FESSNC: Fast Exponentially Stable and Safe Neural Controller

Jingdong Zhang [*1,2] **Luan Yang** [*2] **Qunxi Zhu** [2,3,4] **Wei Lin** [1,2,3,4]

Abstract

In order to stabilize nonlinear systems modeled by stochastic differential equations, we design a Fast Exponentially Stable and Safe Neural Controller (FESSNC) for fast learning controllers. Our framework is parameterized by neural networks, and realizing both rigorous exponential stability and safety guarantees. Concretely, we design heuristic methods to learn the exponentially stable and the safe controllers, respectively, in light of the classical theory of stochastic exponential stability and our established theorem on guaranteeing the almost-sure safety for stochastic dynamics. More significantly, to rigorously ensure the stability and the safety guarantees for the learned controllers, we develop a projection operator, projecting to the space of exponentially-stable and safe controllers. To reduce the highly computational cost for solving the projection operation, approximate projection operators are delicately proposed with closed forms that map the learned controllers to the target controller space. Furthermore, we employ Hutchinson's trace estimator for a scalable unbiased estimate of the Hessian matrix that is used in the projection operator, which thus allows for reducing computational cost and, therefore, can accelerate the training and testing processes. More importantly, our approximate projection operations are applicable to the non-parametric control methods, improving their stability and safety performance. We empirically demonstrate the superiority of the FESSNC over the existing methods.

[*]Equal contribution [1]School of Mathematical Sciences, LMNS, and SCMS, Fudan University, China. [2]Research Institute of Intelligent Complex Systems, Fudan University, China. [3]State Key Laboratory of Medical Neurobiology and MOE Frontiers Center for Brain Science, Institutes of Brain Science, Fudan University, China. [4]Shanghai Artificial Intelligence Laboratory, China. Correspondence to: Qunxi Zhu <qxzhu16@fudan.edu.cn>, Wei Lin <wlin@fudan.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Stabilizing the nonlinear systems modeled by stochastic differential equations (SDEs) is a challenging focal task in many fields of mathematics and engineering. The neural networks (NNs) equipped with the suitable loss of stability conditions have remarkable abilities in learning the stabilization controllers for the underlying systems, such as the neural Lyapunov control for ordinary differential equations (ODEs) (Chang et al., 2019) and the neural stochastic control for SDEs (Zhang et al., 2022). However, these frameworks, which heuristically train the NNs on the finite samples from the data space, are suffered from the lack of a rigorous stability guarantee for the learned neural controllers, i.e., satisfying the anticipated conditions on the whole data space.

Safety is another major concern for controlled dynamics, such as robotic and automotive systems (Wang et al., 2016). Recent advances in the control barrier function theory pave a direct way to impose the safety for the nonlinear SDEs. Several algorithms have theoretically investigated the structures of the barrier functions and designed the optimal control via quadratic programming (QP) or Sum-of-Squares (SOS) optimization according to the theoretical results (Sarkar et al., 2020; Mazouz et al.). Although all these algorithms try to obtain the controllers that rigorously satisfy the safety conditions, they either suffer high computational costs or only consider the safety with probability $\delta < 1$ rather than the probability one which is urgently required in the safety-critical applications.

In this paper, our goal is to provide a framework for fast learning the neural controllers with rigorous exponential stability and safety guarantees such that the trajectories of the controlled SDEs exponentially converge to the target equilibria and remain in the safe region all the time. The major contributions of this paper are summarized as follows.

- We train the neural controllers with the exponential stability conditions, and impose an exponential stability guarantee for the learned controller by projecting it into the space of the exponentially stable controllers. To avoid the high computational complexity of solving the projection operation, we propose an approximate projection operator $\hat{\pi}_{es}$ with a closed form that can

map the controller to the target space if and only if the state space of the controlled SDEs is bounded.

- We provide a theorem for restricting the controlled SDEs in the predefined bounded safe region based on the zeroing barrier functions. Similarly, we train the neural controller according to the safety loss and project it to the space of safe controllers with the approximate projection operator $\hat{\pi}_{sf}$. Theoretically, it can be verified that one can combine $\hat{\pi}_{es}$ and $\hat{\pi}_{sf}$ to provide both exponential stability and safety guarantees for the neural controllers.

- We introduce an unbiased stochastic estimator of trace estimation of the Hessian matrix involved in the projection operator with cheap computation.

- We demonstrate the efficacy of the FESSNC on a variety of classic control problems, such as, the synchronization of coupled oscillators. In addition to the neural network controllers, we also show our framework can be applied to the non-parametric controllers to provide stability and safety guarantee for them. Our code is available at `github.com/jingddong-zhang/FESSNC`.

2. Preliminaries

2.1. Problem Formulation

To begin with, we consider the SDE in a general form of

$$\mathrm{d}\boldsymbol{x}(t) = f(\boldsymbol{x}(t))\mathrm{d}t + g(\boldsymbol{x}(t))\mathrm{d}B_t,\ t \geq 0, \tag{1}$$

where $\boldsymbol{x}(0) = \boldsymbol{x}_0 \in \mathbb{R}^d$, $\{\boldsymbol{x}(t)\}_{t\geq 0} \subset \mathcal{X}$ is the state space, the drift term $f : \mathbb{R}^d \to \mathbb{R}^d$ and the diffusion term $g : \mathbb{R}^d \to \mathbb{R}^{d\times r}$ are Borel-measurable functions, and B_t is a standard r-dimensional (r-D) Brownian motion defined on probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ with a filtration $\{\mathcal{F}_t\}_{t\geq 0}$ satisfying the regular conditions. The state $\boldsymbol{x}^*$ with $f(\boldsymbol{x}^*) = \mathbf{0}$ and $g(\boldsymbol{x}^*) = \mathbf{0}$ is called equilibrium or zero solution.

Notations. Denote by $\|\cdot\|$ the Euclidean norm for a vector. Denote by $\|\cdot\|_{C(\mathbb{R}^d)}$ the maximum norm for the continuous function in $C(\mathbb{R}^d)$. $a \ll b$ means that the number a is much smaller than b. For a function $V \in C^2(\mathbb{R}^d)$, denote by ∇V and $\mathcal{H}V$ the gradient and the Hessian matrix, respectively.

Problem Statement We assume that the zero solution of the Eq. (1) is unstable, i.e. $\lim_{t\to\infty} \boldsymbol{x}(t) \neq \boldsymbol{x}^*$ on some set of positive measures. We aim to stabilize the zero solution using the neural controller with rigorous stability and safety guarantees that can be trained quickly. Specifically, we leverage the NNs to design an appropriate controller $\boldsymbol{u}(\boldsymbol{x})$ with $\boldsymbol{u}(\boldsymbol{x}^*) = \mathbf{0}$ such that the controlled system

$$\mathrm{d}\boldsymbol{x}(t) = f_{\boldsymbol{u}}(\boldsymbol{x}(t))\mathrm{d}t + g(\boldsymbol{x}(t))\mathrm{d}B_t \tag{2}$$

is steered to the zero solution with the whole controlled trajectory staying in the safety region, where $f_{\boldsymbol{u}} \triangleq f + \boldsymbol{u}$. Additionally, the neural controller should be optimized quickly in the training stage as well as in the testing stage such that it can be applied in the actual scenarios.

Basic Assumptions. To guarantee the existence of unique solutions, it is sufficient for Eq. (2) to own an initial condition and locally Lipschitz drift and diffusion terms. Therefore, we require the neural controllers to be locally Lipschitz continuous. This requirement is not overly onerous since the NNs can be designed by the compositions of Lipschitz continuous functions such as ReLU and affine functions. Without loss of generality, we assume that $\boldsymbol{x}^* = \mathbf{0}$.

To investigate the analytical properties of the controlled SDE (2), we introduce the stochastic derivative operator as follows.

Definition 2.1 ***(Derivative Operator)*** *Define the differential operator $\mathcal{L}_{\boldsymbol{u}}$ associated with Eq. (2) by*

$$\mathcal{L}_{\boldsymbol{u}} \triangleq \sum_{i=1}^{d} (f_{\boldsymbol{u}}(\boldsymbol{x}))_i \frac{\partial}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{d} [g(\boldsymbol{x})g^\top(\boldsymbol{x})]_{ij} \frac{\partial^2}{\partial x_i \partial x_j}.$$

The subscript of $\mathcal{L}_{\boldsymbol{u}}$ represents the dependence on $\boldsymbol{u}$.

According to the above definition of the derivative operator, an operation of $\mathcal{L}_{\boldsymbol{u}}$ on the function $V \in C^2(\mathbb{R}^d;\mathbb{R})$ yields:

$$\mathcal{L}V(\boldsymbol{x}) = \nabla V(\boldsymbol{x})^\top f_{\boldsymbol{u}}(\boldsymbol{x}) + \frac{1}{2}\mathrm{Tr}\left[g^\top(\boldsymbol{x})\mathcal{H}V(\boldsymbol{x})g(\boldsymbol{x})\right].$$

Next, we provide stability and safety analyses for Eq. (2) based on this derivative operator.

2.2. Exponential Stability

In stochastic stability theory, an exponentially stable dynamical system implies that all solutions in some region around an equilibrium exponentially converge to this equilibrium almost surely (a.s.). Generally, stochastic stability theory investigates the exponential stability for the dynamics by studying the behavior of a potential function V along the trajectories. Suppose the equilibrium is the minimum point of some potential function V. The convergence of the solution $\boldsymbol{x}(t)$ is equivalent to the descent of the potential energy $V(\boldsymbol{x}(t))$. In order to guarantee the exponential convergence, we can impose special structures for V such that the potential field $\mathcal{L}_{\boldsymbol{u}}V$ around the minimum is sinkable enough to attract the states from the high potential levels, as described in the following theorem.

Theorem 2.1 *(Mao, 2007) For the controlled SDE (2), suppose that there exists a potential function $V \in C^2(\mathcal{X}; \mathbb{R}_+)$ with $V(\mathbf{0}) = 0$, constants $p > 0$, $\varepsilon > 0$, $c \in \mathbb{R}$ such that*

$$\begin{aligned} &\varepsilon \|\boldsymbol{x}\|^p \leq V(\boldsymbol{x}), \\ &\mathcal{L}_{\boldsymbol{u}} V(\boldsymbol{x}) \leq cV(\boldsymbol{x}), \end{aligned} \tag{3}$$

for all $\boldsymbol{x} \in \mathcal{X}$. Then,

$$\limsup_{t\to\infty} \frac{1}{t} \log \|\boldsymbol{x}(t; t_0, \boldsymbol{x}_0)\| \leq \frac{c}{p} \ \ a.s.,$$

for all solution $\boldsymbol{x}(t; t_0, \boldsymbol{x}_0)$ initiated from $(t_0, \boldsymbol{x}_0)$. In particular, if $c < 0$, the zero solution of Eq. (2) is exponentially stable almost surely.

Learning Exponentially Stable Controller. In order to learn the exponentially stable controller for SDEs, Zhang et al. (Zhang et al., 2022) design a heuristic neural controller framework that integrates the conditions in Eq. (3) to the loss function. However, they can only assure the expected conditions are satisfied on finite training data instead of the whole state space. Hence, it lacks a rigorous stability guarantee and cannot be applied in stability-critical scenarios. In this paper, we circumvent this drawback for heuristic neural controllers through a projection operator.

Difference From the Lyapunov Theory One may naturally regard the potential function V and the Theorem 2.1, respectively, as the classic Lyapunov function and the Lyapunov stability theory for ODEs. The main difference is that the Lyapunov stability guarantees the decrease of the potential function, but the stochastic theory can only guarantee that the potential function eventually converges to a minimum. In other words, trajectories of exponentially stable SDEs may reach undesired states due to the stochasticity, which is not safe as we elaborate in the following subsection.

2.3. Safe Controller Synthesis

Safety-critical systems in application domains, such as automatic drive and medicine, require their state trajectories to stay in the specified safe region in order to prevent loss of life and economic harm. Here, we formalize the concept of safety in the following definition.

Definition 2.2 ***(Safe Controller)*** *The controller in the controlled system (2) is a safe controller if every solution $\boldsymbol{x}(t)$ initiated from $\boldsymbol{x}(0) \in \text{int}(\mathcal{C})$ satisfying:*

$$\forall t \geq 0 : \ \boldsymbol{x}(t) \in \mathcal{C} \ a.s.,$$

where $\mathcal{C}$ is a specified safe region induced by a locally Lipschitz function h as $\mathcal{C} = \{\boldsymbol{x} : h(\boldsymbol{x}) \geq 0\}$ and $\mathbf{0} \in \mathcal{C}$.

Recent works on the safe controller synthesis typically treat $\partial\mathcal{C} = \{\boldsymbol{x} : h(\boldsymbol{x}) = 0\}$ as a barrier, and require the controlled dynamics to satisfy additional conditions such that the trajectories never cross the barrier (Clark, 2019). In addition, existing works design complicated barrier functions, and/or the controller through the online quadratic program (QP) method, which requires a high computational cost (Fan et al., 2020; Sarkar et al., 2020). In Section 4, we propose a theorem for safety guarantee with a simpler barrier function and design an offline safe controller based on this result.

Connection to Safe RL The flow maps $\{S_t\}_{t\geq 0}$ of the Eq. (2) can be modeled as:

$$S_t(\boldsymbol{x}_0) \triangleq \boldsymbol{x}_0 + \int_0^t f_{\boldsymbol{u}}(\boldsymbol{x}(s))\mathrm{d}s + \int_0^t g(\boldsymbol{x}(s))\mathrm{d}B_s, \tag{4}$$

which induce a class of Markovian decision processes (MDPs) (Filar & Vrieze, 2012). Hence, the safe controller synthesis with a fixed time point t can be regarded as the safe reinforcement learning (RL) problems (Garcıa & Fernández, 2015). The major gap for applying RL methods to our problem is that, generally, the analytical expression of S_t is intractable given Eq. (2). And the unbounded diffusion term g further incurs difficulty for handling safety issues in RL. Besides, existing RL methods consider the safety in the meaning of average over trajectory or risk probability (Chow et al., 2017; Sootla et al., 2022), rather than satisfying the safety constraint almost surely (or with the probability one) as proposed in this paper.

3. Neural Controller with Exponential Stability Guarantee

We leverage the NNs to learn the neural controller with the exponential stability guarantee, i.e. satisfying the exponential stability condition as specified in Theorem 2.1. The specific process is divided into two steps: 1) we propose a heuristic neural framework for learning the neural controller according to an optimization problem based on Theorem 2.1; 2) A projection operator is developed to pull the learned controller into the stable domain, and we provide an operator with a closed form to approximate this projection operator in Theorem 3.1. Furthermore, we analyze the rationality of the conditions in Theorem 3.1 based on the safety of the controller.

3.1. Heuristic Neural Controller

A consensus in the control field is that the controllers should be as simple and small as possible (Lewis et al., 2012). Hence, we use a class of Lipschitz functions on $\mathcal{X}$, denoted as $\text{Lip}(\mathcal{X})$, to represent the space of candidate controllers. Incorporating with the Theorem 2.1, we can obtain the following optimal problem for learning the stabilization con-

troller:

$$\arg \min_{\boldsymbol{u}\in \mathrm{Lip}(\mathcal{X})} \frac{1}{2}\|\boldsymbol{u}(\boldsymbol{x})^\top R\boldsymbol{u}(\boldsymbol{x})\|_{C(\mathbb{R}^d)}$$
$$\text{s.t. } \varepsilon\|\boldsymbol{x}\|^p - V(\boldsymbol{x}) \le 0,$$
$$\mathcal{L}_{\boldsymbol{u}}V(\boldsymbol{x}) - cV(\boldsymbol{x}) \le 0,$$

where $\varepsilon > 0$, $c < 0$, $p > 0$ are predefined hyper-parameters, and semi-positive-definite matrix R measures the importance of different control components.

Neural Network Framework. In order to find the closed-form controller $\boldsymbol{u}$, we transform the above optimization problem into an NN framework. Specifically, we parameterize the controller by an NN, i.e., $\boldsymbol{u} = \boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x})$ with $\boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{0}) = \mathbf{0}$, where $\boldsymbol{\theta}$ is trainable parameter vector. Additionally, we constrain the Lipschitz constant of the neural controller by utilizing the spectral normalization method (Yoshida & Miyato, 2017; Miyato et al., 2018). The other Lipschitz neural networks can be used as well (Virmaux & Scaman, 2018; Liu et al., 2022a). Then, we construct the neural potential function $V_{\boldsymbol{\theta}}$ by using the convex functions (Amos et al., 2017) such that it satisfies the positive definite condition, i.e., $V_{\boldsymbol{\theta}}(\boldsymbol{x}) \ge 0$ and $V_{\boldsymbol{\theta}}(\mathbf{0}) = 0$. Second-order differentiable activation functions are used to guarantee $V_{\boldsymbol{\theta}} \in C^2(\mathbb{R};\mathbb{R}_+)$. We add L^p regularization term $\varepsilon\|\boldsymbol{x}\|^p$ with $\varepsilon \ll 1$ to $V_{\boldsymbol{\theta}}$ such that $V_{\boldsymbol{\theta}}(\boldsymbol{x}) \ge \varepsilon\|\boldsymbol{x}\|^p$. Then, we define the supervised stabilization loss function as follows.

Definition 3.1 ***(Stabilization Loss)*** *Consider a candidate potential function $V_{\boldsymbol{\theta}}$ and a controller $\boldsymbol{u}_{\boldsymbol{\theta}}$ for the controlled system (2). The exponential stabilization loss is defined as*

$$L_{es}(\boldsymbol{\theta}) \triangleq \frac{1}{N}\sum_{i=1}^{N}\big[\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top R\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \lambda_1 \max\big(0, \mathcal{L}_{\boldsymbol{u}_{\boldsymbol{\theta}}}V_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - cV_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\big)\big], \tag{5}$$

where $\lambda_1 > 0$ is a hyper-parameter representing the weight factor and $\{(\boldsymbol{x}_i)\}_{i=1}^N$ is the dataset.

Weaknesses of The Heuristic Framework. Two aspects could be improved in the current learning framework. First, the NNs trained on finite samples cannot guarantee the condition in the loss function is also satisfied in the infinite set $\mathcal{X}$. Second, the computational complexity of computing the Hessian matrix $\mathcal{H}V$ is $\mathcal{O}(d^2)$, a major bottleneck for applications in high-dimensional systems. We introduce the projection operation in the following subsection to overcome the first weakness. We defer the solution to the second weakness in Section 5.

3.2. Projection Operator for Exponential Stability

We define the space of the exponentially stable controllers as follows:

$$\mathcal{U}_{es}(V,\mathcal{X}) \triangleq \{\boldsymbol{u} \in \mathrm{Lip}(\mathcal{X}) : \mathcal{L}_{\boldsymbol{u}}V - cV \le 0\}.$$

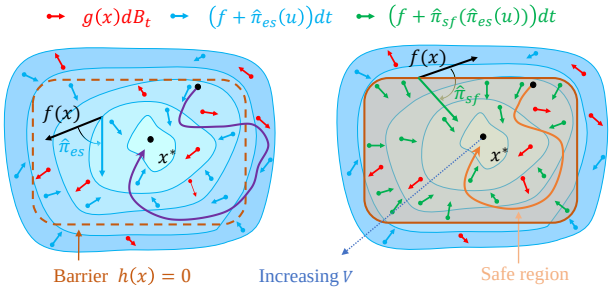

Figure 1. Illustration of the stable and safe controlled dynamics. **Left:** Operator $\hat{\pi}_{es}$ projects candidate controller $\boldsymbol{u}$ to the exponentially stable domain, i.e. $\mathcal{L}_{\boldsymbol{u}}V \le -cV$, s.t. any trajectory will exponentially converge to the equilibrium point $\boldsymbol{x}^*$. The trajectory cannot stay in the safe region (green shaded area) due to the stochasticity. **Right:** Operator $\hat{\pi}_{sf}$ project the controller to the safe domain, i.e. $\mathcal{L}_{\boldsymbol{u}}h \ge -\alpha(h)$, to restrict the trajectory (orange line) in the safe region.

To force the learned controller in $\mathcal{U}_{es}(V,\mathcal{X})$, a common technique is to find the L^2 projection of the controller which is defined as follows.

Definition 3.2 ***(Projection Operator)*** *Denote by $\pi(\boldsymbol{u},\mathcal{U})$ the projection from $\boldsymbol{u}$ onto the target space $\mathcal{U}$ as*

$$\pi(\boldsymbol{u},\mathcal{U}) \in \arg\min_{\tilde{\boldsymbol{u}}\in\mathcal{U}} \frac{1}{2}\|\boldsymbol{u}(\boldsymbol{x}) - \tilde{\boldsymbol{u}}(\boldsymbol{x})\|^2_{C(\mathbb{R}^d)}.$$

In General, it is difficult to obtain a closed form of the above projection operator. Recently, the QP method has been used to solve the following simplified projection for each data $\boldsymbol{x}$ as an alternative (Chow et al., 2019),

$$\tilde{\pi}(\boldsymbol{u})(\boldsymbol{x}) \in \arg\min \frac{1}{2}\|\boldsymbol{u}(\boldsymbol{x}) - \tilde{\boldsymbol{u}}(\boldsymbol{x})\|^2,$$
$$\text{s.t. } \mathcal{L}_{\tilde{\boldsymbol{u}}}V(\boldsymbol{x}) - cV(\boldsymbol{x}) \le 0.$$

However, this simplified operation $\tilde{\pi}$ cannot assure the continuity of the projection and causes additional computational costs. In the following theorem, we approximate the projection operator π with a closed form.

Theorem 3.1 ***(Approximate Projection Operator)*** *For a candidate controller $\boldsymbol{u}$ and the target space $\mathcal{U}_{es}(V,\mathcal{X})$, an approximate projection operator is defined as follows,*

$$\hat{\pi}_{es}(\boldsymbol{u},\mathcal{U}_{es}(V,\mathcal{X})) \triangleq \boldsymbol{u} - \frac{\max(0, \mathcal{L}_{\boldsymbol{u}}V - cV)}{\|\nabla V\|^2}\cdot\nabla V, \tag{6}$$

then, the operator $\hat{\pi}_{es}$ satisfies $\hat{\pi}_{es}(\boldsymbol{u},\mathcal{U}_{es}(V,\mathcal{X})) \in \mathcal{U}_{es}(V,\mathcal{X})$ if and only if the state space $\mathcal{X}$ of the controlled systems (2) is bounded.

The proof is shown in the Appendix A.1.2. We can directly apply this approximate projection operator to the learned

controller to provide the exponential stability guarantee when $\mathcal{X}$ is bounded. However, this bounded condition may not hold on a general exponentially stable system, the following example illustrates this point.

Example 3.2 *(**Exponential Martingale**) Consider a 1-D SDE described by* $\mathrm{d}x = ax\mathrm{d}t + bx\mathrm{d}B_t$, *where* $a < 0$, $|a| > \frac{1}{2}b^2$ *and* $x_0 > 0$, *the zero solution is exponentially stable, but for any* $M > 0$, $\mathbb{P}(x(t) > M) > 0$, *i.e.* $\mathcal{X} = \mathbb{R}$.

See the poof in the Appendix A.1.3. This example implies that the boundedness of state space $\mathcal{X}$ cannot be induced under the exponential stability. As introduced in Section 2.3, if the controller is safe in the bounded safe region C, then $\mathcal{X}$ is bounded for any $\boldsymbol{x}_0 \in \mathcal{C}$. We, therefore, take safety into consideration to bound the $\mathcal{X}$. Furthermore, this example shows that a stable controller may not be safe.

4. Neural Controller with Safety Guarantee

In this section, we present an approach for safe controller synthesis. Similar to the analysis in Section 3, the safe controller's construction can be divided into two steps as well: 1) training a heuristic neural safe controller (Section 4.1), and 2) projecting the learned controller to obtain the safety guarantee in the post-training stage (Section 4.2). In particular, we propose a new sufficient condition for the safe controller based on the barrier function theory (Ames et al., 2016). Then one can design a neural safe controller according to this result. In addition, we show that the intersection of the space of the safe controller and $\mathcal{U}_{es}$ is not empty, which assures the existence of a controller with both safety and stability guarantees.

4.1. Zeroing Barrier Function for Safety

As specified in Section 2.3, the primary task for safe controller synthesis is to impose additional structures for the controlled system (2) to restrict the trajectory, never approaching the barrier $\partial\mathcal{C}$. Here, we present a concise structure for safety based on the zeroing barrier function.

Definition 4.1 *(Ames et al., 2016) A continuous function* $\alpha : (0, a) \to (-\infty, \infty)$ *is said to belong to class-*$\mathcal{K}$ *for some* $a > 0$ *if it is strictly increasing and* $\alpha(0) = 0$.

Theorem 4.1 *For the SDE (2), if there exists a class-*$\mathcal{K}$ *function* $\alpha(x)$ *such that*

$$\mathcal{L}_{\boldsymbol{u}} h(\boldsymbol{x}) \geq -\alpha(h(\boldsymbol{x})) \tag{7}$$

for $\boldsymbol{x} \in \mathcal{C}$. *Then, the controller* $\boldsymbol{u}$ *is a safe controller, and* $h(\boldsymbol{x})$ *is called a zeroing barrier function (ZBF).*

The proof is provided in Appendix A.1.4. An intuition behind this theorem is that $\alpha(h)$ can be regarded as a potential function. When the trajectory $\boldsymbol{x}(t)$ approaches the barrier $\partial\mathcal{C}$, the potential $\alpha(h(\boldsymbol{x}(t)))$ decreases to the minimum 0 so that $\mathcal{L}_{\boldsymbol{u}} h \geq 0$, which in turn lifts the potential to pull the trajectory back to the interior of $\mathcal{C}$.

Neural Safe Controller. Given the ZBF $h(\boldsymbol{x})$, we can also formulate the safety problem as an optimization problem,

$$\begin{aligned} &\arg \min_{\boldsymbol{u} \in \mathrm{Lip}(\mathcal{C}), \alpha \in \mathcal{K}} \frac{1}{2} \|\boldsymbol{u}(\boldsymbol{x})^\top R \boldsymbol{u}(\boldsymbol{x})\|_{C(\mathbb{R}^d \times \mathbb{R}_+)} \\ &\text{s.t. } -\mathcal{L}_{\boldsymbol{u}} h(\boldsymbol{x}) - \alpha(h(\boldsymbol{x})) \leq 0. \end{aligned}$$

To convert this problem into a learning-based framework, we adopt the same neural controller $\boldsymbol{u}_{\boldsymbol{\theta}}$ as in Section 3.1. We apply the monotonic NNs to construct the candidate extended class-$\mathcal{K}$ function as $\alpha_{\boldsymbol{\theta}}(x) = \int_0^x q_{\boldsymbol{\theta}}(s)\mathrm{d}s$, where $q_{\boldsymbol{\theta}}(\cdot)$, an NN, is definitely positive (Wehenkel & Louppe, 2019). Then we train the monotonic NNs with the safe loss as follows.

Definition 4.2 *(**Safety Loss**) For a candidate class-*$\mathcal{K}$ *function* $\alpha_{\boldsymbol{\theta}}$ *and a controller* $\boldsymbol{u}_{\boldsymbol{\theta}}$, *the safety stabilization loss is defined as*

$$\begin{aligned} L_{sf}(\boldsymbol{\theta}) \triangleq \frac{1}{N} \sum_{i=1}^{N} \big[&\boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top R \boldsymbol{u}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \\ &+ \lambda_2 \max\left(0, -\mathcal{L}_{\boldsymbol{u}_{\boldsymbol{\theta}}} h(\boldsymbol{x}_i) - \alpha_{\boldsymbol{\theta}}(h(\boldsymbol{x}_i))\right) \big], \end{aligned} \tag{8}$$

where $\lambda_2 > 0$ *is a tunable weight factor and* $\{(\boldsymbol{x}_i)\}_{i=1}^N$ *is the dataset.*

4.2. Projection Operator for Safety

Notably, the learned neural safe controller may not satisfy the expected condition in Eq. (7). However, we can project it to the safe controller's space $\mathcal{U}_{sf}(\alpha, \mathcal{C}) \triangleq \{\boldsymbol{u} \in \mathcal{U}(\mathcal{C}) : -\mathcal{L}_{\boldsymbol{u}} h - \alpha(h) \leq 0\}$ as $\pi(\boldsymbol{u}_{\boldsymbol{\theta}}, \mathcal{U}_{sf}(\alpha_{\boldsymbol{\theta}}, \mathcal{C}))$. To efficiently implement the projection, we construct an approximate projection operator with a closed form again.

Theorem 4.2 *For the bounded safe region* $\mathcal{C}$, *the approximate projection operator defined as*

$$\hat{\pi}_{sf}(\boldsymbol{u}, \mathcal{U}_{sf}(\alpha, \mathcal{C})) \triangleq \boldsymbol{u} + \frac{\max(0, -\mathcal{L}_{\boldsymbol{u}} h - \alpha(h))}{\|\nabla h\|^2} \cdot \nabla h, \tag{9}$$

satisfies $\hat{\pi}_{sf}(\boldsymbol{u}, \mathcal{U}_{sf}(\alpha, \mathcal{C})) \in \mathcal{U}_{sf}(\alpha, \mathcal{C})$.

Remark 4.3 *To maintain* $\hat{\pi}_{sf}(\boldsymbol{u}, \mathcal{U}_{sf}(\alpha, \mathcal{C}))(\mathbf{0}) = \mathbf{0}$, *we further require* $\nabla h(\mathbf{0}) = \mathbf{0}$, *this condition is easy to satisfy since one can modify the ZBF around the equilibrium without change the safe region* $\mathcal{C}$.

The proof is shown in Appendix A.1.5. Based on this approximate projection operator in the post-training stage, one can transform the learned controller into a safe controller $\hat{\pi}_{sf}(\boldsymbol{u}_{\boldsymbol{\theta}}, \mathcal{U}_{sf}(\alpha_{\boldsymbol{\theta}}, \mathcal{C}))$.

Controller with Both Safety and Exponential Stability. Actually, we can jointly train the $\boldsymbol{u}_{\boldsymbol{\theta}}, V_{\boldsymbol{\theta}}, \alpha_{\boldsymbol{\theta}}$ with the summed loss $L_{es}(\boldsymbol{\theta}) + L_{sf}(\boldsymbol{\theta})$ to learn a safe and exponentially stable controller with the finite sample size. A natural question is whether we can composite the approximate projection operator $\hat{\pi}_{es}$ and $\hat{\pi}_{sf}$ together to get the controller with rigorous safety and stability guarantees or not. To answer this question, we study the relationship between the safe controller space $\mathcal{U}_{sf}(\cdot, \mathcal{C})$ and the exponentially stable controller space $\mathcal{U}_{es}(\cdot, \mathcal{C})$.

Theorem 4.4 ***(Existence of the Safe and Exponentially Stable Controller)*** *For any ZBF $h(\boldsymbol{x})$ with a bounded safe region $\mathcal{C}$ and a class-$\mathcal{K}$ function α, there exists a potential function V, s.t. $\mathcal{U}_{es}(V, \mathcal{C}) \cap \mathcal{U}_{sf}(\alpha, \mathcal{C}) \neq \emptyset$.*

Proof Sketch. For the special case that $h(\mathbf{0}) = \max_{\boldsymbol{x} \in \mathcal{C}} h(\boldsymbol{x})$, we can construct the potential function as $V(\boldsymbol{x}) = h(\mathbf{0}) - h(\boldsymbol{x})$ s.t. $V \geq k\|\boldsymbol{x}\|$ for some $k > 0$. Then it can be shown that there exists $\boldsymbol{u} \in \mathcal{U}_{sf}(\alpha, \mathcal{C})$ s.t. $\mathcal{L}_{\boldsymbol{u}} V \leq -kV$. If $h(\mathbf{0}) \neq \max_{\boldsymbol{x} \in \mathcal{C}} h(\boldsymbol{x})$, we can modify the ZBF as $\tilde{h} = h + \lambda$ s.t. $\tilde{h}(\mathbf{0}) = \max_{\boldsymbol{x} \in \mathcal{C}} \tilde{h}(\boldsymbol{x})$ without changing the safe region, where λ is a smooth approximation of the Dirac function $\delta_{\mathbf{0}}$. Similar to the special case, we can derive the remaining parts of the theorem. The detail of the proof is shown in Appendix A.1.6.

Remark 4.5 *The bounded safe region condition is Theorem 4.4 can be relaxed to unbounded cube $[a_1, b_1] \times \cdots \times [a_d, b_d]$, $-\infty \leq a_i < b_i \leq \infty$, where at least one interval $[a_i, b_i]$ is bounded. The proof is provided in Appendix A.1.6.*

According to Theorem 4.4, we can first apply operator $\hat{\pi}_{sf}$ to the learned controller $\boldsymbol{u}_{\boldsymbol{\theta}}$ to obtain the safety guarantee and restrict the state space $\mathcal{X} \subset \mathcal{C}$. Since $\mathcal{X}$ is bounded now, then, we can apply operator $\hat{\pi}_{es}$ to the safe controller to obtain the exponential stability guarantee according to Theorem 3.1.

5. Accelerate the Training and the Projection

In general, computing $\mathcal{L}_{\boldsymbol{u}} V$ costs $\mathcal{O}(d^2)$. The term $\mathrm{Tr}[g^\top \mathcal{H} V g]$ requires computing a Hessian matrix of V. The same is valid for computing $\mathcal{L}_{\boldsymbol{u}} h$. Here, two tricks can be used to achieve low computational costs. First, due to the commutativity of the trace operator, we can get the following unbiased estimate:

$$\mathrm{Tr}[g^\top \mathcal{H} V g] = \mathbb{E}[\xi^\top \mathcal{H} V g g^\top \xi] = \mathbb{E}[(\nabla(\xi^\top \nabla V))^\top g g^\top \xi],$$

where ξ is a d-D noise vector with mean zero and variance I. The Monte Carlo estimator for the above expectation is known as the Hutchinson's trace estimator (Hutchinson, 1989). Second, if g is a vector independent of $\boldsymbol{x}$, computing the gradient of the vector-Jacobian product $g^\top \nabla V$ costs $\mathcal{O}(d)$, and the remaining part is simply a vector-Jacobian product $\mathrm{Tr}[g^\top \mathcal{H} V g] = g^\top \nabla(g^\top \nabla V)$. These two tricks can be used according to the shape of the diffusion term $g(\boldsymbol{x}, t) \in \mathbb{R}^{d \times r}$, i.e., matrix ($r > 1$) or vector ($r = 1$).

Case 1: $r > 1$**.** In this case, g is a matrix, and we use the former trick to accelerate the training stage. Assuming the cost of evaluating $\mathrm{Tr}[g^\top \mathcal{H} V g]$ is on the order of $\mathcal{O}(Nd^2)$ where d is the dimensionality of the data and N is the size of the dataset, then the cost of computing the following Hutchinson's trace estimator via automatic differentiation, which only involve vector-Jacobian products, is $\mathcal{O}(NMd)$,

$$\mathrm{Tr}[g^\top \mathcal{H} V g] \approx \frac{1}{M} \sum_{i=1}^{M} (\nabla(\xi_i^\top \nabla V))^\top g g^\top \xi_i, \tag{10}$$

where M is the number of samples in the Monte Carlo estimation with $M \ll d$. In practice, M can be tuned to trade off the variance and the computational cost. As suggested in (Grathwohl et al., 2018; Song et al., 2020), $M = 1$ is already a good choice, which worked well in our experiments. Notice that although the approximate projection operator applied to the learned controller in the projection stage also involves computing the $\mathcal{L}_u V$, we cannot use this trick here because it does not equal the real trace.

Case 2: $r = 1$**.** Since g is a vector, we can use the latter trick by removing the dependency of g on data $\boldsymbol{x}$. Specifically, after getting the $g(\boldsymbol{x})$ on data $\boldsymbol{x}$, we apply the `detach` operator in Pytorch (Paszke et al., 2019) to g to prevent calculating the gradients w.r.t $\boldsymbol{x}$. Then the computational cost reduces from $\mathcal{O}(Nd^2)$ to $\mathcal{O}(Nd)$. This trick is shown as follows:

$$\mathrm{Tr}[g^\top \mathcal{H} V g] = g^\top \nabla((g.\texttt{detach}())^\top \nabla V). \tag{11}$$

We note that the result in Eq. (11) is an identity equation. Hence, it is an ideal choice when $r = 1$. Since the derivative operator $\mathcal{L}_{\boldsymbol{u}}$ is calculated in both the training and post-training projection stages, and these two cases always cover at least one stage, we can accelerate the whole framework of the proposed safe and exponentially stable controller. We summarized the complete framework of the proposed FESSNC in Algorithm 1.

6. Experiments

In this section, we demonstrate the superiority of the FESSNC over existing methods using several case studies, then we validate the scalability of FESSNC to the high dimensional system with an application in synchronizing the coupled oscillators. More details of the experiments can be found in Appendix B.

Algorithm 1 FESSNC

Input: Dynamics f, g, ZBF h, safe region $\mathcal{C}$, max iterations m, learning rate γ, initial parameters $\boldsymbol{\theta}$, weight factors $\lambda_{1,2}$, noise vector samples $\{\xi_i\}_{i=1}^M$.
Output: Controller $\boldsymbol{u}_{\boldsymbol{\theta}}$, potential function $V_{\boldsymbol{\theta}}$, class $\mathcal{K}$ function $\alpha_{\boldsymbol{\theta}}$.
for $r = 1 : m$ **do**
  $\{\boldsymbol{x}_i\}_{i=1}^N \sim \mathcal{C}$ ▷ Sample training data
  $L(\boldsymbol{\theta}) = L_{es}(\boldsymbol{\theta}) + L_{sf}(\boldsymbol{\theta})$ ▷ Compute loss (5)(8)(10)
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \cdot \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ ▷ Update parameters
Return: $\hat{\pi}_{es}(\hat{\pi}_{sf}(\boldsymbol{u}_{\boldsymbol{\theta}}, \mathcal{U}_{sf}(\alpha_{\boldsymbol{\theta}}, C)), \mathcal{U}_{es}(V_{\boldsymbol{\theta}}, \mathcal{C}))$
  ▷ Safety and exponential stability guarantee (6)(9)(11)

6.1. Case Studies

We first show that our approach is able to produce reliable control policies with stability and safety guarantees for each task. We then show the advantages of the FESSNC in computational costs compared with representative baselines.

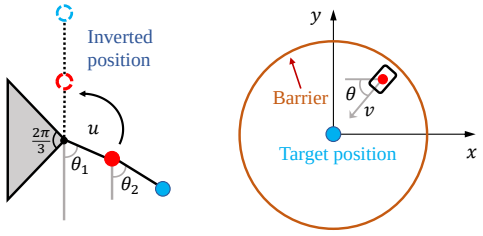

Figure 2. Schematic diagrams of the double pendulum (**left**) and the kinetic bicycle (**right**) with state constraints.

Models We consider two classic control models, the fully actuated double pendulum swing-up and the fully actuated planar kinetic bicycle motion. The double pendulum is a 4-D system with 2 angles and 2 angular velocities, $(\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2)$ (Deisenroth et al., 2013). We add multiplication noise force to the system such that the angular acceleration is perturbed related to angles. The objective is to learn a control strategy that steers the double pendulum to the inverted position within the safe region where the inner pendulum can only spin in $[-\pi/6, 7\pi/6]$, see Figure 2 (left). The kinematic bicycle model for car-like vehicles has 4 variables (x, y, θ, v), where (x, y) is the bicycle's position, θ is the direction of motion, and v is the velocity (Rajamani, 2011). The goal is to steer the noise-perturbed bicycle to the target position without crossing the round wall with a radius of 2, see Figure 2 (right).

Baselines To simulate the trajectories of the controlled system, we approximate the flow S_t of the SDEs (4) with the Euler–Maruyama method (Särkkä & Solin, 2019). Then the obtained MDPs (or discrete dynamics) can be handled with model-based RL methods. We compare the proposed FESSNC with the GP-MPC algorithm (Kamthe & Deisenroth, 2018) and the BALSA algorithm (Fan et al., 2020), respectively the most data-efficient MPC-based and QP-based RL algorithms to date.

We conduct 5 independent experiments, and the results are shown in Figure 3-4, and Table 4. In the figures, the solid lines are obtained by averaging the sampled trajectories, while the shaded areas stand for the variance regions.

Performance. As shown in Figure 3, all three methods satisfy the safety constraint for the double pendulum task. The FESSNC and GP-MPC can quickly steer the pendulum to the inverted position, but the GP-MPC cannot balance the pendulum abidingly, whereas the FESSNC does. The BALSA performs poorly because it requires the potential function V and ZBF h to be defined prior, and the predefined functions may induce disjoint safe controllers space and exponentially stable controllers space. The results in the kinetic bicycle task show the same situations as the double pendulum in Figure 4. The motion trajectories of the bicycle under the FESSNC are the most robust.

Computational Cost. We compare the training time of one complete control process for the FESSNC and baselines. The results are shown in Table 4. We can see that the GP-MPC costs much more time than the others. This is because the GP-MPC method has to roll out the forward dynamics to obtain the predictive and adjoint states, and then find the optimal control through gradient descent. The BALSA is slightly quicker than FESSNC, but its performance is far inferior to the latter. Moreover, QP-based solver costs at least $\mathcal{O}(d^2)$ for d-D system (Kouzoupis et al., 2018), which makes it hard to be applied in the high-dimensional system, whereas our FESSNC does.

Further investigations. We further compare the FESSNC with the existing learning based controllers, including SYNC (Zhang et al., 2023), NNDM controller

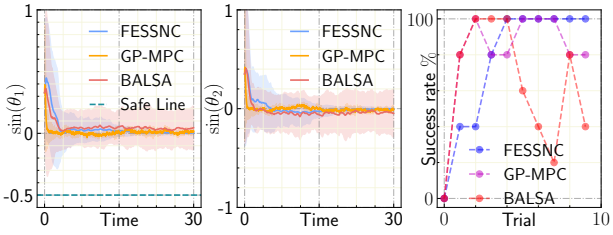

Figure 3. Results of the constraint double pendulum via different methods. (**left** resp., **center**)) Sine values along the trajectories of θ_1 (resp., θ_2). Safe region is $\sin(\theta_1) \leq \frac{1}{2}$. (**right**) The success rate of the three methods is 3 seconds per trial. We define 'success' if the pendulum tip's angle is closer than $\frac{\pi}{40}$ to the upright position for consecutive 3 seconds.

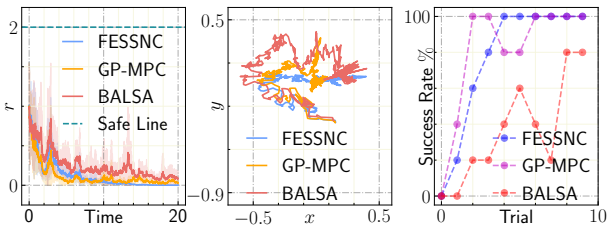

Figure 4. Results of the constraint kinematic bicycle model. (**left**) The distance r between the car and the target position. The safe region is $x^2 + y^2 \leq 4$. (**center**) The mean of motion trajectories. (**right**) The success rate of the three methods is 3 seconds per trial. We define 'success' if the car is closer than 0.1 to the target position for consecutive 2 seconds.

Table 1. Average amount of time in seconds needed to complete a control process for each method.

Method	Double Pendulum	Kinetic Bicycle
FESSNC	10.35s	6.39s
GP-MPC	146.64s	26.68s
BALSA	3.08s	2.18s

(NNDMC) (Mazouz et al.) and RSM controller (RSMC) (Lechner et al., 2022), using the kinetic bicycle model. The results provided in Table 2 demonstrate the superiority of the FESSNC over the existing methods in terms of stability, safety, scalability and control energy. For the stability guarantee, we achieve the exponential stability (ES), which is stronger than the asymptotic stability (AS). For the safety guarantee, we assure the controlled trajectories stay in the safe region almost surely (a.s.) instead of with some probability $p < 1$. We emphasize that we are the first to achieve the rigorously theoretical guarantee for stability and safety, while all the other methods rely on the numerical approximation to some extent in the guarantee process. We provide more details in Appendix B.3.1.

Extension to the nonparametric settings. Although we employ the neural controller as the candidate controller, our framework also works for the nonparametric controller by directly applying the approximated operators in Eqs. (6),(9) to it. We apply our framework to the kernel machine based controller and compare it with the FESSNC using a 3-link pendulum control task, the results shown in Figure 5 demonstrate our approximate projection operations have the ability to improve the performance of the kernel based method without further training. More details are provided in Appendix B.5

6.2. Synchronization of Coupled Oscillators

The numerical experiments in the previous section demonstrated the feasibility of the proposed framework. Here, we

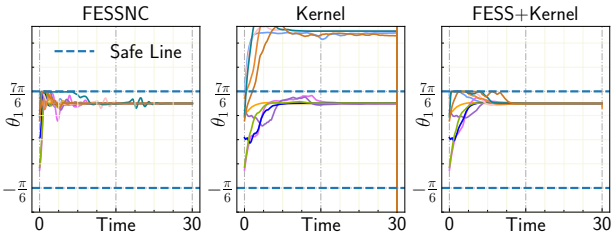

Figure 5. The angle of the first link $\theta_1(t)$ over 10 time trajectories. The green dashed lines represent the boundary of the safe region.

introduce a further application of FESSNC in the synchronization of coupled oscillators to validate the scalability of FESSNC in high dimensional system. Phase synchronization phenomenon in coupled oscillatory systems has been extensively studied in the context of collective behavior of complex networks from all scales and domains (Rosenblum et al., 2001; Sauseng & Klimesch, 2008; Strogatz, 2004). In this section, we show that the FESSNC can synchronize the incoherent coupled dynamics in stochastic settings. Specifically, we consider a small world network of n coupled FitzHugh-Nagumo (FHN) models (Conley & Smoller, 1986) under the stochastic coupled forces as:

$$\mathrm{d}v_i = (v_i - \frac{v_i^3}{3} - w_i + 1)\mathrm{d}t + \frac{1}{3}\sum_{i=1}^{n} L_{ij} v_j \mathrm{d}B_t,$$
$$\mathrm{d}w_i = 0.1(v_i + 0.7 - 0.8w_i)\mathrm{d}t,$$

where the fast variable v and the slow variable w correspond to the membrane potential and the recovery variable in an excitable nerve cell, respectively (Keener & Sneyd, 1998). We select $n = 50$ s.t. the networked dynamic is a 100-dimensional system. We denote the state variables by $\boldsymbol{x} = (v, w)$. Although the single FHN model has a stable limit cycle Γ, the synchronization manifold $\mathcal{M} \triangleq \{\boldsymbol{x}_1(t) \subset \Gamma : \boldsymbol{x}_i(t) = \boldsymbol{x}_1(t), \forall i\}$ of the coupled system is not stable due to the stochasticity. We now stabilize the coupled system to the synchronization manifold $\mathcal{M}$ by stabilizing the equilibrium of the equation of variance $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{x}_1$ with the FESSNC. The safe region is $\{\tilde{\boldsymbol{x}}_{1:n} : \max_{1\leq i\leq n}(\tilde{v}_i, \tilde{w}_i) \leq 5\}$ under the ZBF $h(\tilde{\boldsymbol{x}}_{i:n}) = 25 - \max_{1\leq i\leq n}(\tilde{v}_i^2, \tilde{w}_i^2)$. The results are shown in Figure 6. It can be seen that the controlled coupled FHN dynamics converge to the synchronization manifold quickly without crossing the safe region. The experimental details are provided in Appendix.

7. Related Works

Stabilization of Nonlinear Systems via Deep Learning Recent works have focused on finding the stabilization controllers for dynamical systems through deep learning (Chang et al., 2019; Schlaginhaufen et al., 2021; Dawson et al., 2022; Zhang et al., 2022). Prior works have also studied

Table 2. Comparison of learning-based controllers.

Index	Method				
	FESSNC	SYNC	NNDMC	RSMC	RSMC+ICNN
Training time (sec)	**14**	125	209	435	987
Safety rate(%)	**100**	90	100	10	90
Success rate (%)	**100**	90	60	0	90
Control energy	**1.5**	1.8	2.6	14.5	1.8
Computational Complexity	$\mathcal{O}(\mathbf{d})$	$\mathcal{O}(k^d d^2)$	$\mathcal{O}(k^d d^3)$	$\mathcal{O}(k^d)$	$\mathcal{O}(k^d)$
Stability guarantee (Type)	Yes (ES)	Yes (AS)	No	Yes (AS)	Yes (AS)
Safety guarantee (Type)	Yes (a.s.)	Yes (a.s.)	Yes (probability)	No	No
Type of guarantee	Theoretical	Numerical	Numerical	Numerical	Numerical

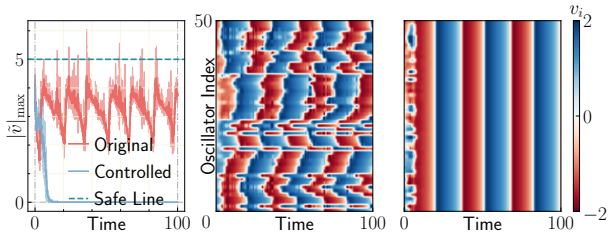

Figure 6. The maximum value of the variances' modulus $|\tilde{v}|_{\max} \triangleq \max_{1\le i\le 50} |v_i|$ along the trajectories of the original dynamics and the controlled dynamics (**left**). The solid lines are obtained by averaging the 5 sampled trajectories, while the shaded areas stand for the variance regions. Dynamics of the small-world network of coupled FHN without control (**center**) and with control (**right**). The color represents the membrane potential state v_i of each oscillator.

how to rigorously satisfy the expected conditions in the deep learning settings, including projecting the learned dynamics to the stable region (Kolter & Manek, 2019) and solving a constrained optimization problem at each parameters updating iteration (Chow et al., 2019). The drawbacks of these methods are that they either change the equilibria of the original dynamics or cause large computational costs.

Control Barrier Functions for Safety Recent works have sought to incorporate reciprocal barrier functions, namely the reciprocal of $h(\boldsymbol{x})$, with suppressing explosion techniques to devise safe controllers (Clark, 2019; Jankovic, 2018; Ames et al., 2019). These works impose complicated conditions for the safety, which makes it difficult for us to construct the projection operator. In contrast to the RBFs, ZBFs have (Taylor et al., 2020; Taylor & Ames, 2020) be used to construct the safety-critical control for ODEs with more simple structures. Dawson et al. have utilized the ZBFs in the neural robotic control framework (Dawson et al., 2022). We notice that only the RBFs for SDEs have been intensely studied while the ZBFs for SDEs have not been investigated yet due to the difficulties in theoretical analyses (Clark, 2019; 2021).

8. Conclusion and Future Works

We have proposed FESSNC, a two-stage framework that learns the neural controllers for stabilizing SDEs with exponential stability and safety guarantees. We present a new theorem on safety criteria for SDEs based on the ZBFs that are more concise than the existing theoretical results. In addition, we develop projection operators to project the learned controllers to the space of exponentially-stable and safe controllers, and we further present approximate projection operators with closed forms to reduce the computational costs. Notably, we prove the existence of both exponentially stable and safe controllers. We accelerate the training and testing processes with trace estimation methods. The empirical studies show our framework, an offline control policy, performs better the the SOTA online methods GP-MPC and BALSA. In addition to neural controllers, our projection operators can be applied to the nonparametric controllers to improve their performance.

Our work can be applied in stability-critical and safety-critical real-world scenarios. An exciting direction for future work would be to extend the existing model-based offline framework to online learning or model-free cases in specific scenarios (Zhang et al., 2024; Zhu et al., 2019). Moreover, we consider the deterministic control instead of the stochastic control proposed in (Zhang et al., 2022) because it's difficult to construct an approximate projection operator for stochastic control, and we leave it for future work. Finally, how to extend our current fully actuated control framework to the under actuated control settings appeared in many real-world scenarios is a challenging direction.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgements

J. Zhang is supported by the China Scholarship Council (No. 202306100154). Q. Zhu is supported by the China Postdoctoral Science Foundation (No. 2022M720817), by the Shanghai Postdoctoral Excellence Program (No. 2021091), and by the STCSM (Nos. 21511100200, 22ZR1407300, 22dz1200502, and 23YF1402500). W. Lin is supported by the NSFC (Grant No. 11925103), by the STCSM (Grants No. 22JC1402500 and No. 22JC1401402), and by the SMEC (Grant No. 2023ZKZD04). The computational work presented in this article is supported by the CFFF platform of Fudan University.

References

Ames, A. D., Xu, X., Grizzle, J. W., and Tabuada, P. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

Ames, A. D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., and Tabuada, P. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pp. 3420–3431, 2019. doi: 10.23919/ECC.2019.8796030.

Amos, B., Xu, L., and Kolter, J. Z. Input convex neural networks. In *International Conference on Machine Learning*, pp. 146–155. PMLR, 2017.

Chang, Y.-C., Roohi, N., and Gao, S. Neural lyapunov control. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 3245–3254, 2019.

Chow, Y., Ghavamzadeh, M., Janson, L., and Pavone, M. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

Chow, Y., Nachum, O., Faust, A., Duenez-Guzman, E., and Ghavamzadeh, M. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*, 2019.

Clark, A. Control barrier functions for complete and incomplete information stochastic systems. In *2019 American Control Conference (ACC)*, pp. 2928–2935. IEEE, 2019.

Clark, A. Control barrier functions for stochastic systems. *Automatica*, 130:109688, 2021.

Conley, C. and Smoller, J. A. Bifurcation and stability of stationary solutions of the fitz-hugh-nagumo equations. 1986.

Dawson, C., Qin, Z., Gao, S., and Fan, C. Safe nonlinear control using robust neural lyapunov-barrier functions. In *Conference on Robot Learning*, pp. 1724–1735. PMLR, 2022.

Deisenroth, M. P., Fox, D., and Rasmussen, C. E. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, 2013.

Fan, D. D., Nguyen, J., Thakker, R., Alatur, N., Aghamohammadi, A.-a., and Theodorou, E. A. Bayesian learning-based adaptive control for safety critical systems. In *2020 IEEE international conference on robotics and automation (ICRA)*, pp. 4093–4099. IEEE, 2020.

Filar, J. and Vrieze, K. *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

Garcıa, J. and Fernández, F. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Hutchinson, M. F. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Jankovic, M. Robust control barrier functions for constrained stabilization of nonlinear systems. *Automatica*, 96:359–367, 2018.

Kamthe, S. and Deisenroth, M. Data-efficient reinforcement learning with probabilistic model predictive control. In *International conference on artificial intelligence and statistics*, pp. 1701–1710. PMLR, 2018.

Keener, J. and Sneyd, J. Mathematical physiology, volume 8 of interdisciplinary applied mathematics. *Springer-Verlag, New York*, 200:400, 1998.

Kolter, J. Z. and Manek, G. Learning stable deep dynamics models. *Advances in Neural Information Processing Systems*, 32:11128–11136, 2019.

Kouzoupis, D., Frison, G., Zanelli, A., and Diehl, M. Recent advances in quadratic programming algorithms for nonlinear model predictive control. *Vietnam Journal of Mathematics*, 46(4):863–882, 2018.

Lechner, M., Žikelić, Đ., Chatterjee, K., and Henzinger, T. A. Stability verification in stochastic control systems via neural network supermartingales. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7326–7336, 2022.

Lewis, F. L., Vrabie, D., and Syrmos, V. L. *Optimal control*. John Wiley & Sons, 2012.

Liu, H.-T. D., Williams, F., Jacobson, A., Fidler, S., and Litany, O. Learning smooth neural functions via lipschitz regularization. *arXiv preprint arXiv:2202.08345*, 2022a.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022b.

Mao, X. *Stochastic differential equations and applications*. Elsevier, 2007.

Mazouz, R., Muvvala, K., Babu, A. R., Laurenti, L., and Lahijanian, M. Safety guarantees for neural network dynamic systems via stochastic barrier functions. In *Advances in Neural Information Processing Systems*.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Rajamani, R. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

Rosenblum, M., Pikovsky, A., Kurths, J., Schäfer, C., and Tass, P. A. Phase synchronization: from theory to data analysis. In *Handbook of biological physics*, volume 4, pp. 279–321. Elsevier, 2001.

Sarkar, M., Ghose, D., and Theodorou, E. A. High-relative degree stochastic control lyapunov and barrier functions. *arXiv preprint arXiv:2004.03856*, 2020.

Särkkä, S. and Solin, A. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Sauseng, P. and Klimesch, W. What does phase information of oscillatory brain activity tell us about cognitive processes? *Neuroscience & Biobehavioral Reviews*, 32 (5):1001–1013, 2008.

Schlaginhaufen, A., Wenk, P., Krause, A., and Dorfler, F. Learning stable deep dynamics models for partially observed or delayed dynamical systems. *Advances in Neural Information Processing Systems*, 34:11870–11882, 2021.

Song, Y., Garg, S., Shi, J., and Ermon, S. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pp. 574–584. PMLR, 2020.

Sootla, A., Cowen-Rivers, A. I., Jafferjee, T., Wang, Z., Mguni, D. H., Wang, J., and Ammar, H. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pp. 20423–20443. PMLR, 2022.

Strogatz, S. Sync: The emerging science of spontaneous order. 2004.

Taylor, A., Singletary, A., Yue, Y., and Ames, A. Learning for safety-critical control with control barrier functions. In *Learning for Dynamics and Control*, pp. 708–717. PMLR, 2020.

Taylor, A. J. and Ames, A. D. Adaptive safety with control barrier functions. In *2020 American Control Conference (ACC)*, pp. 1399–1405. IEEE, 2020.

Virmaux, A. and Scaman, K. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

Wang, L., Ames, A. D., and Egerstedt, M. Multi-objective compositions for collision-free connectivity maintenance in teams of mobile robots. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 2659–2664. IEEE, 2016.

Wehenkel, A. and Louppe, G. Unconstrained monotonic neural networks. *Advances in neural information processing systems*, 32, 2019.

Yoshida, Y. and Miyato, T. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

Zhang, J., Zhu, Q., and Lin, W. Neural stochastic control. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=5wI7gNopMHW.

Zhang, J., Zhu, Q., Yang, W., and Lin, W. Sync: Safety-aware neural control for stabilizing stochastic delay-differential equations. In *The eleventh international conference on learning representations*, 2023.

Zhang, J., Zhu, Q., and Lin, W. Learning hamiltonian neural koopman operator and simultaneously sustaining and discovering conservation laws. *Physical Review Research*, 6 (1):L012031, 2024.

Zhu, Q., Ma, H., and Lin, W. Detecting unstable periodic orbits based only on time series: When adaptive delayed feedback control meets reservoir computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(9), 2019.

A. Appendix

A.1. Proofs and Derivations

A.1.1. Notations and Preliminaries

In this section, we introduce some basic definitions and notations and then provide the proofs of the theoretical results.

Notations. Throughout the paper, we employ the following notation. Let $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ be a complete probability space with a filtration $\{\mathcal{F}_t\}_{t\geq 0}$ satisfying the usual conditions (i.e. it is increasing and right continuous while $\mathcal{F}_0$ contains all $\mathbb{P}$-null sets). If x, y are real numbers, then $x \wedge y$ denotes the minimum of x and y, $x \vee y$ denotes the maximum of x and y, $x \ll y$ denotes x is much smaller than b and $a \gg b$ vice versa. Let $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ be the inner product of vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$. For a second continuous function $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$, let ∇f denote the gradient of $f(\boldsymbol{x})$, that is, $\mathcal{H}f$ denote the Hessian matrix of f. For the two sets A, B, let $A \subset B$ denote that A is covered in B. Denote by $\log$ the base e logarithmic function. Denote by $\|\cdot\|$ the L^2-norm for any given vector in $\mathbb{R}^d$. Denote by $|\cdot|$ the absolute value of a scalar number or the modulus length of a complex number. For $A = (a_{ij})$, a matrix of dimension $d \times r$, denote by $\|A\|_{\mathrm{F}}^2 = \sum_{i=1}^{d}\sum_{j=1}^{r} a_{ij}^2$ the Frobenius norm.

Definition A.1 *(Martingale) The stochastic process X_t on $t \geq 0$ is called a martingale (submartingale) on probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ with a filtration $\{\mathcal{F}_t\}_{t\geq 0}$ satisfying the usual conditions, if the following two conditions hold:* (1) *X_t is $\mathcal{F}_t-$measurable for any $t \geq 0$;* (2) *$\mathbb{E}[X_t|\mathcal{F}_s] = X_s (\mathbb{E}[X_t|\mathcal{F}_s] = X_s)$ for any $t > s \geq 0$.*

Definition A.2 *(Stopping Time) Given probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ and a mapping $\tau : \Omega \to [0, \infty)$, we call τ an $\{\mathcal{F}_t\}_{t\geq 0}$ stopping time, for any $t \geq 0$, $\tau \leq t \in \mathcal{F}_t$,*

Definition A.3 *(Local Martingale) The stochastic process $X_t,\ t \geq 0$ is called a local martingale, if there exists a family of stopping times $\{\tau_n\}_{n\in\mathbb{Z}_+}$ such that $\lim_{n\to\infty} \tau_n = \infty,\ a.s.$ and $\{X_{t\wedge\tau_n}\}_{n\in\mathbb{Z}_+}$ is a martingale.*

Definition A.4 *(Itô's Process) Let B_t be a d-dimensional Brownian motion on probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$. A ($d$-dimensional) Itô's process is a stochastic process X_t on $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ in the form of*

$$X_t = X_0 + \int_0^t u(s,w)\mathrm{d}s + \int_0^t \mathrm{d}v(s,w)B_s \,(\Leftrightarrow \mathrm{d}X_t = u(t,w)\mathrm{d}t + v(t,w)\mathrm{d}B_t)\,,$$

where u and v satisfy the constraints as follows:

$$\mathbb{P}\left[\int_0^t \|v(s,w)\|^2\mathrm{d}s < \infty \text{ for all } t \geq 0\right] = 1,$$
$$\mathbb{P}\left[\int_0^t \|u(s,w)\|\mathrm{d}s < \infty \text{ for all } t \geq 0\right] = 1.$$

Definition A.5 *(Itô's Formula) Let X_t be a d-dimensional Itô's process given by*

$$\mathrm{d}X_t = u\mathrm{d}t + v\mathrm{d}B_t.$$

Let $f(t, \boldsymbol{x}) \in C^{1,2}([0,\infty) \times \mathbb{R}^d)$. Then, $Y_t = f(t, X_t)$ is an Itô's process as well, satisfying

$$\mathrm{d}Y_t = \frac{\partial h}{\partial t}(t, X_t)\mathrm{d}t + \nabla_{\boldsymbol{x}} f(t, X_t)\cdot \mathrm{d}X_t + \frac{1}{2}\mathrm{Tr}(\mathrm{d}X_t^\top \mathcal{H}f(t, X_t)\mathrm{d}X_t).$$

A.1.2. Proof of Theorem 3.1

To begin with, we check the inequality constraint in $\mathcal{U}_{es}(V, \mathcal{X})$ is satisfied by the projection element, that is

$$\mathcal{L}_{\boldsymbol{u}} V\big|_{\boldsymbol{u}=\hat{\pi}_{es}(\boldsymbol{u}, \mathcal{U}_{es}(V,\mathcal{X}))} \leq cV.$$

From the definition of the derivative operator, we have

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{u}} V\big|_{\boldsymbol{u}=\hat{\pi}_{es}(\boldsymbol{u}, \mathcal{U}_{es}(V,\mathcal{X}))} &= \nabla V \cdot (f + \boldsymbol{u} - \frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|^2}\cdot \nabla V) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H}V g\right] \\
&= \nabla V \cdot (f + \boldsymbol{u}) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H}V g\right] - \nabla V \cdot \frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|^2}\cdot \nabla V \\
&= \mathcal{L}_{\boldsymbol{u}} V - \max(0, \mathcal{L}_{\boldsymbol{u}} V - cV) \leq cV.
\end{aligned}$$

Next, we show the equivalent condition of the Lipschitz continuity of projection element. Notice that $\boldsymbol{u} \in \text{Lip}(\mathcal{X})$, then we have

$$\hat{\pi}_{es}(\boldsymbol{u}, \mathcal{U}_{es}(V, \mathcal{X})) \in \text{Lip}(\mathcal{X}), \iff \frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|^2} \cdot \nabla V \in \text{Lip}(\mathcal{X}).$$

Since $\|\frac{\nabla V}{\|\nabla V\|}\|$ is a continuous unit vector, and naturally is Lipschitz continuous, we only need to consider the remaining term $\frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|}$. According to the definition, all the functions occured in this term are continuous, so we only need to bound this term to obtain the global Lipschitz continuity, that is

$$\frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|} \in \text{Lip}(\mathcal{X}), \iff \sup_{\boldsymbol{x} \in \mathcal{X}} \frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|} < +\infty.$$

When $\mathcal{L}_{\boldsymbol{u}} V \leq cV$, obviously we have $\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV) = 0 < +\infty$, otherwise, since $V \geq \varepsilon \|\boldsymbol{x}\|^p$ and $c < 0$, we have

$$\mathcal{L}_{\boldsymbol{u}} V - cV \geq \mathcal{L}_{\boldsymbol{u}} V - c\varepsilon \|\boldsymbol{x}\|^p \approx \mathcal{O}(\|\boldsymbol{x}\|^p) \to \infty (\|\boldsymbol{x}\| \to \infty).$$

Thus, we have

$$\sup_{\boldsymbol{x} \in \mathcal{X}} \frac{\max(0, \mathcal{L}_{\boldsymbol{u}} V - cV)}{\|\nabla V\|} < +\infty \iff \sup_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\| < +\infty,$$

which completes the proof.

A.1.3. Proof of Example 3.2

Consider a 1-D SDE as follows:

$$\mathrm{d}x = ax\mathrm{d}t + bx\mathrm{d}B_t,$$

Then, according to the Ito formula, we have

$$\begin{aligned} \mathrm{d}\log x &= \frac{\mathrm{d}x}{x} - \frac{(\mathrm{d}x)^2}{2x^2} \\ &= a\mathrm{d}t + b\mathrm{d}B_t - \frac{b^2}{2}\mathrm{d}t. \end{aligned}$$

Integrate the both sides we have

$$\log x(t) = x(0) + ((a - \frac{b^2}{2})t + bB(t)).$$

Then we have

$$x(t) = x_0 \exp((a - \frac{b^2}{2})t + bB(t)),\ x_0 > 0.$$

Then for any $M > 0$, we have

$$x(t) > M, \iff x(0) \exp((a - \frac{b^2}{2})t + bB(t)), \iff B(t) > \frac{1}{b}(\log \frac{M}{x_0} - (a - \frac{b^2}{2})t).$$

Notice that $B(t) \sim \mathcal{N}(0, t)$ is a Gaussian noise, it naturally holds that

$$\mathbb{P}(B(t) > \frac{1}{b}(\log \frac{M}{x_0} - (a - \frac{b^2}{2})t)) > 0.$$

Next, for the exponential stability, let potential $V = \frac{1}{2}x^2$, then we have

$$\mathcal{L}V = (a + \frac{b^2}{2})x^2 = 2(a + \frac{b^2}{2})V.$$

Since $a < 0,\ |a| > \frac{1}{2}b^2$ and $x_0 > 0$, we have $a + \frac{b^2}{2} < 0$, then according to Theorem 2.1, the zero solution is exponential stable. The proof is completed.

A.1.4. PROOF OF THEOREM 4.1

Notice that each sample path of $\boldsymbol{x}(t)$ is continuous and $h(\boldsymbol{x})$ is also continuous. This implies that $h(\boldsymbol{x}(t)) > 0 \iff \boldsymbol{x}(t) \in \text{int}(\mathcal{C})$. Now we prove $h(\boldsymbol{x}(t)) > 0$ $a.s.$ with initial $h(\boldsymbol{x}(0)) > 0$, which is equivalent to $\tau = \infty$ $a.s.$, where stopping time $\tau = \inf\{t \geq 0 : h(\boldsymbol{x}(t)) = 0\}$. we prove it by contradiction. If $\tau = \infty$ $a.s.$ was false, then we can find a pair of constants $T > 0$ and $M \gg 1$ for $\mathbb{P}(B) > 0$, where

$$B = \{w \in \Omega : \tau < T \text{ and } \|\boldsymbol{x}(t)\| \leq M, \forall 0 \leq t \leq T\}.$$

But, by the standing hypotheses, there exists a positive constant K_M such that

$$\alpha(x) \leq K_M x,\ \forall |x| \leq \sup_{\|\boldsymbol{x}\| \leq M} h(\boldsymbol{x}) < \infty.$$

Then, for $w \in B$ and $t \leq T$,

$$\alpha(h(\boldsymbol{x}(t))) \leq K_M h(\boldsymbol{x}(t)).$$

Now, for any $\varepsilon \in (0, h(\boldsymbol{x}(0)))$, define the stopping time

$$\tau_\varepsilon = \inf\{t \geq 0 : h(\boldsymbol{x}(t)) \notin (\varepsilon, h(\boldsymbol{x}(0))\}.$$

By Itô's formula,

$$\begin{aligned} \mathrm{d}h(\boldsymbol{x}(t)) &= \mathcal{L}h\mathrm{d}t + \nabla h^\top g \mathrm{d}B_t \\ &\geq -\alpha(h)\mathrm{d}t + \nabla h^\top G \mathrm{d}B_t. \end{aligned}$$

Take expectation on both sides with respect to τ_ε on set B,

$$\begin{aligned} \mathbb{E}[h(\boldsymbol{x}(\tau_\varepsilon \wedge t))\mathbb{1}_B] &\geq h(\boldsymbol{x}(0)) - \int_0^t \mathbb{E}[\lambda(h(\boldsymbol{x}(\tau_\varepsilon \wedge s)))\mathbb{1}_B]\mathrm{d}s \\ &\geq h(\boldsymbol{x}(0)) - \int_0^t \mathbb{E}[K_M h(\boldsymbol{x}(\tau_\varepsilon \wedge s))\mathbb{1}_B]\mathrm{d}s. \end{aligned}$$

By Gronwall's inequality,

$$\mathbb{E}[h(\boldsymbol{x}(\tau_\varepsilon \wedge t))\mathbb{1}_B] \geq h(\boldsymbol{x}(0))e^{-K_M(\tau_\varepsilon \wedge t)\mathbb{P}(B)}.$$

Note that for $w \in B$, $\tau_\varepsilon \leq T$ and $h(\boldsymbol{x}(\tau_\varepsilon)) = \varepsilon$. The above inequality, therefore, implies that

$$\mathbb{E}[\varepsilon \mathbb{P}(B)] = \varepsilon \mathbb{P}(B) \geq h(\boldsymbol{x}(0))e^{-K_M(\tau_\varepsilon \wedge T)\mathbb{P}(B)} \geq h(\boldsymbol{x}(0))e^{-K_M T\mathbb{P}(B)}.$$

Letting $\varepsilon \to 0$ yields that $0 \geq h(\boldsymbol{x}(0))e^{-K_M T\mathbb{P}(B)}$, but this contradicts the definition of B and $\mathbb{P}(B) > 0$.
The proof is completed.

A.1.5. PROOF OF THEOREM 4.2

Similar to section A.1.2, we only have to check the safety constraint is satisfied by the projection element, that is

$$\mathcal{L}_{\boldsymbol{u}} h\big|_{\boldsymbol{u}=\hat{\pi}_{sf}(\boldsymbol{u}, \mathcal{U}_{sf}(\alpha, \mathcal{C}))} \geq -\alpha(h).$$

From the definitions we have

$$\begin{aligned} \mathcal{L}_{\boldsymbol{u}} h\big|_{\boldsymbol{u}=\hat{\pi}_{es}(\boldsymbol{u}, \mathcal{U}_{es}(V, \mathcal{X}))} &= \nabla h \cdot (f + \boldsymbol{u} + \frac{\max(0, -\mathcal{L}_{\boldsymbol{u}} h - \alpha(h))}{\|\nabla h\|^2} \cdot \nabla h) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H} h g\right] \\ &= \nabla h \cdot (f + \boldsymbol{u}) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H} h g\right] + \nabla h \cdot \frac{\max(0, -\mathcal{L}_{\boldsymbol{u}} h - \alpha(h))}{\|\nabla h\|^2} \cdot \nabla h \\ &= \mathcal{L}_{\boldsymbol{u}} h + \max(0, -\mathcal{L}_{\boldsymbol{u}} h - \alpha(h)) \geq -\alpha(h). \end{aligned}$$

Notice that the state space now is $\mathcal{C}$, which is bounded, then the proof is completed.

A.1.6. PROOF OF THEOREM 4.4

We prove there exists a controller $\boldsymbol{u}$ in $\mathcal{U}_{es}(V, \mathcal{C}) \cap \mathcal{U}_{sf}(\alpha, \mathcal{C})$ in two steps.

Step 1. To begin with, we consider the special case that $\mathbf{0} \in \arg\max_{\boldsymbol{x}\in\mathcal{C}} h(\boldsymbol{x})$ and $\mathbf{0}$ is the only maximum point in $\mathcal{C}$. Notice that

$$\{\boldsymbol{u} : \mathcal{L}_u h \geq -\alpha(h) + a\} \subset \{\boldsymbol{u} : \mathcal{L}_u h \geq -\alpha(h)\},\ \forall a > 0.$$

Thus, we choose $a = kh(\mathbf{0})$ with k to be defined, then we can get $\boldsymbol{u}_0 \in \{\boldsymbol{u} : \mathcal{L}_u h \geq -\alpha(h) + kh(\mathbf{0})\}$, s.t. $\boldsymbol{u}_0 \in \mathcal{U}_{sf}(\alpha, \mathcal{C})$. We now prove that there exists a potential function V s.t. $\boldsymbol{u}_0 \in \mathcal{U}_{es}(V, \mathcal{C})$. Since $\mathcal{C}$ is bounded, the class-$\mathcal{K}$ function α has linear bounds in $\mathcal{C}$ as

$$mx \leq \alpha(x) \leq qx,$$

for some $q > 0$. Then for $\boldsymbol{u}_0$, the following inequality holds,

$$\mathcal{L}_{\boldsymbol{u}_0} h \geq -qh + kh(\mathbf{0}).$$

Now let $k = q$, we have

$$\mathcal{L}_{\boldsymbol{u}_0} h \geq q(h(\mathbf{0}) - h).$$

Define $V(\boldsymbol{x}) = h(\mathbf{0}) - h(\boldsymbol{x}) \geq 0$, due to the continuity on the bounded region $\mathcal{C}$, there exists some positive number $l > 0$ s.t. $V(\boldsymbol{x}) \geq l\|\boldsymbol{x}\|$ in $\mathcal{C}$. Hence, V is a well-defined potential function. Notice that

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{u}_0} V &= \nabla V \cdot (f + \boldsymbol{u}_0) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H} h g\right], \\
\nabla V &= -\nabla h,\ \ \mathcal{H}V = -\mathcal{H}h.
\end{aligned}$$

We have

$$\mathcal{L}_{\boldsymbol{u}_0} V = -\mathcal{L}_{\boldsymbol{u}_0} h \leq -q(h(\mathbf{0}) - h) = -qV,$$

which means $\boldsymbol{u}_0 \in \mathcal{U}_{es}(V, \mathcal{C}) \cap \mathcal{U}_{sf}(\alpha, \mathcal{C})$.

Step 2. Now we consider the general case. Since $\mathbf{0} \in \mathrm{int}(\mathcal{C})$, and $\{h(\boldsymbol{x}) = 0\} = \partial\mathcal{C}$, there exists a small neighborhood $O(\mathbf{0}, \varepsilon),\ \varepsilon > 0$ of $\mathbf{0}$ s.t. $O(\mathbf{0}, \varepsilon) \subset \mathrm{int}(\mathcal{C})$. Then we can modify h on $O(\mathbf{0}, \varepsilon)$ without changing the safe region $\mathcal{C}$. Define the smooth approximation of the Dirac function $\delta_{\mathbf{0}}$ as

$$\lambda(\boldsymbol{x}) \triangleq \exp\big(-\frac{1}{\|\boldsymbol{x}\|^2 - \varepsilon}\big).$$

Since h is bounded on $\mathcal{C}$, we can always choose $M \gg 1$ s.t.

$$\mathbf{0} \in \arg\max_{\boldsymbol{x}\in\mathcal{C}}, \tilde{h},\ \tilde{h} = h + M\lambda.$$

According to Step 1, we can choose any $\boldsymbol{u} \in \{\mathcal{L}_{\boldsymbol{u}}\tilde{h} \geq -\alpha(\tilde{h}) + q\tilde{h}(\mathbf{0})\}$, and let $\tilde{V} = \tilde{h}(\mathbf{0}) - \tilde{h}$, s.t. $\boldsymbol{u} \in \mathcal{U}_{es}(\tilde{V}, \mathcal{C})$. Now we only need to show that there exists $\boldsymbol{u} \in \{\mathcal{L}_{\boldsymbol{u}}\tilde{h} \geq -\alpha(\tilde{h}) + q\tilde{h}(\mathbf{0})\} \cap \mathcal{U}_{sf}(\alpha, \mathcal{C})$. Notice that

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{u}}\tilde{h} &= \nabla(h + M\lambda) \cdot (f + u) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H}(h + M\lambda) g\right] \\
&= \left(\nabla h \cdot (f + u) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H} h g\right]\right) + M\left(\nabla\lambda \cdot (f + u) + \frac{1}{2}\mathrm{Tr}\left[g^\top \mathcal{H}\lambda g\right]\right) \\
&= \mathcal{L}_{\boldsymbol{u}} h + M\mathcal{L}_{\boldsymbol{u}}\lambda.
\end{aligned}$$

From the median theorem, we have

$$\alpha(\tilde{h}) = \alpha(h + M\lambda) = \alpha(h) + \alpha^{'}(\xi)M\lambda \leq \alpha(h) + LM\lambda,\ \xi \in (0, L),\ L = \max_{\boldsymbol{x}\in\mathcal{C}} h(\boldsymbol{x}).$$

Then we have

$$\begin{aligned}
&\mathcal{L}_{\boldsymbol{u}}\tilde{h} \geq -\alpha(\tilde{h}) + q\tilde{h}(\mathbf{0}) \geq -\alpha(h) - LM\lambda + q\tilde{h}(\mathbf{0}), \\
&\mathcal{L}_{\boldsymbol{u}}\tilde{h} \geq -\alpha(h) - LM\lambda + q\tilde{h}(\mathbf{0}),\ \Longleftrightarrow\ \mathcal{L}_{\boldsymbol{u}} h \geq -\alpha(h) - LM\lambda + q\tilde{h}(\mathbf{0}) - M\mathcal{L}_{\boldsymbol{u}}\lambda.
\end{aligned}$$

Let $N = \max(0, \sup_{\boldsymbol{x}\in O(\mathbf{0},\varepsilon)} -LM\lambda + q\tilde{h}(\mathbf{0}) - M\mathcal{L}_{\boldsymbol{u}}\lambda)$, then we have

$$\mathcal{L}_{\boldsymbol{u}}h \geq -\alpha(h) - LM\lambda + q\tilde{h}(\mathbf{0}) - M\mathcal{L}_{\boldsymbol{u}}\lambda \geq -\alpha(h) + N.$$

From step 1, we know that $\{\mathcal{L}_{\boldsymbol{u}}h \geq -\alpha(h) + N\} \subset \mathcal{U}_{sf}(\alpha, \mathcal{C})$, now we choose $\boldsymbol{u} \in \{\mathcal{L}_{\boldsymbol{u}}h \geq -\alpha(h) + N\}$ and complete the proof.

For the relaxation in Remark 4.5, we notice that when safe region is covered in unbounded cube as that in Remark 4.5, the safe function h only depends on the variables that locate in the bounded interval. Without loss of generality, we set the first j variables locate in bounded safe region. Then h only depends on $\boldsymbol{x} = (x_1, \cdots, x_j)^\top$ and is bounded in $[a_1, b_1] \times \cdots \times [a_j, b_j]$, hence there still exist $m,\ q$ s.t. $mx\alpha(x) \leq qx$ as required in **Step 1**. Then we repeat the rest of the above proof to complete the proof of Remark 4.5.

B. Experimental Configurations

In this section, we provide the detailed descriptions for the experimental configurations of the control problems in the main text. The computing device that we use for calculating our examples includes a single i7-10870 CPU with 16GB memory, and we train all the parameters with Adam optimizer. Our code is available at https://github.com/jingddong-zhang/FESSNC.

B.1. Neural Network Structures

- For constructing the potential function V, we utilize the ICNN as (Amos et al., 2017):

$$\begin{aligned}
\boldsymbol{z}_1 &= \sigma(W_0\boldsymbol{x} + b_0),\\
\boldsymbol{z}_{i+1} &= \sigma(U_i\boldsymbol{z}_i + W_i\boldsymbol{x} + b_i),\ i = 1, \cdots, k-1,\\
p(\boldsymbol{x}) &\equiv \boldsymbol{z}_k,\\
V(\boldsymbol{x}) &= \sigma(p(\boldsymbol{x}) - p(\mathbf{0})) + \varepsilon\|\boldsymbol{x}\|^2,
\end{aligned}$$

  where σ is the smoothed **ReLU** function as defined in the main text, $W_i \in \mathbb{R}^{h_i\times d},\ U_i \in (\mathbb{R}_+ \cup \{0\})^{h_i\times h_{i-1}}$, $\boldsymbol{x} \in \mathbb{R}^d$, and, for simplicity, this ICNN function is denoted by ICNN$(h_0, h_1, \cdots, h_{k-1})$;

- The class-$\mathcal{K}$ function α is constructed as:

$$\begin{aligned}
\boldsymbol{q}_1 &= \text{ReLU}(W_0 s + b_1),\\
\boldsymbol{q}_{i+1} &= \text{ReLU}(W_i\boldsymbol{q}_i + b_i), i = 1, \cdots, k-2,\\
\boldsymbol{q}_k &= \text{ELU}(W_{k-1}\boldsymbol{q}_{k-1} + b_{k-1}),\\
\alpha(x) &= \int_0^x q_k(s)\mathrm{d}s
\end{aligned}$$

  where $W_i \in \mathbb{R}^{h_{i+1}\times h_i}$, and this class-$\mathcal{K}$ function is denoted by $\mathcal{K}(h_0, h_1, \cdots, h_k)$;

- The neural control function (nonlinear version) is constructed as:

$$\begin{aligned}
\boldsymbol{z}_1 &= \mathcal{F}(W_0\boldsymbol{x} + B_1),\\
\boldsymbol{z}_{i+1} &= \mathcal{F}(W_i\boldsymbol{z}_i + b_i),\ i = 1, \cdots, k-1,\\
\mathbf{NN}(\boldsymbol{x}) &\equiv W_k\boldsymbol{z}_k,\\
\boldsymbol{u}(\boldsymbol{x}) &= \text{diag}(\boldsymbol{x})\mathbf{NN}(\boldsymbol{x}),
\end{aligned}$$

  where $\mathcal{F}(\cdot)$ is the activation function, $W_i \in \mathbb{R}^{h_{i+1}\times h_i}$, and this control function is denoted by Control$(h_0, h_1, \cdots, h_{k+1})$.

B.2. Double Pendulum

Here we model the state of the fully actuated noise-perturbed double pendulum as

$$
\begin{aligned}
\mathrm{d}\theta_1 &= z_1 \mathrm{d}t \\
\mathrm{d}z_1 &= \frac{m_2 g \sin(\theta_2)\cos(\theta_1-\theta_2) - m_2\sin(\theta_1-\theta_2)[l_1 z_1^2\cos(\theta_1-\theta_2) + l_2 z_2^2] - (m_1+m_2)g\sin(\theta_1)}{l_1[m_1 + m_2\sin^2(\theta_1-\theta_2)]}\mathrm{d}t + \sin(\theta_1)\mathrm{d}B_t \\
\mathrm{d}\theta_2(t) &= z_2 \mathrm{d}t, \\
\mathrm{d}z_2 &= \frac{(m_1+m_2)[l_1 z_1^2\sin(\theta_1-\theta_2) - g\sin(\theta_2) + g\sin(\theta_1)\cos(\theta_1-\theta_2)] + m_2 l_2 z_2^2\sin(\theta_1-\theta_2)\cos(\theta_1-\theta_2)}{l_2[m_1 + m_2\sin^2(\theta_1-\theta_2)]}\mathrm{d}t \\
&\quad + \sin(\theta_2)\mathrm{d}B_t.
\end{aligned}
$$

We transform the equilibrium to the zero by the coordinate transformation $\tilde{\theta}_{1,2} = \theta_{1,2} - \pi$. the The ZBF is $h(\theta_{1,2}, z_{1,2}) = \sin(\theta_1) + 0.5 \iff h(\tilde{\theta}_{1,2}, z_{1,2}) = 0.5 - \sin(\tilde{\theta}_1)$. Then we consider the dynamic for $\boldsymbol{x} = (\tilde{\theta}_{1,2}, z_{1,2})$ For training controller $\boldsymbol{u}$, we sample 500 data from the safe region $[-\pi/6 - \pi, 7\pi/6 - \pi] \times [-5, 5]^3$, we construct the NNs as follows.

FESSNC We parameterize $V(\boldsymbol{x})$ as ICNN$(4, 12, 12, 1)$, $\alpha(x)$ as $\mathcal{K}(1, 10, 10, 1)$, $\boldsymbol{u}(\boldsymbol{x})$ as Control$(4, 12, 12, 4)$. We set $\varepsilon =$ 1e-3, $c = -0.1$. We train the parameters with $\mathrm{lr} = 0.1$ for 300 steps.

GP-MPC Since we have known the dynamics, there is no need to fitting the dynamics as GP flows from data. We obtain the dynamics of moment $\boldsymbol{z} = (\mu, \Sigma)$ of distribution $p(\boldsymbol{x})$ as

$$
\begin{aligned}
\mu_{t+1} &= \mu_t + dt(\cdot f(\mu_t) + \boldsymbol{u}_t) \\
\Sigma_{t+1} &= \Sigma_t + dt \cdot g(\mu_t)
\end{aligned}
$$

where f, g are the drift and diffusion terms, respectively. We denote the above dynamic as

$$
\boldsymbol{z}_{t+1} = \tilde{f}(\boldsymbol{z}_t, \boldsymbol{u}_t).
$$

Then we solve the MPC problem as

$$
\begin{aligned}
\min_{\boldsymbol{u}_{0:k-1}} \Phi(\boldsymbol{z}_k) + \sum_{i=0}^{k-1} l(\boldsymbol{z}_i, u_i), \\
\text{s.t. } \boldsymbol{z}_{i+1} = \tilde{f}(\boldsymbol{z}_i, \boldsymbol{u}_i),\ \boldsymbol{z}_0 = (\boldsymbol{x}, \boldsymbol{0}),
\end{aligned}
$$

and apply the first control element $\boldsymbol{u}_0$ to the current dynamic. We use the PMP method as proposed in (Kamthe & Deisenroth, 2018) to solve the above problem. For safety and stability, we select cost function as

$$
\begin{aligned}
\Phi(\boldsymbol{z}) &= p_1\|\mu\|^2 + p_3\|\Sigma\|^2, \\
l(\boldsymbol{z}, \boldsymbol{u}) &= p_2 \mathrm{ReLU}(\sin(\tilde{\theta}_1) - 0.5) + \|\boldsymbol{z}\|^2.
\end{aligned}
$$

We set $p_1 = 6, p_2 = 2.5, p_3 = 0.5, k = 10$ and find the optimal control sequence via the gradient descent of the Hamiltonian function in PMP.

QP The object function in QP is set as

$$
\begin{aligned}
\min_{\boldsymbol{u}, d_1, d_2} \frac{1}{2}\|\boldsymbol{u}\|^2 + p_1 d_1^2 + p_2 d_2^2, \\
\text{s.t.} \mathcal{L}_{\boldsymbol{u}} V - \varepsilon V \le d_1, \\
\mathcal{L}_{\boldsymbol{u}} \frac{1}{h(\boldsymbol{x})} - \gamma h(\boldsymbol{x}) \le d_2,
\end{aligned}
$$

where $d_{1,2}$ are the relaxation numbers. We choose $V = \frac{1}{2}\|\boldsymbol{x}\|^2, p_1 = 10, p_2 = 10, \varepsilon = 0.5, \gamma = 2$.

We use Euler–Maruyama numerical scheme to simulate the system without and with control, and the random seeds are set as $\{1, 4, 6, 8, 9\}$.

B.3. Planar kinematic bicycle model

Here we model the state of the common noise-perturbed kinematic bicycle system as $\boldsymbol{x} = (x, y, \theta, v)^\top$, where x, y are the coordinate positions in phase plane, θ is the heading, v is the velocity. The dynamic is as follows,

$$\begin{aligned}
\mathrm{d}x(t) &= v(t)\cos\theta(t)\mathrm{d}t + x(t)\mathrm{d}B_t,\\
\mathrm{d}y(t) &= v(t)\sin\theta(t)\mathrm{d}t + y(t)\mathrm{d}B_t,\\
\mathrm{d}\theta(t) &= v(t)\mathrm{d}t,\\
\mathrm{d}v(t) &= (x(t)^2 + y(t)^2)\mathrm{d}t.
\end{aligned}$$

The ZBF is $h(x, y, \theta, v) = 2^2 - (x^2 + y^2)$. For training controller $\boldsymbol{u}$, we sample 500 data $(r\cos(w), r\sin(w), \theta, v)$ from the safe region $(r, w, \theta, v) \in [0, 3] \times [0, 2\pi] \times [-3, 3]^2$, we construct the NNs as follows.

FESSNC We parameterize $V(\boldsymbol{x})$ as ICNN$(4, 12, 12, 1)$, $\alpha(x)$ as $\mathcal{K}(1, 10, 10, 1)$, $\boldsymbol{u}(\boldsymbol{x})$ as Control$(4, 12, 12, 4)$. We set $\varepsilon =$ 1e-3, $c = -0.5$. We train the parameters with $\mathrm{lr} = 0.05$ for 500 steps.

GP-MPC Since we have known the dynamics, there is no need to fitting the dynamics as GP flows from data. We obtain the dynamics of moment $\boldsymbol{z} = (\mu, \Sigma)$ of distribution $p(\boldsymbol{x})$ as

$$\begin{aligned}
\mu_{t+1} =& \mu_t + dt(\cdot f(\mu_t) + \boldsymbol{u}_t),\\
\Sigma_{t+1} =& \Sigma_t + dt \cdot g(\mu_t),
\end{aligned}$$

where f, g are the drift and diffusion terms, respectively. We denote the above dynamic as

$$\boldsymbol{z}_{t+1} = \tilde{f}(\boldsymbol{z}_t, \boldsymbol{u}_t)$$

Than we solve the MPC problem as

$$\begin{aligned}
\min_{\boldsymbol{u}_{0:k-1}} \Phi(\boldsymbol{z}_k) + \sum_{i=0}^{k-1} l(\boldsymbol{z}_i, u_i),\\
\text{s.t. } \boldsymbol{z}_{i+1} = \tilde{f}(\boldsymbol{z}_i, \boldsymbol{u}_i),\ \boldsymbol{z}_0 = (\boldsymbol{x}, \boldsymbol{0}),
\end{aligned}$$

and apply the first control element $\boldsymbol{u}_0$ to the current dynamic. We use the PMP method as proposed in (Kamthe & Deisenroth, 2018) to solve the above problem. For safety and stability, we select cost function as

$$\begin{aligned}
\Phi(\boldsymbol{z}) =& p_1\|\mu\|^2 + p_3\|\Sigma\|^2,\\
l(\boldsymbol{z}, \boldsymbol{u}) =& p_2\mathrm{ReLU}(\boldsymbol{z}_x^2 + \boldsymbol{z}_y^2 - 2^2) + \|\boldsymbol{z}\|^2.
\end{aligned}$$

We set $p_1 = 5, p_2 = 2.5, p_3 = 0.5, k = 5$ and find the optimal control sequence via the gradient descent of the Hamiltonian function in PMP.

QP The object function in QP is set as

$$\begin{aligned}
\min_{\boldsymbol{u}, d_1, d_2} \frac{1}{2}\|\boldsymbol{u}\|^2 + p_1 d_1^2 + p_2 d_2^2,\\
\text{s.t.} \mathcal{L}_{\boldsymbol{u}} V - \varepsilon V \le d_1,\\
\mathcal{L}_{\boldsymbol{u}} \frac{1}{h(\boldsymbol{x})} - \gamma h(\boldsymbol{x}) \le d_2,
\end{aligned}$$

where $d_{1,2}$ are the relaxation numbers. We choose $V = \frac{1}{2}\|\boldsymbol{x}\|^2, p_1 = 10, p_2 = 10, \varepsilon = 0.5, \gamma = 2$.

We use Euler–Maruyama numerical scheme to simulate the system without and with control, and the random seeds are set as $\{3, 5, 6, 9, 10\}$.

B.3.1. FURTHER COMPARISON WITH LEARNING-BASED CONTROLLERS

We further compare the proposed FESSNC with the neural network based control methods to stabilize SDEs, we consider the SYNC (Zhang et al., 2023), NNDMC (Mazouz et al.) and RSMC (Lechner et al., 2022) methods. All these considered methods provide stability or safety guarantee for the controller based on the finite decomposition of the compact state space and a fine design of the loss function, which belong to the numerical guarantee and cannot assure the obtained controllers satisfy the stability or safety guarantee rigorously. The model details of the compared methods are listed as follows,

SYNC We use the grid discretization method with mesh size $r = 10$ to obtain 10^4 grid data in safe region $[-2, 2]^2 \times [-3, 3]^2$. We parameterize the controller as Control$(4, 12, 12, 4)$, the batch size is set as $N = 500$, we train the parameters with $\mathrm{lr} = 0.05$ for 500 steps with the same loss function as follows

$$
\begin{aligned}
L_{sf}(\boldsymbol{\theta}, \boldsymbol{\theta}_\alpha) &= \frac{1}{N}\sum_{i=1}^{N} \max\left\{0, -\mathcal{L}h(\boldsymbol{x}) - \alpha(h(\boldsymbol{x})) + 4Mr\right\}. \\
L_{st}(\boldsymbol{\theta}) &= \frac{1}{N}\sum_{i=1}^{N} \left[\max\left(0, (\alpha-2)\|{\boldsymbol{x}_i}^\top g(\boldsymbol{x}_i)\|^2 + \|\boldsymbol{x}_i\|^2(2\langle \boldsymbol{x}_i, f_{\boldsymbol{u}}(\boldsymbol{x}_i)\rangle + \|g(\boldsymbol{x}_i)\|_{\mathrm{F}}^2)\right)\right], \\
L &= L_{sf} + L_{st}
\end{aligned}
$$

Here $f_{\boldsymbol{u}}$ and g represent the controlled drift and diffusion terms, respectively. We set the stability strength as $\alpha = 0.8$, the Lipschitz constant of the corresponding functions in the safe region can be calculated as $M = 4\max q(x)$ where $\alpha(x) = \int_0^x q(z)\mathrm{d}z$.

NNDMC We first train the stabilization controller $\boldsymbol{u}_{st}$ with the same structure as that in FESSNC, then, we obtain the neural network dynamical model (NNDM) under control. In order to provide safety guarantee for the NNDM, we discretize the safe region to grid data as same in SYNC, then we use the `auto LiRPA` package to compute the linear lower bound $lb_q = (lb_{qx}, lb_{qy}, lb_{q\theta}, lb_{qv})^\top$ and upper bound $ub_q = (ub_{qx}, ub_{qy}, ub_{q\theta}, ub_{qv})^\top$ of the NNDM on each grid subspace $q \in Q$. Here Q is the set of all grid subspace. The input to the auto LiRPA model is the center point of the grid subspace, i.e., the center point of $(x, y, \theta, v) \in [a_1, b_1] \times [a_2, b_2] \times [a_3, b_3] \times [a_4, b_4]$ is $(\frac{a_1+b_1}{2}, \frac{a_2+b_2}{2}, \frac{a_3+b_3}{2}, \frac{a_4+b_4}{2})$. Since we know the safe function is this case, we do not have to solve the SOS problem to obtain the safe function and the minimizer of the safe function is $(0, 0, 0, 0)$. We then use the obtained lower and upper bound to solve the linear programming (LP) problem to obtain the safe controller $\boldsymbol{u}_{sf}$. Finally, the controller $\boldsymbol{u} = \boldsymbol{u}_{sf} + \boldsymbol{u}_{st}$ is applied to the original system. According to (Mazouz et al.), the LP problem is solved as follows,

$$
\begin{aligned}
\min_{a_1,a_2,a_3,a_4,u_{qx},u_{qy},u_{q\theta},u_{qv}} \quad & \sum_{i=1}^{4} a_i \\
\text{s.t.} \quad & -a_1 \le x \le a_1, \\
& -a_2 \le y \le a_2, \\
& -a_3 \le \theta \le a_3, \\
& -a_4 \le v \le a_4, \\
& lb_{qx} + u_{qx} \le x \le ub_{qx} + u_{qx}, \\
& lb_{qy} + u_{qy} \le y \le ub_{qy} + u_{qy}, \\
& lb_{q\theta} + u_{q\theta} \le \theta \le ub_{q\theta} + u_{q\theta}, \\
& lb_{qv} + u_{qv} \le v \le ub_{qv} + u_{qv}, \ \forall q \in Q.
\end{aligned}
$$

The final safe controller takes value as $\boldsymbol{u}(x, y, \theta, v) = (u_{qx}, u_{qy}, u_{q\theta}, u_{qv}),\ \forall (x, y, \theta, v)^\top \in q$.

RSMC According to (Lechner et al., 2022), the training data should be discretization of the compact state space, here we employ the same grid data as above. For the supermartingale function V, we parameterize it with the standard forward

neural network FNN$(4, 12, 12, 1)$. The training parameters are the same as above. The loss function is set as

$$
\begin{aligned}
L_{st} &= \frac{1}{N}\sum_{i=1}^{j}\left[\text{ReLU}\left(\frac{1}{m}\sum_{k=1}^{m}V(\boldsymbol{z}_i^k) - V(\boldsymbol{z}_i) + \tau K\right)\right] \\
\boldsymbol{z}_i^k &= \boldsymbol{z}_i + dt\cdot f_{\boldsymbol{u}}(\boldsymbol{z}_i) + \sqrt{dt}\xi_k g(\boldsymbol{z}_i),\ \boldsymbol{z} = (x, y, \theta, v)^\top,\ \xi_k \sim \mathcal{N}(0, I), \\
L_{lip} &= \text{ReLU}\left(\text{Lip}_V - \frac{\kappa}{\tau(\text{Lip}_f(\text{Lip}_{\boldsymbol{u}} + 1) + 1)}\right), \\
L &= L_{st} + L_{lip}.
\end{aligned}
$$

We set the tolerance error $\kappa = 0.1$, $\tau = 6/10$, $m = 5$ is the sample size to estimate the expectation, the Euler step size $dt = 0.01$, and the Lipschitz constants Lip_V, $\text{Lip}_{\boldsymbol{u}}$ and calculated by the method in (Goodfellow et al., 2014).

RSMC+ICNN We directly replace the FNN supermartingale V as ICNN$(4, 12, 12, 1)$.

We use Euler–Maruyama numerical scheme to simulate the system without and with control, and the random seeds are set as $\{3, 6, 9, 10, 11, 12, 14, 15, 16, 28\}$. The results are summarized in the Table 4. We train all the methods with `batch size` $= 500$ for 500 steps and calculate the training time. We test the learned controller on 10 sample temporal trajectories over 20 seconds. The results are summarized in the following table. The safety rate is the ratio of time that the controlled state stays in the safe region to total time. The success rate is the number of controlled states that satisfy the safety constraint and is closer than 0.1 to the target position for consecutive 2 seconds to 10. The control energy is calculated as $\int_0^T ||u(t)||^2 dt$. We notice that the RSMC failed in the task, but we could improve the performance of this method if we replace the original parameterized V function in [R3] with the ICNN constructed in our paper. We can see that our FESSNC achieves the best performance with rather low energy. For the computational complexity, d is the state dimension and k is the grid size for discretizing the state space needed in all the methods in [R1-R3]. The results show that only our FESSNC avoids the curse of dimensionality, and hence our FESSNC can scale to high-dimensional tasks. For the stability guarantee, only our method achieves exponential stability (ES) while the other methods achieve weaker asymptotic stability (AS). For the safety guarantee, our method holds almost surely (a.s.) for any controlled trajectory, that is $\mathbb{P}(x(t) \in \mathcal{C}, t \geq 0) = 1$, while the NNDMC method holds in probability, that is $\mathbb{P}(x(t) \in \mathcal{C}, 0 \leq t \leq N) > 1 - N\delta$ for some $\delta \in (0, 1)$. Last but not least, all the existing methods provide the numerical guarantee for the neural controller using the finite decomposition of the compact state space as the training data, but none of these methods can assure the stability or safety condition is rigorously satisfied after training. In contrast, we are the first to provide the theoretical stability and safety guarantee for the learned controller based on the analytical approximate projections.

Table 3. Performance and comparison of learning-based controllers.

Index	Method				
	FESSNC	SYNC	NNDMC	RSMC	RSMC+ICNN
Training time (sec)	**14**	125	209	435	987
Safety rate(%)	**100**	90	100	10	90
Success rate (%)	**100**	90	60	0	90
Control energy	**1.5**	1.8	2.6	14.5	1.8
Complexity	$\mathcal{O}(\mathbf{d})$	$\mathcal{O}(k^d d^2)$	$\mathcal{O}(k^d d^3)$	$\mathcal{O}(k^d)$	$\mathcal{O}(k^d)$
Stability guarantee (Type)	Yes (ES)	Yes (AS)	No	Yes (AS)	Yes (AS)
Safety guarantee (Type)	Yes (a.s.)	Yes (a.s.)	Yes (probability)	No	No
Type of guarantee	Theoretical	Numerical	Numerical	Numerical	Numerical

B.3.2. Ablation study

We further investigate the influence of the hyperparameters in our framework, including k, λ_1, λ_2 relating to the strength of exponential stability, the weight of stability loss, and the weight of safety loss, respectively. We train the FESSNC under different combinations of hyperparameters $\{k, \lambda_1, \lambda_2\} \in \{0.1, 0.5, 1.0\}^3$, then we test them using 10 sample trajectories.

We summarize the results in the following table, in which the Error is the minimum distance between the controlled bicycle to the target position, i.e., Error $= \min_t r(t) = ||(x(t), y(t))||$, the Std is the standard deviation of $r(t)$, and we use $\max r(t)$ to assess the risk to cross the boundary of safe region $r \leq 2$. The controllers under all the hyperparameter combinations perform fairly well, implying our method is not very sensitive to the selection of the hyperparameters.

Table 4. Performance and comparison of learning-based controllers.

		$\lambda_1 = 0.1$			$\lambda_1 = 0.5$			$\lambda_1 = 1.0$		
		$k = 0.1$	$k = 0.5$	$k = 1.0$	$k = 0.1$	$k = 0.5$	$k = 1.0$	$k = 0.1$	$k = 0.5$	$k = 1.0$
Error	$\lambda_2 = 0.1$	0.02	0.02	0.02	**4e-4**	0.01	0.05	0.018	0.05	0.01
	$\lambda_2 = 0.5$	0.11	0.03	0.03	**2e-3**	0.02	0.01	0.03	0.05	0.04
	$\lambda_2 = 1.0$	0.08	0.04	0.03	**8e-4**	0.01	0.06	0.03	0.01	0.07
Std	$\lambda_2 = 0.1$	0.06	0.38	0.35	**1e-3**	0.41	2.07	0.20	0.27	0.10
	$\lambda_2 = 0.5$	0.29	0.32	0.34	**2e-3**	0.04	0.05	0.17	0.68	0.10
	$\lambda_2 = 1.0$	0.29	0.58	0.38	**4e-3**	0.26	0.16	0.33	0.15	0.21
$\max r(t)$	$\lambda_2 = 0.1$	1.81	**1.73**	1.90	1.94	1.82	1.83	1.78	1.95	1.83
	$\lambda_2 = 0.5$	1.91	**1.66**	1.86	1.90	l1.77	1.83	1.76	1.87	1.81
	$\lambda_2 = 1.0$	1.72	1.70	1.78	1.88	1.89	1.77	**1.67**	1.80	1.77

B.4. Fitzhugh-Nagumo model

Consider the coupled FHN model $\boldsymbol{x}_{1:50}$ as

$$\mathrm{d}\boldsymbol{x}_i = f(\boldsymbol{x}_i)\mathrm{d}t + \sum_{j=1}^{N} g(\boldsymbol{x}_j)\mathrm{d}B_t$$

where

$$\begin{aligned} \boldsymbol{x}_i &= v_i, w_i, \\ f(\boldsymbol{x}_i) &= (v_i - \frac{v_i^3}{3} - w_i + 1, 0.1(v_i + 0.7 - 0.8w_i))^\top, \\ g(\boldsymbol{x}_i) &= (\frac{1}{3}v_i, 0)^\top \end{aligned}$$

Let $\tilde{\boldsymbol{x}} = (\tilde{\boldsymbol{x}}_1, \cdots, \tilde{\boldsymbol{x}}_{50})^\top = (\boldsymbol{x}_1 - \boldsymbol{s}, \cdots, \boldsymbol{x}_{50} - \boldsymbol{s})^\top$, where $\boldsymbol{s}$ is on the synchronization manifold s.t. $\dot{\boldsymbol{s}} = f(\boldsymbol{s})$, then we have the following variance equation

$$\mathrm{d}\delta = (I \otimes \nabla f)\delta \mathrm{d}t + (I \otimes \nabla g)\delta \mathrm{d}B_t.$$

We only need to stabilize the variance δ to zero, and the ZBF is set as $h(\delta) = 5^2 - \max_{1 \leq i \leq 50}(\tilde{v}_i^2, \tilde{w}_i^2)$. We parameterize $V(\boldsymbol{x})$ as ICNN$(100, 100, 100, 1)$, $\alpha(x)$ as $\mathcal{K}(1, 10, 10, 1)$, $\boldsymbol{u}(\boldsymbol{x})$ as Control$(100, 200, 200, 100)$. We set $\varepsilon =$ 1e-3, $c = -0.1$. We train the parameters with $\mathrm{lr} = 0.1$ for 300 steps. We use Euler–Maruyama numerical scheme to simulate the system without and with control, and the random seeds are set as $\{1, 4, 5, 9, 15\}$.

B.5. 3-link Pendulum

In this section, we test our framework on a complex 3-link planar pendulum, which possesses 6 state variables $(\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3)$ representing the 3 link angles and the 3 angle velocities of three joints. For sake of simplicity, we suppose the joints and links have unit mass m_i, unit length l_i, unit moment of inertia I_i, and unit relative position l_{ci} of the center of gravity. The corresponding controlled system under noise perturbed is:

$$\begin{aligned} \mathrm{d}\boldsymbol{x} &= \boldsymbol{y}\mathrm{d}t, \\ \mathrm{d}\boldsymbol{y} &= \left\{\boldsymbol{M}(\boldsymbol{x})^{-1}[-\boldsymbol{N}(\boldsymbol{x}, \boldsymbol{y})\boldsymbol{y} - \boldsymbol{Q}(\boldsymbol{x})] + \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{y})\right\}\mathrm{d}t + \boldsymbol{g}(\boldsymbol{x})\mathrm{d}B_2(t), \end{aligned}$$

where $\boldsymbol{x} = (\theta_1, \theta_2, \theta_3)^\top$, $\boldsymbol{y} = (\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3)^\top$, $\boldsymbol{M}, \boldsymbol{N} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{Q} \in \mathbb{R}^3$, $\boldsymbol{g}(\boldsymbol{x}) = (\sin(\theta_1), \sin(\theta_2), \sin(\theta_3))^\top$ with

$$M_{ij} = a_{ij}\cos(x_j - x_i),\ \ N_{ij} = -a_{ij}y_j\sin(x_j - x_i),\ \ Q_i = -b_i\sin(x_i),$$
$$a_{ii} = I_i + m_i l_{ci}^2 + l_i^2 \textstyle\sum_{k=i+1}^3 m_k,\ \ a_{ij} = a_{ji} = m_j l_i l_{cj} + l_i l_j \sum_{k=j+1}^3 m_k,$$
$$b_i = m_i l_{ci} + l_i \textstyle\sum_{k=i+1}^3 m_k.$$

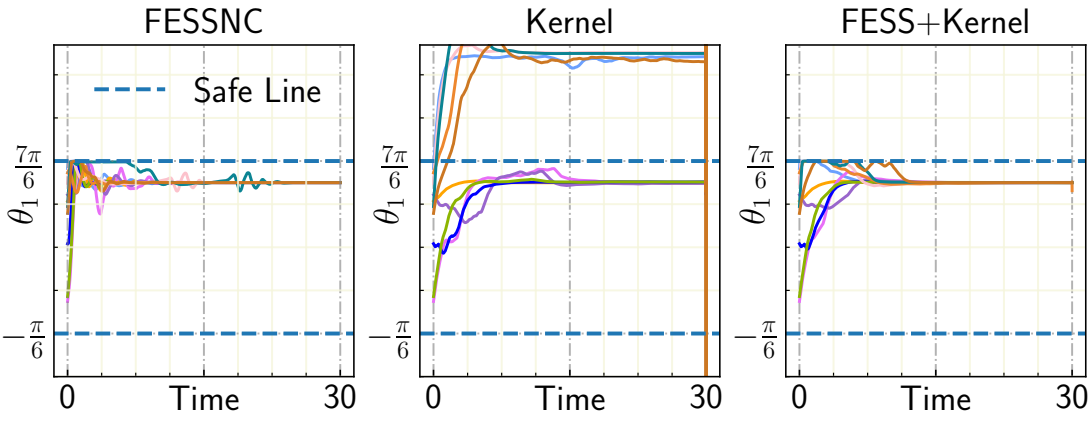

Figure 7. The angle of the first link $\theta_1(t)$ over 10 time trajectories. The green dashed lines represent the boundary of the safe region.

The mission is to stabilize the 3-link pendulum to the upright position $\theta_i = \pi,\ i = 1, 2, 3$ without crossing the safe region $\theta_1 \in (-\frac{\pi}{6}, \frac{7\pi}{6})$ (the same safe region as that for double pendulum in Section B.2). In addition to test the efficacy the proposed FESSNC, we also investigate the extension of our framework to nonparametric setting. We employ the kernel machine based controller as a benchmark of nonparametric controller. Specifically, we consider the implementation in Rectified Flow (Liu et al., 2022b) with Gaussian RBF kernel to design a model-based stabilization controller, named Kernel. This controller naturally lacks stability and safety guarantee, hence we apply our approximate projection operators to this controller to provide expected guarantees, the corresponding controller is dubbed as FESS+Kernel. The detailed model structures are set as follows,

We transform the equilibrium to the zero by the coordinate transformation $\tilde{\theta}_{1,2} = \theta_{1,2} - \pi$. the The ZBF is $h(\theta_{1,2}, z_{1,2}) = \sin(\theta_1) + 0.5 \iff h(\tilde{\theta}_{1,2}, z_{1,2}) = 0.5 - \sin(\tilde{\theta}_1)$. Then we consider the dynamic for $\boldsymbol{x} = (\tilde{\theta}_{1,2}, z_{1,2})$. For training controller $\boldsymbol{u}$, we sample 500 data from the safe region $[-\pi/6 - \pi, 7\pi/6 - \pi] \times [-5, 5]^3$, we construct the NNs as follows.

FESSNC We parameterize $V(\boldsymbol{x})$ as ICNN$(6, 12, 12, 1)$, $\alpha(x)$ as $\mathcal{K}(1, 10, 10, 1)$, $\boldsymbol{u}(\boldsymbol{x})$ as Control$(6, 18, 18, 6)$. We mask the output of the Control$(6, 18, 18, 6)$ with matrix $[0, 0, 0, 1, 1, 1]$ s.t. the control only acts on the $\boldsymbol{y}$ term. We set $\varepsilon =$ 1e-3, $c = -0.1$. We train the parameters with $\mathrm{lr} = 0.1$ for 300 steps.

Kernel We sample 10000 points from the safe region as the sample from initial distribution π_0, we set 10000 zero data as the samples of the target distribution π_1. We set the RBF kernel as $k(\boldsymbol{z}_1, \boldsymbol{z}_2) = \exp(-\|\boldsymbol{z}_1 - \boldsymbol{z}_2\|^2/h),\ \boldsymbol{z} = (\boldsymbol{x}^\top, \boldsymbol{y}^\top)^\top$ with bandwidth $h = 1e - 3$. To move the controlled system from π_0 to π_1, the controller is designed as,

$$\boldsymbol{u}(\boldsymbol{z}, t) = \mathbb{E}_{\tilde{\boldsymbol{z}}_0\sim\pi_0, \tilde{\boldsymbol{z}}_1\sim\pi_1}\left[\frac{\tilde{\boldsymbol{z}}_1 - \boldsymbol{z}}{1-t}\frac{k(\boldsymbol{z}(\tilde{t}), \boldsymbol{z})}{\mathbb{E}_{\tilde{\boldsymbol{z}}_0\sim\pi_0, \tilde{\boldsymbol{z}}_1\sim\pi_1}[k(\boldsymbol{z}(\tilde{t}), \boldsymbol{z})]}\right] - \boldsymbol{f}(\boldsymbol{z}).$$

Here we take the empirical expectation using samples $\tilde{\boldsymbol{z}}_0 \sim \pi_0$ and $\tilde{\boldsymbol{z}}_1 \sim \pi_1$, and we denote $\boldsymbol{f}(\boldsymbol{z})$ by the original drift term of the 3-link pendulum.

FESS+Kernel We directly apply our approximate projection operators in Eqs. (9)(6) to the Kernel controller, and denote the obtained controller as FESS+Kernel.

We test these controllers and summarize the results in Fig. 7. In can be seen that although the kernel machine based controller performs poorly in this task, our framework can successfully improve the Kernel controller with safety and stability guarantee.