017) based on robust optimization. This method successfully passed all adversarial attack evaluations in Athalye et al. (2018). However, such training-based methods have some limitations. First, adversarial training can be expensive for large-scale applications (Gan et al., 2020). Second, one often requires some information about the type of attacks anticipated, e.g. adversarial training simulates an attack using projected gradient ascent under a chosen norm, thus is adapted to such types of attacks. Data-based defenses, such as reactive defense (Song et al., 2017; Samangouei et al., 2018), are introduced as alternatives to training-based defenses in part to alleviate some of the aforementioned issues. However, there is limited understanding of the working principles behind these methods, and to date, they have not been able to achieve the state-of-the-art performance (Athalye et al., 2018).

To address the aforementioned challenges, it is natural to consider a self-healing process that emulates the mechanisms of a robust biological immune system. In a figurative sense, self-healing properties can be ascribed to systems or processes that, by nature or design, tend to correct any disturbances brought into them. For instance, 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 (Rajapakse and Groudine, 2011). 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 (Tang et al., 2012; Lee et al., 2012; Goyal et al., 2011; Liu et al., 2011; Chien et al., 2012; Keskin et al., 2010; Sadhu et al., 2013; Sun et al., 2014). In the context of machine learning, a self-healing process is expected to fix or mitigate some undesired issues by itself.

In this paper, we realize this proposal via a closed-loop control method. Significantly differing from the attack-and-defense methods, a self-healing process does not need attack/perturbation information, and it focuses on detecting and fixing possible errors by the neural network itself. This allows a neural network to handle many types of attacks and perturbations simultaneously.

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

- **Closed-loop control formulation and margin-based analysis for post-training self-healing.** We consider a closed-loop control formulation to achieve self-healing in the post-training stage, to improve the robustness of a given neural network under a broad class of unforeseen perturbations/attacks. This self-healing formulation has two key components: embedding functions at both input and hidden layers to detect the

possible errors, and a control process to adjust the neurons 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.

- **Fast numerical solver for the control objective function.** The self-healing neural network is implemented via closed-loop control, and this implementation causes computing overhead in the inference. In order to reduce the computing overhead, we solve the Pontryagin's Maximum Principle via the method of successive approximations. This numerical solver allows us to handle both deep and wide neural networks.

- **Theoretical error analysis.** We provide an error analysis of the proposed framework in its most general form by considering nonlinear dynamics with nonlinear embedding manifolds. The theoretical setup aligns with our algorithm implementation without simplification.

- **Empirical validation on several datasets.** On two standard and one challenging datasets, we empirically verify that the proposed closed-loop control implementation of self-healing can consistently improve the robustness of the pre-trained models against various perturbations.

Our preliminary result was reported in (Chen et al., 2021). This extended work includes the following additional contributions: a broader vision of closed-loop control and self-healing methods, the margin-based analysis of the loss function, an accelerated PMP solver, and a more generic error analysis in the nonlinear setting.

2. An Optimal Control-based Self-Healing Neural Network Framework

This section introduces the shared robustness issue in integrated circuits (IC) and in neural networks. We show that the self-healing techniques widely used in IC design can be used to improve the robustness of neural networks due to the theoretical similarities between these two seemingly disconnected domains.

2.1 Self-Healing in IC Design

In this work, we use the term "self-healing" to describe the capability of automatically correcting (possibly after detecting) the possible errors in a neural network. This idea has been well studied in the IC design community to fix the errors caused by nano-scale fabrication process variations in analog, mixed-signal, and digital system design (Tang et al., 2012; Lee et al., 2012; Goyal et al., 2011; Liu et al., 2011; Chien et al., 2012; Keskin et al., 2010; Sadhu et al., 2013; Sun et al., 2014). In practice, it is hard to precisely control the geometric or material parameters in IC fabrication, which causes lots of circuit chips to underperform or even fail to work. To address this issue, two techniques are widely used: yield optimization and self-healing. Yield optimization (Zhang and Styblinski, 2013; Wang et al., 2017; Li et al., 2006; Cui et al., 2020; He and Zhang, 2021) is similar to adversarial training: it chooses the optimal circuit parameters in the design phase to minimize the failure probability assuming that an exact probability density function of the

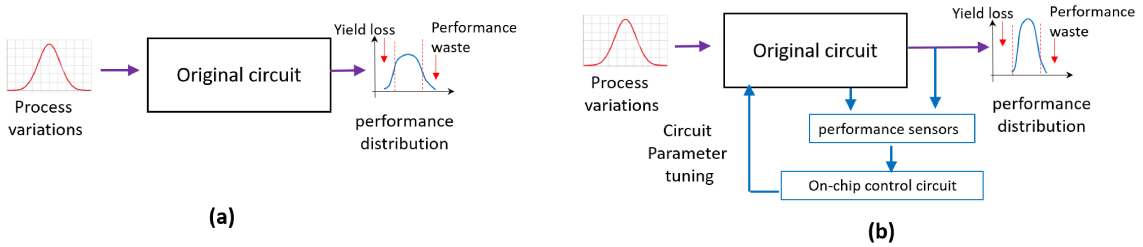


Figure 1: (a) Standard circuit design without self-healing. The result can have significant yield loss and performance waste; (b) Self-healing circuit with on-chip performance monitor and control, resulting higher yield and less performance waste.

process variation is given. Self-healing, on the other hand, intends to fix the possible circuit errors in the post-design phase without knowing the distribution of process variations. We have similar challenges in trustworthy neural network design: it is hard to foresee what types of attacks or perturbations will occur in the practical deployment of a neural network model. Therefore post-training correction can be used to fix many potential errors beyond the capability of adversarial training.

Among various possible self-healing implementations, closed-loop control has achieved great success in practical chip design (Tang et al., 2012; Lee et al., 2012). The key idea is shown in Fig. 1. In Fig. 1 (a), a normal circuit, possibly after yield optimization (which tries to maximize the success rate of a circuit under various uncertainties), may still suffer from significant yield loss or performance waste due to the unpredictability of practical process variations. As shown in Fig. 1 (b), in order to address this issue, some on-chip global or local sensors can be added to monitor critical performance metrics. A control circuit is further added on the chip to tune some circuit parameters (e.g., bias currents, supply voltage, or variable capacitors) to fix the possible errors, such that the output performance distribution is adjusted to center around the desired region with a higher circuit yield and less performance waste.

The same idea can be employed to design self-healing neural networks due to the following similarities between electronic circuits and neural networks:

- Certain types of electronic circuits have similar mathematical formulations to certain types of neural networks. Specifically, an electronic circuit network can be described by an ordinary differential equation (ODE) $d\mathbf{x}/dt = \mathbf{f}(\mathbf{x}, t)$ based on modified nodal analysis (Ho et al., 1975), where the time-varying state variables $\mathbf{x}$ denote nodal voltages and branch currents. Recent studies have clearly shown that certain types of neural networks (such as residual neural networks, and recurrent neural networks) can be seen as a numerical discretization of continuous ODEs (E, 2017; Li et al., 2017; Haber and Ruthotto, 2017; Chen et al., 2018), and the hidden states at layer t can be regarded as a time-domain snapshot of the ODE at time point t.

- Both integrated circuits and neural networks suffer from some uncertainty issues. In IC design, the circuit performance is highly influenced by noise and process variations $\boldsymbol{\epsilon}$,

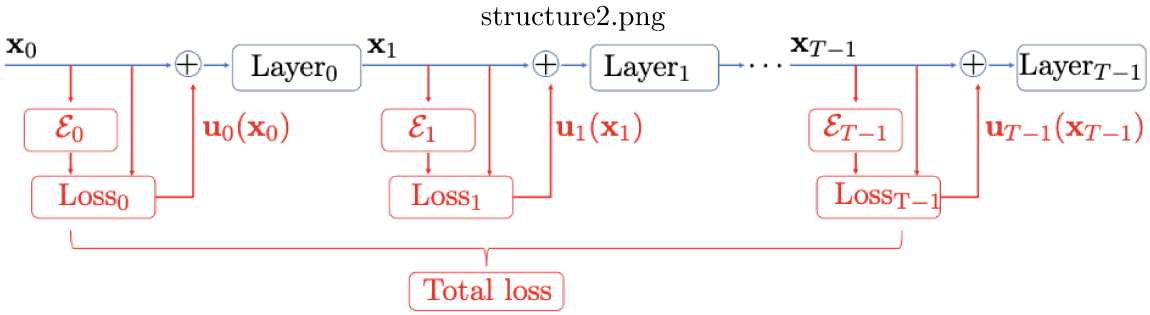


Figure 2: The structures of feed-forward neural network (blue) and the proposed closed-loop control method (red).

resulting in a modified governing ODE $d\mathbf{x}(\boldsymbol{\epsilon})/dt = \mathbf{f}(\mathbf{x}(\boldsymbol{\epsilon}), \boldsymbol{\epsilon}, t)$. In neural network design, the prediction is highly influenced by data corruption and attacks. As a result, robust design and training become important in both domains. This issue has been handled in the design phase via robust or stochastic optimization [e.g., yield optimization in IC design (Antreich et al., 1994; Li et al., 2004; Cui et al., 2020; He and Zhang, 2021) or adversarial training in neural network design (Madry et al., 2017; Zhang et al., 2019; Gowal et al., 2020)] which gets $\boldsymbol{\epsilon}$ involved in the optimization process.

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

This work will implement post-training self-healing via closed-loop control to achieve better robustness of neural networks. Similar to the self-healing circuit design (Lee et al., 2012; Goyal et al., 2011; Liu et al., 2011; Chien et al., 2012; Keskin et al., 2010; Sadhu et al., 2013; Sun et al., 2014), some performance monitors and control blocks can be added to a given T-layer neural network as shown in Fig. 2. Specifically, we consider residual neural networks because they can be regarded as a forward-Euler discretization of a continuous ODE with the t-th layer as a time-domain snapshot at time point t. At every layer, an embedding function $\mathcal{E}_t(\cdot)$ is used to monitor the performance of a hidden layer and generate a loss. The control signal $\mathbf{u}_t(\mathbf{x}_t)$ is computed by optimizing the total loss that 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 to $\mathbf{u}_t$ that is generated based on the state $\mathbf{x}_t$.

Remark 1 *Our proposed neural network architecture in Fig. 2 should not be misunderstood as an open-loop control. In dynamic systems $\mathbf{x}_0$ (input data of a neural network) is an initial condition, and the excitation input signal is $\mathbf{u}_t$ (which is 0 in a 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 $\mathbf{x}_{t+1} = F_t(\mathbf{x}_t + \mathbf{u}_t)$. 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:

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

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.

In order to achieve better robustness via the above self-healing closed-loop control, several fundamental questions should be answered:

- How shall we design the control objective function (1), such that the obtained controls can indeed correct the possible errors and improve model robustness?
- How can we solve the control problem efficiently, such that the extra latency is minimized in the inference?
- What is the working principle and theoretical performance guarantees of the self-healing neural network?

These key questions will be answered through Section 3 to Section 5.

3. Design of Self-Healing via Optimal Control

In this section, we propose a control objective function for self-healing robust neural networks in solving classification problems. With a margin-based analysis, we demonstrate that this control objective function enlarges the classification margin of the decision boundary.

3.1 Towards Better Robustness: Control Loss via Manifold Projection

In general, the control objective function Eq. (1) should have two parts: 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[2] $f(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$, where we assume that all data samples lie in $\mathbb{R}^d$ and there exits 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 Eq. (1) as follows,

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

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" (Fefferman et al., 2016), 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 Eq. (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, the role of control regularization is analysed in Appendix D.

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

$$\mathcal{E}_t(\mathbf{x}_t) := \arg\min_{\mathbf{z}\in\mathcal{M}_t} \frac{1}{2}\|\mathbf{x}_t - \mathbf{z}\|_2^2. \tag{3}$$

The manifold projection can be considered a constrained optimization. Given that $\mathcal{M}_t$ is a compact set, the solution of Eq. (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,

$$\begin{aligned}\mathcal{E}_t^*(\mathcal{M}_t, \cdot) = \arg\min_{\mathcal{E}_t} \frac{1}{N}\sum_{i=1}^{N} \underbrace{\mathrm{CE}(\mathbf{x}_{i,T}, \mathbf{y}_i)}_{\text{model information}} + \underbrace{\|\mathcal{E}_t(\mathcal{M}_t, \mathbf{x}_{i,t}) - \mathbf{x}_{i,t}\|_2^2}_{\text{data information}}, \\ \text{s.t. } \mathbf{x}_{i,t+1} = F_t(\mathbf{x}_{i,t}, \boldsymbol{\theta}_t, \mathbf{u}_{i,t}),\ \mathbf{u}_{i,t} = \mathcal{E}_t(\mathbf{x}_{i,t}) - \mathbf{x}_{i,t},\end{aligned} \tag{4}$$

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 Eq. (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, Eq. (4) can be modified to only optimize the data information (Schmidhuber, 2015).

2. a submersion is a differentiable map between differentiable manifolds whose differential is everywhere surjective

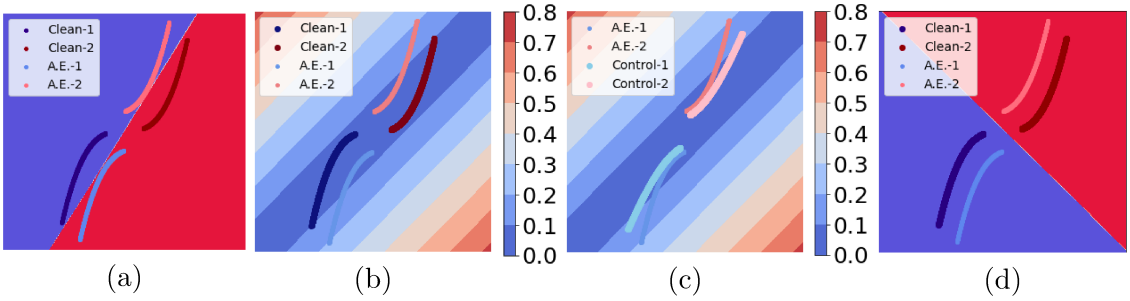


Figure 3: (a): Clean-1 and Clean-2 represent clean data of class 1 and 2, respectively. A.E.-1 and A.E.-2 are their adversarially perturbed counterparts. (b): the reconstruction loss field. (c): the controlled counterpart. (d): Manifold projection modifies decision boundary.

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

$$\min_{\overline{\mathbf{u}}} \mathbb{E}_{(\mathbf{x}_0,\mathbf{y})\sim\mathcal{D}} \sum_{t=0}^{T-1} \|f_t(\mathbf{x}_t+\mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2,$$
$$\text{s.t. } \mathbf{x}_{t+1} = F_t(\mathbf{x}_t, \boldsymbol{\theta}_t, \mathbf{u}_t),\ t = 0, \cdots, T-1. \tag{5}$$

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

3.2 A Margin-based Analysis On the Running Loss

We discuss the effectiveness of the running loss in Eq. (2) by considering the robustness issue in deep learning. To simplify the problem setting, we consider a special case of the control objective function in Eq. (5) where control is only applied at one layer. Specifically, we assume that the control is applied to the input ($T = 1$) and that the applied control is not penalized ($c = 0$). The analysis in a generic t-th layer can be done similarly by seeing $\mathbf{x}_{t-1}$ 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 Eq. (3).

Manifold projection enlarges decision boundary For any given input $\tilde{\mathbf{x}}$, an optimal control process solves the constrained optimization Eq. (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 (Madry et al., 2017). We illustrate these phenomena with a numerical example. Fig. 3 shows a binary classification problem in $\mathbb{R}^2$, where blue and red regions represent the classification predictions (their joint line represents decision boundary of the underlying classifier). In Fig. 3 (a), the given

classifier has accuracies of 100% and 0% on clean data and against adversarial examples respectively. Fig. 3 (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 Fig. 3 (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 Fig. 3 (d).

Remark 2 *In this simplified linear case, the embedding manifold $\mathcal{M}$ is the 1-D linear subspace highlighted as the darkest blue in Fig. 3 (b) (c). Specifically, any data point in this subspace incurs zero reconstruction loss. Therefore, the constrained optimization problem in Eq.* (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.

$$d_{\mathcal{M}}(F(\cdot)) := \frac{1}{2} \inf_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}} \mathcal{R}_{\mathcal{M}}(\mathbf{x}_1, \mathbf{x}_2), \quad \text{s.t. } F(\mathbf{x}_1) \neq F(\mathbf{x}_2). \tag{6}$$

- **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.

$$d_e(F(\cdot)) := \inf_{\mathbf{x} \in \mathcal{D}} \inf_{\boldsymbol{\delta} \in \mathbb{R}^d} \|\boldsymbol{\delta}\|_2, \quad \text{s.t. } F(\mathbf{x}) \neq F(\mathbf{x} + \boldsymbol{\delta}). \tag{7}$$

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 Eq. (6)] is the largest classification margin.
- **Manifold projection margin:** The manifold projection Eq. (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 Eq. (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 $d_{\mathcal{M}}(F(\cdot))$.

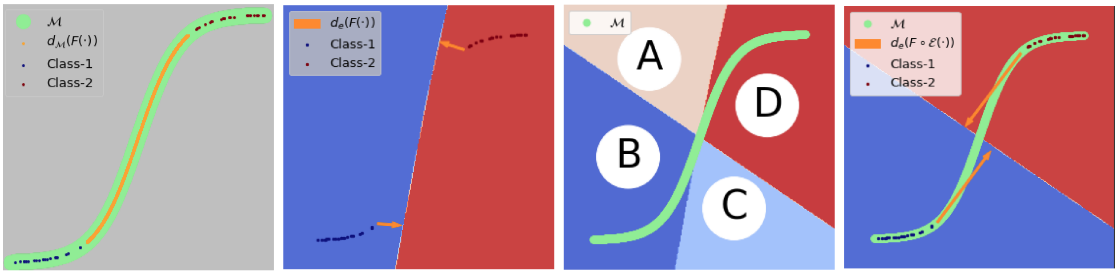
Proposition 3 *Let $\mathcal{M} \subset \mathbb{R}^d$ be a r-dimensional ($r \leq d$) linear subspace that contains the ground-truth manifold $\mathcal{M}^*$, such that $\mathcal{M}^* \subset \mathcal{M}$, $F(\cdot)$ a linear classifier with random decision boundary, then $\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}$.*

The detailed proof is shown in Appendix A. The margin-based analysis explains the design choice of the running loss in Eq. (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. Fig. 4 (a) shows a binary classification dataset embedded in a 1-dimensional manifold ($\mathcal{M}$ is shown as green curve). Given a classifier $F(\cdot)$, the manifold margin $d_{\mathcal{M}}(F(\cdot))$ (orange curve shows $2 \cdot d_{\mathcal{M}}(F(\cdot))$) is shown as the shortest distance that an instance of one class transforms to another. The underlying classifier results in a small Euclidean margin, as shown in Fig. 4 (b). In Fig. 4 (c), subsets-A and B are predictions of class-1, subsets-C and D are predictions of class-2. The manifold projection $\mathcal{E}(\cdot)$ projects subsets-A and D onto the top portion of $\mathcal{M}$, subsets B and C onto the lower portion of $\mathcal{M}$. The decision boundary of classifier and manifold projection form four partitions of $\mathbb{R}^2$. For the composed classifier $F \circ \mathcal{E}(\cdot)$, any samples in regions A and D are predicted as class-2, and samples from regions B and C are predicted as class-1. As a result, Fig. 4 (d) shows the decision boundary of $F \circ \mathcal{E}(\cdot)$, the manifold projection margin (shown in orange) is significantly improved than the Euclidean margin.

4. An Optimal Control Solver for Self-Healing

In this section, we present a general optimal control method to solve the proposed objective function in Eq. (5). A more efficient method is proposed to reduce the inference overhead caused by generating the controls.



(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 4: (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 $\mathcal{M}$, regions C and D are projected onto lower portion of $\mathcal{M}$. (d): the manifold projection margin.

4.1 Control Solver Based on the Pontryagin's Maximum Principle

The proposed self-healing neural network can be achieved by solving the dynamical programming principle (Bellman, 1952). However, this has exponential complexity w.r.t. the state dimension. To overcome the computational challenge, we first describe a general solver for the optimal control problem in Eq. (1) based on Pontryagin's Maximum Principle (Pontryagin, 1987).

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

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

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

$$\mathbf{x}_{t+1}^* = \nabla_p H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*), \qquad (\mathbf{x}_0^*, \mathbf{y}) \sim \mathcal{D}, \tag{8}$$

$$\mathbf{p}_t^* = \nabla_x H(t, \mathbf{x}_t^*, \mathbf{p}_{t+1}^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*), \qquad \mathbf{p}_T^* = \mathbf{0}, \tag{9}$$

plus a maximum condition of the Hamiltonian.

$$H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t^*) \geq H(t, \mathbf{x}_t^*, \mathbf{p}_t^*, \boldsymbol{\theta}_t, \mathbf{u}_t), \ \forall \mathbf{u} \in \mathbb{R}^{d'} \ \text{ and } \ \forall t \in \mathcal{T}. \tag{10}$$

To obtain a numerical solution, one can consider iterating through the forward dynamic Eq. (4.1) to obtain all states $\{\mathbf{x}_t\}_{t=0}^{T-1}$, the backward dynamic Eq. (9) to compute the adjoint states $\{\mathbf{p}_t\}_{t=0}^{T-1}$, and updating the Hamiltonian Eq. (10) with current states and adjoint states via gradient ascent (Chen et al., 2021). This iterative process is continued until convergence.

4.2 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. When the closed-loop control module is deployed for inference, the original forward propagation is now replaced by iterating through the Hamiltonian dynamics. For each input data, solving the optimal control problems requires us to propagate through both forward Eq. (8) and backward adjoint Eq. (9) dynamics and to maximize the Hamiltonian Eq. (10) at all layers. When maximizing the Hamiltonian n times, running the Hamiltonian dynamics approximately increase the time complexity by a factor of n with respect to the standard inference. The computational overhead prevents deploying the closed-loop control module in real-world applications.

A Faster PMP Solver. To address this issue, we consider the method of successive approximation (Chernousko and Lyubushin, 1982) from the optimal condition of the PMP. For a given input data sample, Eq. (8) and (9) generate the state variables and adjoint states respectively for the current controls $\{\mathbf{u}_t\}_{t=0}^{T-1}$. The optimal condition of the objective function in Eq. (1) is achieved via maximizing all Hamiltonians in Eq. (10). Instead of iterating through all three Hamiltonian dynamics for a single update on the control solutions, we can consider optimizing the t^{th} Hamiltonian locally for all $t \in [0, \cdots, T-1]$ with the current state $\mathbf{x}_t$ and adjoint state $\mathbf{p}_{t+1}$. This allows the control solution $\mathbf{u}_t$ to be updated multiple times within one complete iteration. Once a locally optimal control $\mathbf{u}_t^*$ is achieved by maximizing $H(t, \mathbf{x}_t, \mathbf{p}_{t+1}, \boldsymbol{\theta}_t, \mathbf{u}_t)$ w.r.t. $\mathbf{u}_t$, the adjoint state $\mathbf{p}_{t+1}$ is backpropagated to $\mathbf{p}_t$ via the adjoint dynamic in Eq. (9) followed by maximizing $H(t-1, \mathbf{x}_{t-1}, \mathbf{p}_t, \boldsymbol{\theta}_{t-1}, \mathbf{u}_{t-1})$. Under this setting, running the Hamiltonian dynamics (8), (9) and (10), n times can be decomposed into $maxItr$ full iterations and $InnerItr$ local updates. Here, $maxItr$ can be significantly smaller than n since the locally optimal control solutions via $InnerItr$ updates can speed up the overall convergence. Instead of iterating the full Hamiltonian dynamics n times, the proposed fast implementation iterates $maxItr$ full Hamiltonian dynamics and $InnerItr$ local updates.

The detailed implementation is presented in Algorithm 1. Here we summarize this efficient implementation.

1. To begin with, we initialize all controls with the greedy solution, $\mathbf{u}_t = \mathcal{E}_t(\mathbf{x}_t) - \mathbf{x}_t$, by setting the control regularization $c = 0$. This improves the convergence of the Hamiltonian dynamics.

2. We forward propagate the input data via Eq. (8) to obtain all hidden states.

3. Since there is no terminal loss, the initial condition of the adjoint state is $\mathbf{p}_T = \mathbf{0}$. We backpropagate the adjoint states and maximize the Hamiltonian at each layer as follows:

   (a) We compute the adjoint state $\mathbf{p}_t$ from the adjoint dynamics Eq. (9),

   (b) Instead of updating control $\mathbf{u}_t$ once via maximizing the Hamiltonian Eq. (10), we perform *multiple updates* ($InnerItr$ iterations) on control $\mathbf{u}_t$ to achieve the optimal solution $\mathbf{u}_t^*$ that satisfies the maximization condition (Notice that any optimization algorithm can be applied).

4. The backpropagation terminates when it reaches layer $t = 0$. This process repeats for a maximum number of iterations ($maxItr$ iterations).

Algorithm 1: The 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: Output state $\mathbf{x}_T$.

Initialize controls $\{\mathbf{u}_t\}_{t=0}^{T-1}$ with the greedy solution ;
for $i = 0$ *to* maxItr **do**
  $\mathbf{x}_0^i = \mathbf{x}_0 + \mathbf{u}_0^i$; // The controlled initial condition
  for $t = 0$ *to* $T-1$ **do**
    $\mathbf{x}_{t+1}^i = F_t(\mathbf{x}_t^i + \mathbf{u}_t^i)$; // Controlled forward propagation Eq. (8)
  end for
  $\mathbf{p}_T^i = \mathbf{0}$; // The terminal condition of the adjoint state is set to 0
  for $t = T-1$ *to* 0 **do**
    for $\tau = 0$ *to* InnerItr **do**
      $H(t, \mathbf{x}_t^i, \mathbf{p}_{t+1}^i, \boldsymbol{\theta}_t, \mathbf{u}_t^{i,\tau}) = \mathbf{p}_{t+1}^i \cdot F_t(\mathbf{x}_t^i, \boldsymbol{\theta}_t, \mathbf{u}_t^{i,\tau}) - \mathcal{L}(\mathbf{x}_t^i, \mathbf{u}_t^{i,\tau}, \mathcal{E}_t(\mathbf{x}_t^i))$; // Compute Hamiltonian
      $\mathbf{u}_t^{i,\tau+1} = \mathbf{u}_t^{i,\tau} + \mathrm{lr} \cdot \nabla_{\mathbf{u}} H(t, \mathbf{x}_t^i, \mathbf{p}_{t+1}^i, \boldsymbol{\theta}_t, \mathbf{u}_t^{i,\tau})$; // Maximize Hamiltonian w.r.t. control $\mathbf{u}_t$
    end for
    $\mathbf{p}_t^i = \mathbf{p}_{t+1}^i \cdot \nabla_{\mathbf{x}} F(\mathbf{x}_t^i, \boldsymbol{\theta}_t, \mathbf{u}_t^i) - \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}_t^i, \mathbf{u}_t^i, \mathcal{E}_t(\mathbf{x}_t^i))$; // Backward propagation Eq. (9)
  end for
end for

5. Theoretical Error 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 perturbed states originating from a possible attacked or corrupted data sample $\mathbf{x}_{\epsilon,0} = \mathbf{x}_0 + \mathbf{z}$. In our proposed self-healing neural network, the controlled state becomes $\overline{\mathbf{x}}_{\epsilon,t} = \mathbf{x}_{\epsilon,t} + \mathbf{u}_t$. We ask this question: how large is $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|$, i.e., the distance between $\mathbf{x}_t$ and $\overline{\mathbf{x}}_{\epsilon,t}$?

We consider a general deep neural network $F = F_T \circ F_{T-1} \circ \cdots \circ F_0$, where each nonlinear transformation $F_t(\cdot)$ is of class $\mathcal{C}^2$, and each embedding manifold can be described by a $\mathcal{C}^2$ submersion $f(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d-r}$, such that $\mathcal{M} = f^{-1}(\mathbf{0})$. Given an unperturbed state trajectory $\{\mathbf{x}_t \in \mathcal{M}_t\}_{t=0}^{T-1}$, we denote $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ as the tangent space of $\mathcal{M}_t$ at $\mathbf{x}_t$.

This theoretical result is an extension of the linear closed-control setting in our preliminary work (Chen et al., 2021) where an error estimation in the linear setting is derived. We provide the error estimation between $\overline{\mathbf{x}}_{\epsilon,t}$ and $\mathbf{x}_t$ in the linear and nonlinear cases in Section 5.1 and Section 5.2 respectively.

Outline of the proof. Our goal is to derive an error estimation on the clean state $\mathbf{x}_t$ and perturbed state corrected with controls $\overline{\mathbf{x}}_t$. We achieve this analysis with three steps:

1. **Linear control system:** Derivation for the error estimation of the linear system with linear control. This is presented in Section 5.1 as Theorem 4.

2. **Linearization error:** We linearize the given closed-loop controlled dynamical system. This linearization leads to two error sources. We derive an upper bound on linearizing the embedding manifold and nonlinear dynamical system in Appendix C.1 and C.2 respectively.

3. Finally, the error estimation is presented in Theorem 5 from Section 5.2,

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

5.1 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 Eq. (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, $\mathbf{z} = \mathbf{z}^{\|} \oplus \mathbf{z}^{\perp}$. Here $\mathbf{z}^{\|} \in \mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$ 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 (Chen et al., 2021) provides an upper bound of $\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2$.

Theorem 4 *For $t \geq 1$, we have an error estimation for the linear system*

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

where $\gamma_t := \max_{s\leq t}\big(1+\kappa(\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)^2\big)\|\mathbf{I} - (\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)^T(\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)\|_2$, $\kappa(\boldsymbol{\theta})$ 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.

The detailed derivation is presented in Appendix B. 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.

5.2 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 5 *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

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

The detailed proof is provided in Appendix C. From Theorem 5, 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 6 *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.*

6. 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 6.1 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 6.2 and 6.3 evaluate our method on CIFAR-100 and Tiny-ImageNet datasets, which empirically verify the effectiveness and generalizability of the self-healing machinery.

6.1 Experiments On CIFAR-10 Dataset

We evaluate all controlled models under an "oblivious attack" setting [3]. 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 (Zhang et al., 2019)). 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.

- **Robustness evaluations.** We evaluate the performance of all models with clean testing data (**None**), and auto-attack (**AA**) (Croce and Hein, 2020b) that is measured by ℓ_∞, ℓ_2 and ℓ_1 norms. Auto-attack that is an ensemble of two gradient-based auto-PGD attacks (Croce and Hein, 2020b), fast adaptive boundary attack (Croce and Hein, 2020a) and a black-box square attack (Andriushchenko et al., 2020).

- **Embedding functions.** We choose the fully convolutional networks (FCN) (Long et al., 2015) 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 Eq. (4), where both model and data information are used.

3. This consideration is general, e.g. Liao et al. (2018) has adopted this setting in the previous NIPS competition on defense against adversarial attacks.

Table 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**

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

As shown in Table 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.

6.2 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

Table 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**

pre-trained models when those models are trained via momentum SGD and adversarial training (TRADES (Zhang et al., 2019)).

- **Other settings.** The embedding functions and PMP settings follow the same.

In Table 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 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.

6.3 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 (Zhang et al., 2019)) as testing benchmarks.

- **Embedding functions.** We choose SegNet (Badrinarayanan et al., 2017) as an input embedding function, and a 2-layer auto-encoder as an embedding function for the hidden states. The training objective function of the t^{th} embedding function follows Eq. (4), where both model and data information are used.

- **PMP hyper-parameters setting.** The PMP setting follows the same.

Table 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**

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, 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.

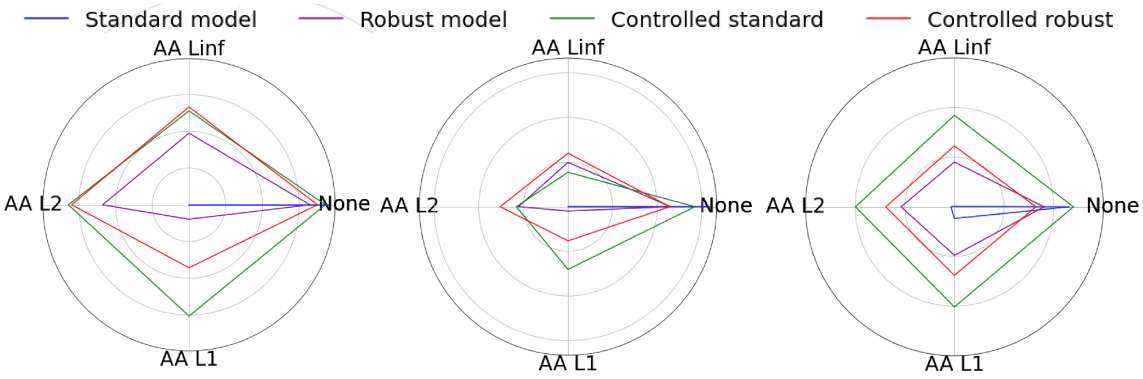
6.4 Summary On Numerical Experiments

Fig. 5 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_∞). 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.

6.5 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 (Liu et al., 2016). For testing, we use the VOC 2007 test with 4952 test images and 20 classes (Everingham et al., 2010). 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) (Long et al., 2015) as an input embedding function and a 2-layer auto-encoder as an embedding function for the hidden states. The PMP hyper-parameter settings are the same as in pre-



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

Figure 5: (a), (b) and (c) are radar plots that summarize RN-18 in Table 1, WRN-76-10 in Table 2, and EfficientNet-b0 in Table 3 respectively.

Table 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**

vious 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 4, despite the high 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.

6.6 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. 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 6.5. Fig. 6 (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 Fig. 6 (middle), an adversarial attack drives the clean image out of the embedding manifold (middle plot). Notice how the adversarial

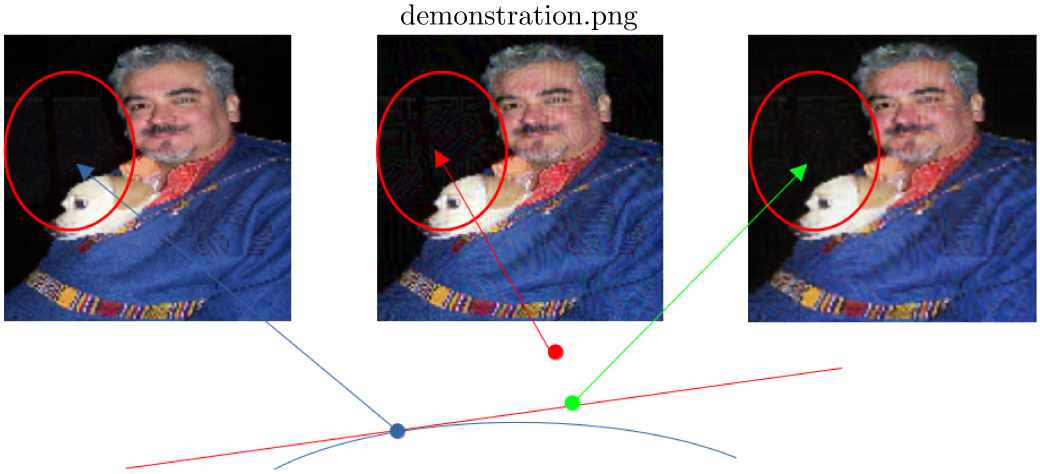


Figure 6: Demonstration of manifold-based control method on VOC input data. Left: a clean image that belongs to classes person and dog. Middle: a perturbed image that is predicted as cow. Right: controlled image. The main difference between those three images is highlighted in the red circle. The red 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.

perturbation changes the texture of the image background as highlighted in the red circle. In Fig. 6 (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 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 high-dimensional data generally lies in a low-dimensional manifold (Fefferman et al., 2016), 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 a 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 Fig. 7, 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. Fig. 7 (a) shows reconstruction loss versus controlled model performance w.r.t. 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 linear embedding subspace increases, the reconstruction loss of 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. Fig. 7 (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 Fig. 7 (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 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, 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 $\mathbb{E}\left[\frac{d_e(F(\cdot))}{d_{\mathcal{M}}(F(\cdot))}\right] \leq \sqrt{\frac{r}{d}}$. 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 w.r.t. 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 Fig. 7 (i) and (g), as the dimension r of the embedding subspace increases, the model robustness against both ℓ_∞ and ℓ_2 perturbations decreases. This validates Proposition 3 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 Fig. 7 (h).

Analysis of numerical approximation errors. In the proposed closed-loop control method, both numerical errors from estimating the ground-truth manifold and solving the PMP can affect the final result.

When a manifold learning setting is chosen, the set of embedding manifolds that can be generated from this method may not include the ground-truth manifold. As shown in

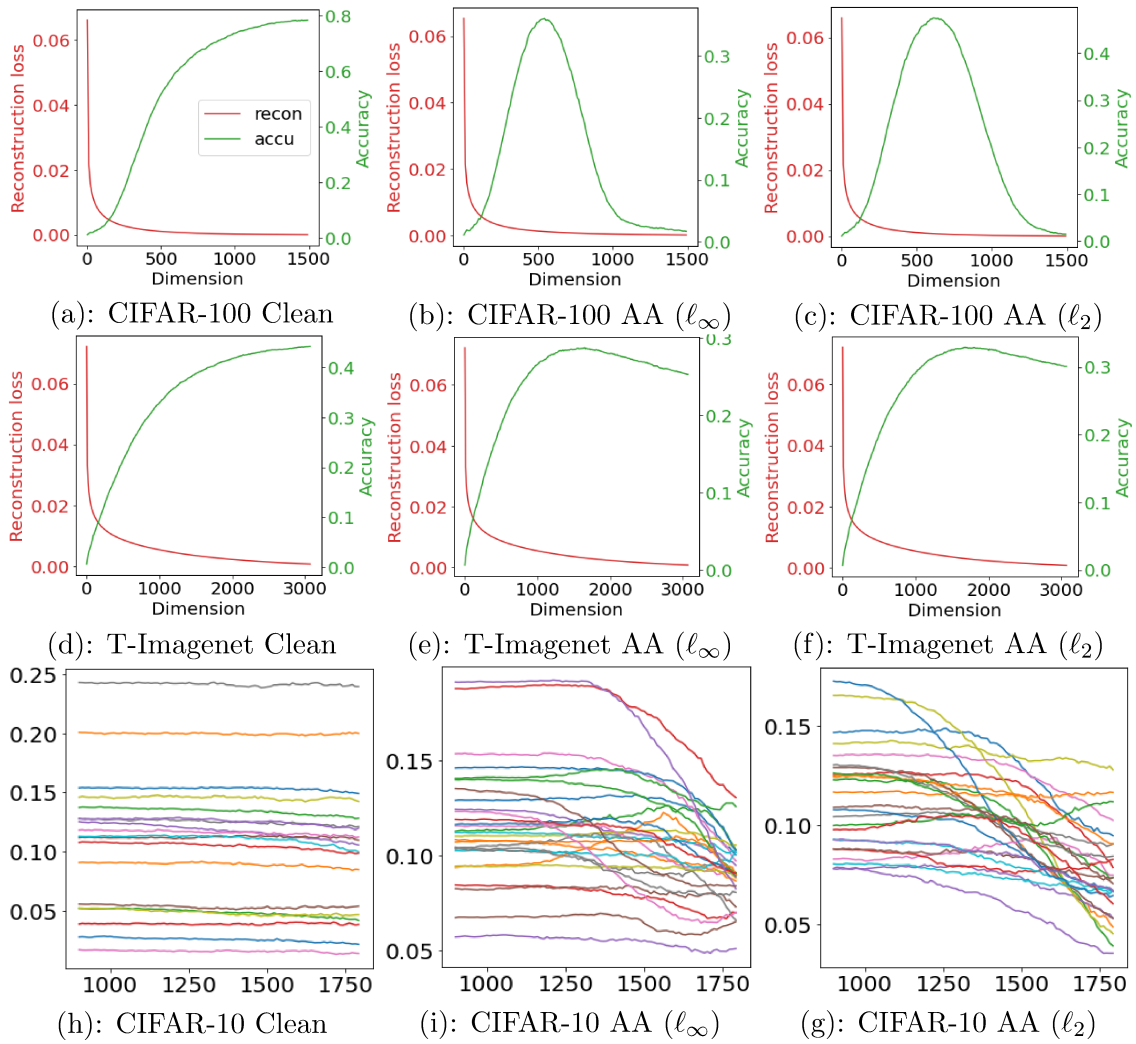


Figure 7: (a) plots reconstruction loss (red y-axis) versus controlled model accuracy (green y-axis) on CIFAR-100 clean test set. The results on ℓ_∞ and ℓ_2 auto-attack perturbed data are shown in (b) and (c) respectively. (d), (e) and (f) conduct the same experiment on Tiny-Imagenet dataset. (h), (i) and (g) study the performance of a linear classifier with a random decision boundary on clean, ℓ_∞ and ℓ_2 auto-attack perturbed data respectively.

Fig. 7, if a chosen linear embedding subspace has a low dimension and it cannot possibly contain the ground-truth manifold, both the controlled model performance and robustness are poor. Here, we provide more intuition about the effect of this approximation error. We build a synthetic binary classification dataset as used in Section 3.2. In Fig. 8 (a), we construct a linear embedding subspace from the noisy data and consider the set where the reconstruction loss is less than 0.2 as the linear embedding subspace. This estimation does not represent the underlying clean data accurately since most clean data is located outside

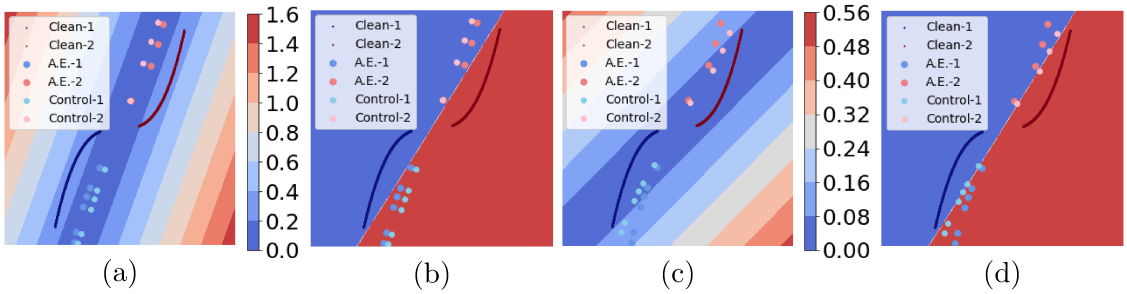


Figure 8: (a) and (b) show the numerical error caused by poorly estimated embedding subspace. (c) and (d) show the effect of solving the PMP. Clean, A.E. and Control represent clean data, adversarially perturbed data and controlled data respectively.

of this subspace. In Fig. 8 (b), since the estimated embedding subspace and the underlying clean data are situated on opposite sides of the decision boundary, the controlled data is still located on the wrong side of the decision boundary and the applied controls are ineffective.

The second numerical approximation error is from solving Pontryagon's Maximum Principle. In general, the objective function in Eq. (5) is highly non-convex due to both nonlinear dynamical systems and complex embedding functions (e.g. auto-encoder). Therefore, there is no guarantee that the solved controls will be the optimal control solution. Given an embedding subspace that accurately encodes the underlying clean data shown in Fig. 8 (c), when the optimization process is not properly solved, the resulting controls are ineffective as shown in Fig. 8 (d).

7. Discussions

7.1 Limitation of the proposed self-healing framework

From both practical and theoretical perspectives, we discuss the limitations of the current work. Those discussions provide insights into the current framework and motivate future research in this direction.

The accuracy of embedding manifolds affects the control performance. As shown in Sec. 3, the objective function of the proposed self-healing framework minimizes the distance between a given state trajectory and the embedding manifolds. The role of embedding manifolds encodes the geometric information of how the clean data behaves in the pre-trained deep neural network. Therefore, an important question is how accurate those embedding manifolds encode the state trajectories. In a case where the embedding manifolds do not precisely resemble the data structures, the running loss in Eq. (2) that measures the distance between perturbed data and the embedding manifold does not have the true information of this applied perturbation. Then the applied controls might lead to wrong predictions, as shown in Section 6.6.

In addition to the precision of embedding manifolds, recall the definition of embedding function in Eq.(3), it searches for the closest counterpart of a given data on the embedding

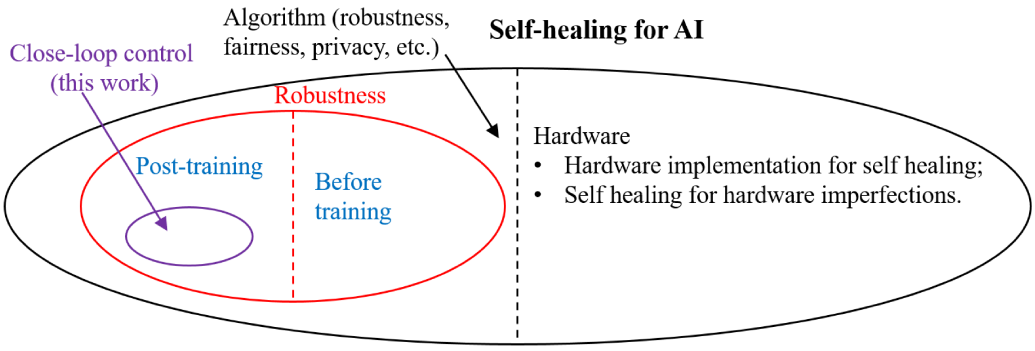


Figure 9: A broader scope of self-healing for neural networks.

manifold. It implies that if data belongs to the embedding manifold, its outcome from the embedding function should stay the same. This property is guaranteed by well-studied linear projection. Given a linear projection operator $\mathbf{P}$, we have $\mathbf{P} \circ \mathbf{Px} = \mathbf{Px}$. However, in the nonlinear case, the shortest path projection $\mathcal{E}(\mathbf{x})$ defined in Eq. (3) does not necessarily hold the projection property. The lack of projection property adds more challenge to measuring the running loss in Eq. (2).

The role of control regularization is unclear. The second limitation is to understanding the role of applying control regularization. As in the conventional optimal control problems, regularizing the applied controls has practical meaning, such as limiting the amount of energy consumption. In the current framework, we have observed that regularizing the applied controls can alleviate the issue of inaccurate embedding manifold, in which case the controls only slightly adjust the perturbed state trajectory. This observation is not theoretically justified in this work, and we will continue to understand this in future work.

7.2 A Broader Scope of Self-Healing

This paper has proposed a self-healing framework implemented with closed-loop control to improve the robustness of given neural network. The control signals are generated and injected into all neurons. Based on this work, many more topics can be investigated in the future. Below, we point out some possible directions.

Extension of the Proposed Framework. An immediate extension of this work is to consider the closed-loop control applied to model parameters (instead of neurons). In order to reduce the control complexity, we may also deploy local performance monitoring and local control instead of monitoring and controlling all neurons. Another more fundamental question is: can we achieve self-healing without using closed-loop control? In (Wang et al., 2021a,b), self-healing is achieved by mimicking the immune system of a human body and without using closed-loop control.

Beyond Post-Training Self-Healing. This work focuses on realizing self-healing after a neural network has already been trained. It may be possible to achieve self-healing in other development stages of a neural network, such as in training and in data acquisition and preparation.

Beyond Robustness. The key idea of self-healing is to automatically fix the possible errors or weakness of a neural network, with or without a performance monitor. This idea may be extended to address other fundamental issues in AI, such as AI fairness, where machine learning models perform unequally against minority subpopulations (Amodei et al., 2016). Specifically, one can construct fair embedding manifolds that exclude the sensitive attributes. This can be done by modifying the objective function of generating embedding manifolds in (4) by adding fairness constraints, such as equal opportunity (Hardt et al., 2016).

Self-healing at the Hardware Level/Computing Platforms. The self-healing perspective brings in many opportunities and challenges at the hardware level. On one hand, the proposed self-healing can cause extra hardware cost in the inference. Therefore, it is important to investigate hardware-efficient self-healing mechanisms, which can provide self-healing capability with minimal hardware overhead. On the other hand, many imperfections in AI hardware may also be addressed via self-healing. Examples include process variations in AI ASIC chip design and software/hardware errors in distributed AI platforms.

Our vision is visualized in Fig. 9. This work is a proof-of-concept demonstration of self-healing for AI robustness, and many more research problems need to be investigated in the future.

8. Conclusion

This paper has improved the robustness of neural networks from a new self-healing perspective. By formulating the problem as a closed-loop control, we show that it is possible for a neural network to automatically detect and fix the possible 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. 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. As pointed out in Section 7, this self-healing method may be extended to investigate other fundamental issues (such as fairness and hardware reliability) of neural networks in the future.

Acknowledgement

Zhuotong Chen and Zheng Zhang are supported by NSF Grant # 2107321 and DOE Grant # DE-SC0021323. Qianxiao Li is supported by the National Research Foundation of Singapore, under the NRF Fellowship (NRF-NRFF13-2021-0005).

Appendix A. Manifold Projection On Classifier Margin

Proposition 7 *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 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 Eq. (6) is equivalent to $d_e(F \circ \mathcal{E}(\cdot))$ in Eq. (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 Eq. (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,

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

■

Appendix B. 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 14, the kernel of $f_t'(\mathbf{x}_t)$ is equivalent to $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$, and the orthogonal projection onto $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ (Eq. (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 Eq. (2), the linear setting uses projection onto a tangent space rather than a nonlinear embedding manifold.

$$J(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2}\|\mathbf{Q}_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2, \tag{11}$$

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 Eq. (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 (Eq. (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 $\mathbf{u}_t^P$ (Eq. (18)) as

$$\mathbf{u}_t^P = -(c\cdot\mathbf{I} + \mathbf{Q}_t^T\mathbf{Q}_t)^{-1}\mathbf{Q}_t^T\mathbf{Q}_t\mathbf{x}_t = -\mathbf{K}_t\mathbf{x}_t, \tag{12}$$

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,

$$\begin{aligned}\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\overline{\mathbf{q}}_{\epsilon,t} + \mathbf{u}_t - \mathbf{x}_t), \\ &= \boldsymbol{\theta}_t(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{K}_t\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t). \end{aligned} \tag{13}$$

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 Eq. (13) can be further shown by adding a $\mathbf{0}$ term of $(\boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t)$

$$\begin{aligned}\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)\overline{\mathbf{q}}_{\epsilon,t} - \boldsymbol{\theta}_t\mathbf{x}_t + \boldsymbol{\theta}_t\mathbf{K}_t\mathbf{x}_t, \\ &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t). \end{aligned} \tag{14}$$

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

Lemma 8 *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 9 *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 10 *Define for* $0 \le s \le t$,

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

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*

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

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

$$\begin{aligned}
\overline{\mathbf{q}}_{\epsilon,1} - \mathbf{x}_1 &= \boldsymbol{\theta}_0(\mathbf{I} - \mathbf{K}_0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \\
&= \boldsymbol{\theta}_0(\alpha \cdot \mathbf{I} + (1-\alpha)\cdot \mathbf{P}_0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), && \text{Lemma 8,} \\
&= \boldsymbol{\theta}_0\mathbf{G}_0^0(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0).
\end{aligned}$$

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 $(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t)$,

$$\begin{aligned}
\overline{\mathbf{q}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t), \\
&= \boldsymbol{\theta}_t(\alpha \cdot \mathbf{I} + (1-\alpha)\cdot \mathbf{P}_t)(\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t), && \text{Lemma 8,} \\
&= \boldsymbol{\theta}_t\mathbf{G}_t^0(\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1})\mathbf{G}_t^1(\boldsymbol{\theta}_{t-2}\boldsymbol{\theta}_{t-3}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\mathbf{G}_{t-2}^{t-2}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0), \\
&= (\boldsymbol{\theta}_t\boldsymbol{\theta}_{t-1}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_t^t\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)(\overline{\mathbf{q}}_{\epsilon,0} - \mathbf{x}_0).
\end{aligned}$$

■

Lemma 11 *For* $t \ge 1$,

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

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 9, $range(\mathbf{P}_t^t) = range(\mathbf{P}_s^s) = Z_{\|}^0$. 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 12 *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,\\
&\le \|\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,\\
&\le \|\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,\\
&\le \|[\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,\\
&\le \|[\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,\\
&\le \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 13 *Let $t\ge 1$. Then for each $s=0,1,\cdots,t$, we have*

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

where

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

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

Theorem 4 *For $t\ge 1$, we have an error estimation for the linear system*

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

where $\gamma_t:=\max_{s\le t}\big(1+\kappa(\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)^2\big)\|\mathbf{I}-(\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)^T(\boldsymbol{\theta}_{s-1}\cdots\boldsymbol{\theta}_0)\|_2$, $\kappa(\boldsymbol{\theta})$ 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 10,

$$\begin{aligned}\|\overline{\mathbf{q}}_{\epsilon,t} - \mathbf{x}_t\|_2^2 &= \|(\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0)(\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)\mathbf{z}\|_2^2, \\ &\leq \|\boldsymbol{\theta}_{t-1}\boldsymbol{\theta}_{t-2}\cdots\boldsymbol{\theta}_0\|_2^2 \cdot \|(\mathbf{G}_{t-1}^{t-1}\cdots\mathbf{G}_0^0)\mathbf{z}\|_2^2, \end{aligned} \tag{15}$$

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 11,

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

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,

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

Using Corollary 13, we have

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

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

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

Thus, we have

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

Recall the error estimation in Eq. (15),

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

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

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

Thus,

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

■

Appendix C. 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:

- Appendix C.1 considers two solutions of the running loss Eq. (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.
- Appendix C.2 provides an $\mathcal{O}(\epsilon^2)$ solution for the linearization error (defined later).
- Finally, Appendix C.3 derives an upper bound for the total error $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|_2$.

C.1 Analysis On Nonlinear Manifold Projection

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

Proposition 14 *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 14 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,

$$\mathbf{P}_{\mathbf{x}} := \mathbf{I} - f'(\mathbf{x})^T f'(\mathbf{x}). \tag{16}$$

In general cases, when $f'(\mathbf{x})$ does not consist of an orthonormal basis, the orthogonal projection in Eq. (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 Eq. (2),

$$\mathbf{u}_{\mathbf{x}}^P(\mathbf{x}_\epsilon) = \arg\min_{\mathbf{u}\in\mathbb{R}^d} \frac{1}{2}\cdot\|f'(\mathbf{x})(\mathbf{x}_\epsilon+\mathbf{u})\|_2^2 + \frac{c}{2}\cdot\|\mathbf{u}\|_2^2. \tag{17}$$

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

$$\mathbf{u}_{\mathbf{x}}^P(\mathbf{x}_\epsilon) = -(c\cdot\mathbf{I}+\mathbf{Q}_{\mathbf{x}})^{-1}\mathbf{Q}_{\mathbf{x}}\mathbf{x}_\epsilon = -(c\cdot\mathbf{I}+f'(\mathbf{x})^T f'(\mathbf{x}))^{-1}f'(\mathbf{x})^T f'(\mathbf{x})\mathbf{x}_\epsilon. \tag{18}$$

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 $\sup_{\mathbf{x}\in\mathbb{R}^d}\|f''(\mathbf{x})\|_* \le \sigma$. 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 Eq. (2),

$$\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon) = \arg\min_{\mathbf{u}\in\mathbb{R}^d} \frac{1}{2}\cdot\|f(\mathbf{x}_\epsilon+\mathbf{u})\|_2^2 + \frac{c}{2}\cdot\|\mathbf{u}\|_2^2. \tag{19}$$

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 15 *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 $0\le\epsilon\le 1$. 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,*

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

Proof Recall the definition of regularized manifold projection in Eq. (19), the optimal solution $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$ admits a exact solution by setting the gradient of Eq. (19) to $\mathbf{0}$,

$$\nabla_{\mathbf{u}}\left(\frac{1}{2}\cdot\|f(\mathbf{x}_\epsilon+\mathbf{u})\|_2^2 + \frac{c}{2}\cdot\|\mathbf{u}\|_2^2\right) = \Big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\Big)^T\Big(f(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\Big) + c\cdot\mathbf{u}. \tag{20}$$

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 $\mathbf{x}\in\mathcal{M}$,

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

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

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

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

$$\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon) = -\Big(f'(\mathbf{x})^T f'(\mathbf{x}) + c\mathbf{I}\Big)^{-1}\Big(\epsilon f'(\mathbf{x})^T f'(\mathbf{x})\mathbf{v} + \mathbf{E}_1 + \mathbf{E}_2\Big),$$

where

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

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 Eq. (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 $\mathbf{u}^P_{\mathbf{x}}(\mathbf{x}_\epsilon)$,

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

Let us simplify the above inequality.

- For any non-negative c,

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

- Recall the gradient of the running loss (Eq. (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 $\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)$,

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

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

$$\begin{aligned}
\|\mathbf{u}^{\mathcal{M}}(\mathbf{x}_\epsilon)\| &\leq \Big\|\bigg(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p}) + c\mathbf{I}\bigg)^{-1}\Big\|_2 \cdot \Big\|\bigg(\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p})\bigg)\Big\|_2(\epsilon), \\
&\leq \Big\|\big(f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})\big)^T f'(\mathbf{p})\Big\|_2^2 \cdot (\epsilon), \\
&\leq \|f'(\mathbf{x}+\epsilon\mathbf{v}+\mathbf{u})^T\|_2^2 \cdot \|f'(\mathbf{p})\|_2^2 \cdot (\epsilon), \\
&\leq \epsilon. \qquad (21)
\end{aligned}$$

- 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,

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

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

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.

C.2 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 14, 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 $\mathcal{T}_{\mathbf{x}_t}\mathcal{M}_t$ (Eq. (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 Eq. (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 $\mathbf{u}_{\mathbf{x}_t}^P(\mathbf{x}_\epsilon)$ (Eq. (18)) is

$$\mathbf{u}_{\mathbf{x}_t}^P(\mathbf{x}_\epsilon) = -(c\cdot\mathbf{I} + f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t))^{-1} f_t'(\mathbf{x}_t)^T f_t'(\mathbf{x}_t)\mathbf{x}_\epsilon = -\mathbf{K}_{\mathbf{x}_t}\mathbf{x}_\epsilon, \tag{22}$$

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 Eq. (2) equipped with an embedding manifold $\mathcal{M}_t$ is defined in Eq. (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 Eq. (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,

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

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 $\mathbf{u}^P_{\mathbf{x}_t}(\overline{\mathbf{x}}_{\epsilon,t})$ in Eq. (22), $\boldsymbol{\theta}_t(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^P_{\mathbf{x}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t) = \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t)$,

$$\begin{aligned}
\overline{\mathbf{x}}_{\epsilon,t+1} - \mathbf{x}_{t+1} &= \boldsymbol{\theta}_t(\mathbf{I} - \mathbf{K}_{\mathbf{x}_t})(\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t) + \boldsymbol{\theta}_t(\mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{u}^P_{\mathbf{x}_t}(\overline{\mathbf{x}}_{\epsilon,t})) \\
&\quad + \frac{1}{2}F_t''(\mathbf{p})^{i,j,k}(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_j(\overline{\mathbf{x}}_{\epsilon,t} + \mathbf{u}^{\mathcal{M}_t}(\overline{\mathbf{x}}_{\epsilon,t}) - \mathbf{x}_t)_k. \qquad (23)
\end{aligned}$$

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 16 *For $t \geq 1$,*

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

where

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

$\alpha = \frac{c}{1+c}$ *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 Eq. (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,

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

Let us simplify the above inequality.

- The orthogonal projection admits $\|\mathbf{I} - \mathbf{K}_{\mathbf{x}_t}\|_2 \leq 1$.
- Recall Proposition 15,

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

  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 Eq. (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 $\mathbf{v} = \mathbf{v}^{\|} \oplus \mathbf{v}^{\perp}$, where $\mathbf{v}^{\|} \in \mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$ and $\mathbf{v}^{\perp}$ lies in the orthogonal complement of $\mathcal{T}_{\mathbf{x}_0}\mathcal{M}_0$.

Recall Theorem 4,

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

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

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

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

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

Since e_t is defined for $t\ge 1$, 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},

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

■

The following proposition solves the difference inequality of linearization error.

Proposition 17 *If the perturbation satisfies*

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

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

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

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 16.

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

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

Then at $t + 1$, recall Proposition 16, given that $e_t \leq 1$ for all $t \leq T$,

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

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 $t \leq T$,

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

therefore,

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

■

Proposition 17 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).

C.3 Error Estimation

Now we reach the main theorem on the error estimation of $\|\overline{\mathbf{x}}_{\epsilon,t} - \mathbf{x}_t\|$.

Theorem 5 *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

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

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 4,

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

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

Recall Proposition 17 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 $t \geq 1$,

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

■

Appendix D. Optimal Control Versus Greedy Solution

We now formally discuss the difference between optimal control and greedy solutions. Let $\mathbf{V}_t$ be a orthonormal basis of the t^{th} linear embedding subspace, $\mathbf{Q}_t = \mathbf{I} - \mathbf{V}_t\mathbf{V}_t^T$ be a orthogonal projection onto the orthogonal complement of $\mathbf{V}_t$. Under the linear setting, the running loss (2) can be realized as

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

the exact solution of the above can be obtained by setting the gradient $\nabla_{\mathbf{u}_t}\mathcal{L} = \mathbf{0}$,

$$\mathbf{u}_t^{\text{greedy}}(\mathbf{x}_t) = -(c \cdot \mathbf{I} + \mathbf{Q}_t^T\mathbf{Q}_t)^{-1}\mathbf{Q}_t^T\mathbf{Q}_t\mathbf{x}_t.$$

Notice that $\mathbf{u}_t^{\text{greedy}}$ is considered as the greedy solution since it optimizes the t^{th} running loss without considering other layers. Furthermore, since $\mathbf{Q}_t = \mathbf{I} - \mathbf{V}_t\mathbf{V}_t^T$,

$$\mathbf{u}_t^{\text{greedy}} = -\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \frac{1}{1+c} & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{1+c} \end{bmatrix} \mathbf{V}_t^T\mathbf{x}_t.$$

The greedy solution $\mathbf{u}_t^{\text{greedy}}$ does not control the state components that lie in the r-dimensional embedding subspace (the first r diagonal elements are 0), it applies regularized control onto the components from orthogonal complements (as the regularization $c \to \infty$, the effect of applying control diminishes to identity mapping).

Now we present the closed-form solution for the optimal control solution. For the simplified linear system that contains linear orthogonal layers $\boldsymbol{\theta}_t$ such that $\boldsymbol{\theta}_t^T\boldsymbol{\theta}_t = \boldsymbol{\theta}_t\boldsymbol{\theta}_t^T = \mathbf{I}$, the following Lemma characterizes the optimal control solution.

Lemma 18 *For a simplified system with linear orthogonal layers, the optimal feedback control* $\mathbf{u}_t^{\text{optimal}}$ *is,*

$$\mathbf{u}_t^{\text{optimal}} = -\mathbf{V}_t \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & 1 - \frac{c}{1+\lambda_{t+1}+c} & 0 \\ 0 & 0 & \cdots & 0 & 1 - \frac{c}{1+\lambda_{t+1}+c} \end{bmatrix} \mathbf{V}_t^T\mathbf{x}_t,$$

where $\mathbf{V}_t$ *is a orthogonal basis of the linear embedding subspace,* $\lambda_T = 0$, $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$.

The detailed derivation is provided in Sec. D.1. When the control regularization $c = 0$, the diagonal elements become 1. In this case, the optimal control is the equivalent to the greedy solution. However, for any $c > 0$, the control regularization at each layer is governed by a difference equation that relates the t^{th} control to all succeeding layers. Essentially, the optimal control solutions regularize the applied controls differently at every layer.

D.1 Proof for Optimal Feedback Control

In this section, we derive the closed-form solution for optimal feedback control. Let $\boldsymbol{\theta}_t$ be the t^{th} linear transformation with control $\mathbf{u}_t$,

$$\mathbf{x}_{t+1} = \boldsymbol{\theta}_t(\mathbf{x}_t + \mathbf{u}_t). \tag{24}$$

Let $\mathbf{V}_t$ be a orthogonal basis of the t^{th} embedding subspace, $\mathbf{Q}_t = \mathbf{I} - \mathbf{V}_t\mathbf{V}_t^T$ be a orthogonal projection onto the orthogonal complement of $\mathbf{V}_t$.

Lemma 19 *For the the sum of the one-step costs over a finite horizon*

$$J(\mathbf{x}_0, \{\mathbf{u}\}_{t=0}^{T-1}) = \sum_{t=0}^{T-1} \frac{1}{2}\|\mathbf{Q}_t(\mathbf{x}_t + \mathbf{u}_t)\|_2^2 + \frac{c}{2}\|\mathbf{u}_t\|_2^2, \text{ s.t. } Eq.\ (24). \tag{25}$$

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

$$\begin{aligned}&\mathbf{P}_t \\ &= \frac{1}{2}\mathbf{Q}_t + \boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t - \frac{1}{2}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t)^T(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t + c_t\mathbf{I})^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t).\end{aligned} \tag{26}$$

with the optimal control solution

$$\mathbf{u}_t = -(\mathbf{Q}_t + c\mathbf{I} + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t)^{-1}(\mathbf{Q}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t)\mathbf{x}_t, \tag{27}$$

Proof The optimal cost-to-go function (value function $V(\mathbf{x}_t)$) of Eq. (25) satisfies

$$V(\mathbf{x}_t) = \min_{\mathbf{u}_t} \frac{1}{2}(\mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\mathbf{u}_t)^T(\mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\mathbf{u}_t) + \frac{c}{2}\cdot\mathbf{u}_t^T\mathbf{u}_t + V(\mathbf{x}_{t+1}), \text{ s.t. } Eq.\ (24). \tag{28}$$

By taking derivative on the right-hand-side of Eq. (28) and considering the dynamical system Eq. (24),

$$\begin{aligned}\frac{dV(\mathbf{x}_t)}{d\mathbf{u}_t} &= \mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\mathbf{u}_t + c\cdot\mathbf{u}_t + \left(\frac{d\mathbf{x}_{t+1}}{d\mathbf{u}_t}\right)^T\frac{dV(\mathbf{x}_{t+1})}{d\mathbf{x}_{t+1}}, \\ &= \mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\mathbf{u}_t + c\cdot\mathbf{u}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\mathbf{x}_{t+1}, \\ &= \mathbf{Q}_t\mathbf{x}_t + \mathbf{Q}_t\mathbf{u}_t + c\cdot\mathbf{u}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t\mathbf{x}_t + 2\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t\mathbf{u}_t.\end{aligned} \tag{29}$$

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

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

By parametrizing the value function $V(\mathbf{x}_t)$ as $\mathbf{x}_t^T\mathbf{P}_t\mathbf{x}_t$, consider the optimal control solution Eq. (27) and the dynamical programming equation Eq. (28),

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

for the second term in the above, recall the optimal control solution $\mathbf{u}_t^*$ from Eq, (27),

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

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

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

which results in the algebraic Riccati equation

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

■

We consider a special case that each linear transformation $\boldsymbol{\theta}_t$ is orthogonal and full-rank, such that $\boldsymbol{\theta}_t^T\boldsymbol{\theta}_t = \boldsymbol{\theta}_t\boldsymbol{\theta}_t^T = \mathbf{I}$, $\forall t$.

Lemma 20 *Given a T-layer system with orthogonal linear transformations, the solution of algebraic Riccati equation Eq.* (26) *is*

$$\mathbf{P}_t = \frac{1}{2}\mathbf{V}_t\begin{bmatrix}0 & 0 & \cdots & 0 & 0\\ 0 & 0 & \cdots & 0 & 0\\ \vdots & \vdots & \ddots & 0 & 0\\ 0 & 0 & \cdots & \lambda_t & 0\\ 0 & 0 & \cdots & 0 & \lambda_t\end{bmatrix}\mathbf{V}_t^T, \tag{30}$$

where $\lambda_T = 0$, $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$.

Proof We prove it by induction on t. Recall the algebraic Riccati equation Eq. (26), given the terminal condition $\mathbf{P}_T = \mathbf{0}$,

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

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

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

since $\boldsymbol{\theta}_t^T\boldsymbol{\theta}_t = \boldsymbol{\theta}_t\boldsymbol{\theta}_t^T = \mathbf{I}$, $\boldsymbol{\theta}_t^T\mathbf{V}_{t+1} = \mathbf{V}_t$, in which case, $\mathbf{Q}_t$ and $\boldsymbol{\theta}_t^T\mathbf{P}_{t+1}\boldsymbol{\theta}_t$ contain the same basis $\mathbf{V}_t$. Recall the algebraic Riccati equation Eq. (26),

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

■

Recall the optimal feedback control in Eq. (27), let $\lambda_T = 0$, $\lambda_t = \frac{c(1+\lambda_{t+1})}{1+\lambda_{t+1}+c}$.

Corollary 21 *For a system with linear orthogonal transformations, the optimal feedback control is*

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

References

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR, 2016.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, pages 484–501. Springer, 2020.

Kurt J Antreich, Helmut E Graeb, and Claudia U Wieser. Circuit analysis and optimization driven by worst-case distances. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(1):57–71, 1994.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018.

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

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

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

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

FL Chernousko and AA Lyubushin. Method of successive approximations for solution of optimal control problems. *Optimal Control Applications and Methods*, 3(2):101–114, 1982.

Charles Chien, Adrian Tang, Frank Hsiao, and Mau-Chung Frank Chang. Dual-control self-healing architecture for high-performance radio SoCs. *IEEE Design & Test of Computers*, 29(6):40–51, 2012.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020b.

Chunfeng Cui, Kaikai Liu, and Zheng 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(12):4958–4970, 2020.

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

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. *Advances in Neural Information Processing Systems*, 33:6616–6628, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

Abhilash Goyal, Madhavan Swaminathan, Abhijit Chatterjee, Duane C Howard, and John 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(10): 1835–1848, 2011.

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

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

Zichang He and Zheng Zhang. PoBO: A polynomial bounding method for chance-constrained yield-aware optimization of photonic ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.

Chung-Wen Ho, Albert Ruehli, and Pierce Brennan. The modified nodal approach to network analysis. *IEEE Transactions on circuits and systems*, 22(6):504–509, 1975.

Gokce Keskin, Jonathan Proesel, and Larry Pileggi. Statistical modeling and post manufacturing configuration for scaled analog CMOS. In *IEEE Custom Integrated Circuits Conference*, pages 1–4, 2010.

Jangjoon Lee, Srikar Bhagavatula, Swarup Bhunia, Kaushik Roy, and Byunghoo Jung. Self-healing design in deep scaled CMOS technologies. *Journal of Circuits, Systems, and Computers*, 21(06):1240011, 2012.

Qianxiao Li, Long Chen, Cheng Tai, and Weinan E. Maximum principle based algorithms for deep learning. *The Journal of Machine Learning Research*, 18(1):5998–6026, 2017.

Xin Li, Padmini Gopalakrishnan, Yang Xu, and T Pileggi. Robust analog/RF circuit design with projection-based posynomial modeling. In *IEEE/ACM International Conference on Computer Aided Design*, pages 855–862, 2004.

Xin Li, Padmini Gopalakrishnan, Yang Xu, and Lawrence T Pileggi. Robust analog/RF circuit design with projection-based performance modeling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(1):2–15, 2006.

Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1787, 2018.

Jenny Yi-Chun Liu, Adrian Tang, Ning-Yi Wang, Qun Jane Gu, Roc Berenguer, Hsieh-Hung Hsieh, Po-Yi Wu, Chewnpu Jou, and Mau-Chung Frank Chang. A V-band self-healing power amplifier with adaptive feedback bias control in 65 nm cmos. In *IEEE Radio Frequency Integrated Circuits Symposium*, pages 1–4, 2011.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

Indika Rajapakse and Mark Groudine. On emerging nuclear order. *Journal of Cell Biology*, 192(5):711–721, 2011.

Bodhisatwa Sadhu, Mark A Ferriss, Arun S Natarajan, Soner Yaldiz, Jean-Olivier Plouchart, Alexander V Rylyakov, Alberto Valdes-Garcia, Benjamin D Parker, Aydin Babakhani, Scott Reynolds, et al. A linearized, low-phase-noise vco-based 25-ghz pll with autonomic biasing. *IEEE Journal of Solid-State Circuits*, 48(5):1138–1150, 2013.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv:1805.06605*, 2018.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61: 85–117, 2015.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Shupeng Sun, Fa Wang, Soner Yaldiz, Xin Li, Lawrence Pileggi, Arun Natarajan, Mark Ferriss, Jean-Olivier Plouchart, Bodhisatwa Sadhu, Ben 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(8):2243–2252, 2014.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Adrian Tang, Frank Hsiao, David Murphy, I-Ning Ku, Jenny Liu, Sandeep D'Souza, Ning-Yi Wang, Hao Wu, Yen-Hsiang Wang, Mandy 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*, pages 316–318, 2012.

Mengshuo Wang, Fan Yang, Changhao Yan, Xuan Zeng, and Xiangdong Hu. Efficient bayesian yield optimization approach for analog and sram circuits. In *Design Automation Conference*, pages 1–6, 2017.

Ren Wang, Tianqi Chen, Stephen Lindsly, Cooper Stansbury, Indika Rajapakse, and Alfred Hero. Immuno-mimetic deep neural networks (immuno-net). *arXiv preprint arXiv:2107.02842*, 2021a.

Ren Wang, Tianqi Chen, Stephen Lindsly, Cooper Stansbury, Alnawaz Rehemtulla, Indika Rajapakse, and Alfred Hero. RAILS: A robust adversarial immune-inspired learning system. *arXiv preprint arXiv:2107.02840*, 2021b.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

Jian Cheng Zhang and MA Styblinski. *Yield and variability optimization of integrated circuits*. Springer Science & Business Media, 2013.