Approximation of Separable Control Lyapunov Functions with Neural Networks ⋆

Mario Sperl [a], Jonas Mysliwitz [a], Lars Grüne [a]

[a] *Mathematical Institute, University of Bayreuth, Bayreuth, Germany*

Abstract

In this paper, we investigate the ability of neural networks to provide curse-of-dimensionality-free approximations of control Lyapunov functions. To achieve this, we first prove an error bound for the approximation of separable functions with neural networks. Subsequently, we discuss conditions on the existence of separable control Lyapunov functions, drawing upon tools from nonlinear control theory. This enables us to bridge the gap between neural networks and the approximation of control Lyapunov functions as we identify conditions that allow neural networks to effectively mitigate the curse of dimensionality when approximating control Lyapunov functions. Moreover, we present a network architecture and a training algorithm to illustrate the theoretical findings on a 10-dimensional control system.

Key words: control Lyapunov functions; neural networks; curse of dimensionality.

1 Introduction

Control Lyapunov functions (clfs) are a well-established tool in nonlinear control theory. They serve as a certificate of asymptotic null-controllability and can also be used to examine robustness against uncertainties and disturbances or to study performance criteria. However, their most common application lies in designing stabilizing feedback laws using the clf as guidance towards the equilibrium [5]. Given some asymptotically controllable system, we are thus interested in finding a corresponding clf. Since, in general, it is quite hard to compute clfs analytically, we rely on numerical methods. However, traditional numerical methods, which rely on a grid-based approach for the computation of the derivative of the clf, suffer from the curse of dimensionality. This means that, to achieve a certain accuracy, the number of required grid points and, thus, the numerical effort grows exponentially in the dimension of the state space. Consequently, such approaches become impractical in high dimensions.

This paper concerns the use of neural networks (NNs) to circumvent the curse of dimensionality for approximating clfs. Our approach is related to the work [39], which investigates structural properties on control systems that allow for an exact representation of a (possibly discontinuous) stabilizing feedback by NNs. Further, there exist several papers that present algorithms for the computation of clfs by NNs, see, e.g. [26,30]. However, while the algorithms therein have similarities with our numerical approach, none of them provides a complexity analysis regarding the curse of dimensionality. Establishing conditions for a curse-of-dimensionality-free approximation of clfs is the main contribution of this work. Addressing this challenge requires the identification of a suitable class of functions that can be approximated by NNs without suffering from the curse of dimensionality.

There exist various recent papers that discuss results regarding a curse-of-dimensionality-free approximation of solutions of particular kinds of partial differential equations, see, e.g., [4,13,17,21]. In particular, some of these references exploit the smoothness of solutions of 2nd order Hamilton-Jacobi-Bellman equations for a curse-of-dimensionality-free approximation to solve optimal control problems. However, when it comes to computing a clf for a deterministic system, which can be characterized as a solution of a particular first-order Hamilton-Jacobi-Bellman equation, we cannot expect such a level of smoothness. Thus, we rely on a different structural assumption that allows NNs to mitigate the curse of dimensionality. To this end, we consider so-called separable

⋆ A preliminary version of this paper was presented at the 12th IFAC Symposium on Nonlinear Control Systems NOLCOS 2022, see [19]. Corresponding author M. Sperl.

Email addresses: mario.sperl@uni-bayreuth.de (Mario Sperl), jonas.mysliwitz@uni-bayreuth.de (Jonas Mysliwitz), lars.gruene@uni-bayreuth.de (Lars Grüne).

functions. Informally speaking, a mapping is called separable if it can be written as a sum of functions that are each defined on some lower-dimensional domain. Separable functions fall into the class of compositional functions. The ability of NNs to avoid the curse of dimensionality for compositional functions has been discussed for instance in [11,23,35]. Compared to general compositional functions, separable functions have a simpler structure that allows for more precise estimates, while the classes of control systems admitting separable clfs are still non-trivial.

Contribution

In this paper, we bridge the gap between NN approximation theory and the computation of clfs via NNs. Based on [18], we provide complexity results regarding the approximation of separable functions. Next, we extend the results for Lyapunov functions in [18] to clfs. Specifically, we use methods from nonlinear control theory to identify conditions on the control system such that a separable clf exists. Additionally, we expand upon the discussions in [19] to explore achieving separability through a state space transformation. Overall, we identify scenarios where NNs can provably avoid the curse of dimensionality in the computation of clfs. Finally, we propose a network architecture and training algorithm. In this context, we would also like to mention those topics that are not part of this paper. While this paper provides an expressivity result and proposes a training algorithm, it does not delve into the analysis of the convergence of the training algorithm or the generalization properties of the NN. Regarding the last point, which is of high importance for practical usage, we would like to refer to the works [12,27,28], where methods to verify that the NN output satisfies the Lyapunov conditions have been developed, thus providing a tool to verify generalization properties. In particular, we would like to point out that [28] leverages a compositional structure of the control system for verification, aligning well with the use of separability for efficient representation discussed in this paper. Moreover, we only consider the case in which smooth clfs exist, which allows us to better focus on the main results of this paper. Non-smooth clfs will be addressed in future research.

Outline

The remainder of this paper is organized as follows: The problem formulation is introduced in the next section. Afterwards, we provide a complexity analysis regarding the approximation of separable functions with NNs. In Section 4 we focus on the existence of separable clfs. To this end, we first discuss the use of techniques from nonlinear control theory that lead to separability and then consider the existence of separable clfs after suitable state space transformations. In Section 5 we illustrate the theoretical findings on a 10-dimensional test case. Finally, Section 6 concludes the paper.

Notation

For $n \in \mathbb{N}$ we set $[n] := \{1, \dots, n\}$ and define $I_n \in \mathbb{R}^{n\times n}$ to be the identity matrix. Let $K \subset \mathbb{R}^n$ be some compact set. Then we denote the infinity norm for continuous functions f on K via $\|f\|_{\infty,K} := \sup_{x\in K}\|f(x)\|$. The symbol D is used to denote the classic differential operator. Moreover, for some multi-index $\alpha \in \mathbb{N}^n$ we use D_α to denote the higher-order partial derivative with respect to α. We make use of the comparison functions $\mathcal{K}$ and $\mathcal{K}_\infty$, where $\mathcal{K}$ denotes all continuous and strictly increasing functions $\gamma\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $\gamma(0) = 0$ and $\mathcal{K}_\infty$ comprises all $\mathcal{K}$-functions that satisfy $\lim_{r\to\infty} \gamma(r) = \infty$.

2 Problem formulation

We consider a control system of the form

$$\dot{x} = f(x, u), \tag{1}$$

where the right-hand side $f\colon \mathbb{R}^n \times U \to \mathbb{R}^n$ is continuous, locally Lipschitz in x, and has an equilibrium at 0, i.e., $f(0, 0) = 0$. The input set is denoted as $U \subset \mathbb{R}^m$ and the admissible control functions are given as the set of measurable and locally essentially bounded functions $u\colon \mathbb{R}_{\geq 0} \to U$. In order to avoid technicalities, we assume our system (1) to be defined on the whole domain $\mathbb{R}^n$. We are interested in stabilizing the system towards the origin. To this end, we assume the control system (1) to be asymptotically controllable. In [38, Theorem 2.5] it has been shown that asymptotic controllability is equivalent to the existence of a clf in the sense of Dini. However, in the scope of this paper, we will only consider the case where our control system (1) admits a continuously differentiable clf, where the Dini derivate equals the gradient. This allows us to ensure compatibility with some theorems from the literature cited in the subsequent sections and avoids distracting technical difficulties. The important case that no smooth clf exists will be investigated in future research, cf. Section 6.

Definition 1 *A continuously differentiable function $V\colon \mathbb{R}^n \to \mathbb{R}$ is called (smooth) control Lyapunov function (clf) for (1) if there exist $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and $\alpha_3 \in \mathcal{K}$ such that for $x \in \mathbb{R}^n$*

$$\alpha_1(\|x\|) \leq V(x) \leq \alpha_2(\|x\|), \tag{2a}$$
$$\inf_{u\in U} DV(x)f(x, u) \leq -\alpha_3(\|x\|). \tag{2b}$$

Note that the existence of $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ such that the two inequalities in (2a) hold is equivalent to the fact that V is positive-definite and radially unbounded, cf. Lemma

4.3 in [25]. The reason for our preference of using the $\mathcal{K}_\infty$ functions α_1 and α_2 is due to our numerical algorithm, see Section 5. Given the existence of a smooth clf, it is our objective to numerically compute an approximation thereof on a compact set $K \subset \mathbb{R}^n$ via NNs.

3 Neural networks approximating separable functions

3.1 Preliminaries on neural networks

From a mathematical point of view, a neural network (NN) is a mapping $x \mapsto W(x;\theta)$ that takes some input vector $x \in \mathbb{R}^n$ and processes it according to its parameters θ in order to return some output. In case of a feedforward network, the value of a neuron y_k^l in layer l with number k is determined via

$$y_k^l = \sigma^l\Big(\sum_{i=1}^{N_{l-1}} w_{k,i}^l y_i^{l-1} + b_k^l\Big), \tag{3}$$

where $w_{k,i}^l \in \mathbb{R}$, $1 \le i \le N_{l-1}$, are weight parameters, $b_k^l \in \mathbb{R}$ constitutes a bias term and $\sigma^l\colon \mathbb{R} \to \mathbb{R}$ is the activation function of layer l. Throughout this work we solely consider feedforward networks with a one-dimensional output $W(x;\theta) \in \mathbb{R}$ and the identity as activation function in the last layer. It has been shown in [10] that the set of functions given by a NN with one hidden layer and a continuous sigmoidal activation function is dense in $C([0,1]^n)$. Since we are interested in the numerical effort needed for approximating a clf, we need a quantitative version of an approximation theorem. To this end, we characterize the complexity of a NN by the number of neurons in its hidden layers. Further, for $p \in \mathbb{N}, r \in \mathbb{R}_{>0}$, and a compact set $K \subset \mathbb{R}^n$ we introduce the Sobolev-like space

$$W_{p,r}(K) := \left\{F \in C^p(K,\mathbb{R}) \;\middle|\; \|F\|_{W_p(K)} \le r\right\},$$

where $\|F\|_{W_p(K)} := \sum_{0\le|\alpha|\le p}\|D_\alpha F\|_{\infty,K}$.

Theorem 2 *Let $R \in \mathbb{R}_{>0}$ and $\sigma \in \mathcal{C}^\infty(\mathbb{R},\mathbb{R})$ be not a polynomial. Then for every $n \in \mathbb{N}$ there exists a constant $\mu_n > 0$ such that for all $M \in \mathbb{N}$ a NN $W(x;\theta)$ with one hidden layer consisting of M neurons and activation function $\sigma^1 = \sigma$ satisfies for all $F \in C^p(K,\mathbb{R})$*

$$\inf_\theta \|W(\cdot;\theta) - F(\cdot)\|_{\infty,K} \le \mu_n M^{-\frac{p}{n}} \widetilde{R}\|F\|_{W_p(K)},$$

where $K := [-R,R]^n$, $\widetilde{R} := \max\{R,1\}$.

Note that the constant μ_n depends on n but is independent of F and M. Theorem 2 has been derived in [31, Theorem 2.1] for the case $R = 1$ and its extension to $R \in \mathbb{R}_{>0}$ is proven in [22, Corollary 1]. The theorem has originally been stated for $\sigma \in \mathcal{C}^\infty(\mathbb{R},\mathbb{R})$ such that there exists $b \in \mathbb{R}$ with $\sigma^{(i)}(b) \neq 0$ for all $i \in \mathbb{N}$. It has been discussed in [35] that this condition is equivalent to σ not being a polynomial. We can conclude from Theorem 2 that the number of neurons needed to provide an approximation up to some accuracy $\varepsilon > 0$ is given by $M = \mathcal{O}(\varepsilon^{-\frac{n}{p}})$, which has been shown in [31] to be best possible. Thus, in general, NNs also suffer from the curse of dimensionality.

3.2 Mitigating the curse of dimensionality with neural networks

The central part of this section is a result showing that NNs are capable of mitigating the curse of dimensionality for so-called separable functions.

Definition 3 *Let $F \in \mathcal{C}^1(\mathbb{R}^n,\mathbb{R})$ and $d \in [n]$. Then F is called (strictly) d-separable if for some $s \in [n]$ there exist $d_1,\dots,d_s \in [d]$ and functions $F_1,\dots,F_s$ with $F_j \in \mathcal{C}^1(\mathbb{R}^{d_j},\mathbb{R})$, $1 \le j \le s$, such that for all $x \in \mathbb{R}^n$ it holds*

$$F(x) = \sum_{j=1}^{s} F_j(z_j), \tag{4}$$

where $z_j = (x_{k_{j-1}},\dots,x_{k_j-1})$ with $k_0 := 1$ and $k_j := k_{j-1} + d_j$, $j \in [s]$.

In other words, if F is a d-separable function, its domain can be split into s subspaces, having the form $\mathbb{R}^n = \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_s}$, such that F can be written as a sum of s functions, which are defined on the respective subspaces. For the purpose of this paper it is sufficient to consider *strictly* separable functions, which means that the intersection of two such subspaces is always the origin, i.e., the domains are not overlapping. Constructions with overlapping regions have been pursued, e.g., in [33,42]. For simplicity, we will omit the term "strictly" in what follows. The benefit of a separable structure can be exploited by a NN as shown in Figure 1. It consists of two hidden layers, where the first one has a linear activation $\sigma^1 = I_1$, and the second one uses some nonlinear activation function σ^2. The sublayers of the second hidden layer can be used to learn the functions F_j in (4).

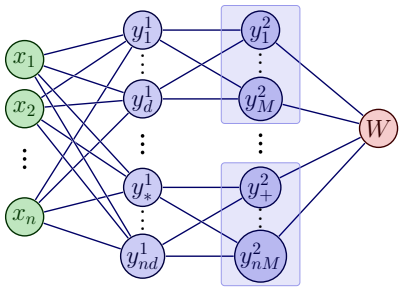


Fig. 1. Architecture of the NN with $* = (n-1)d+1$, $+ = (n-1)M+1$, and $W = W(x;\theta)$.

Theorem 4 *Let $d \in \mathbb{N}$, $r, R \in \mathbb{R}_{>0}$, and $\sigma \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{R})$ be not a polynomial. Define for $n \in \mathbb{N}$ the sets $K_n := [-R, R]^n$ and*

$$\mathcal{F}_{r,d}^{(n)} := \Big\{ F \in W_{1,r}(K_n) \;\Big|\; F \text{ is } d\text{-separable}, F(0) = 0 \Big\}.$$

Then there exists a constant $\mu_d > 0$ such that for all $n \in \mathbb{N}$ and $M \in \mathbb{N}$ the NN $W(x;\theta)$ depicted in Figure 1 with $n(d+M)$ neurons and activation functions $\sigma^1 = I$ and $\sigma^2 = \sigma$ satisfies for all $F \in \mathcal{F}_{r,d}^{(n)}$

$$\inf_\theta \|F(\cdot) - W(\cdot;\theta)\|_{\infty,K_n} \leq nr\mu_d \max\{R,1\} M^{-\frac{1}{d}}.$$

PROOF. By virtue of Theorem 2 there exists a constant $\mu_d > 0$ such that for all $M \in \mathbb{N}$ and $\widetilde{F} \in W_{1,r}(K_d)$ it holds that

$$\inf_{\tilde\theta} \|\widetilde{F}(\cdot) - \widetilde{W}(\cdot;\tilde\theta)\|_{\infty,K_d} \leq r\mu_d M^{-\frac{1}{d}} \max\{R,1\}, \quad (5)$$

where $\widetilde{W}(x;\theta)$ is a single-layer NN with activation function $\sigma^1 = \sigma$ and M neurons in its hidden layer. Now fix $n, M \in \mathbb{N}$ and some $F \in \mathcal{F}_{r,d}^{(n)}$. Since F is d-separable (see Definition 3), we can write $F(x) = \sum_{j=1}^s \tilde{F}_j(z_j)$ for some $s \in [n]$ and $F_j \in \mathcal{C}^1(\mathbb{R}^{d_j}, \mathbb{R})$. As $F(0) = 0$, we have $\sum_{j=1}^s \tilde{F}_j(0) = 0$. Thus, by defining $F_j(z_j) := \tilde{F}_j(z_j) - F_j(0)$, we can assume that $F_j(0) = 0$ for $j \in [s]$. This yields for $z_j \in \mathbb{R}^{d_j}$

$$F_j(z_j) = F_j(z_j) + \sum_{i \neq j} F_i(0) = F(0,\dots,0,z_j,0,\dots,0). \quad (6)$$

Further, observe that for $x \in \mathbb{R}^n$

$$DF(x) = \Big[DF_1(z_1) \; DF_2(z_2) \; \cdots \; DF_s(z_s) \Big]. \quad (7)$$

Consequently, we obtain from (6) and (7) for $j \in [s]$

$$\begin{aligned} \|F_j\|_{W_1(K_d)} &= \sum_{0 \leq |\alpha| \leq 1} \|D_\alpha F_j\|_{\infty,K_d} \\ &\leq \sum_{0 \leq |\alpha| \leq 1} \|D_\alpha F\|_{\infty,K_n} \leq r, \end{aligned}$$

whence $F_j \in W_{1,r}(K_d)$. Now we want to set the weights and biases corresponding to the first hidden layer of the network depicted in Figure 1 such that its first s sublayers contain the vectors z_j, $j \in [s]$, respectively. To this end, we set $b_k^1 = 0$ for $k \in [nd]$ and define $w_{i,i}^1 = 1$ for $i \in [d_1]$ in order to obtain the vector z_1 in the first sublayer, cf. (3). Next, we set $w_{n+i,d_1+i}^1 = 1$ for $i \in [d_2]$ to get z_2 in the second sublayer. We continue this procedure until the s-th sublayer. All remaining weights in the first hidden layer are set to 0. Furthermore, for the output layer we choose $w_{1,i}^3 = 1$ for $i \in [dM]$, $w_{1,i}^3 = 0$ for $i > dM$, and $b_1^3 = 0$. Observe that the output of the NN is now given as

$$W(x;\theta) = \sum_{j=1}^n \sum_{i=1}^M y_{(j-1)M+i}^2 = \sum_{j=1}^s \sum_{i=1}^M y_{(j-1)M+i}^2, \quad (8)$$

where for each $j \in [s]$ the output $\sum_{i=1}^M y_{(j-1)M+i}^2$ of the j-th sublayer can be interpreted as the output of a NN with input z_j and one hidden layer consisting of M neurons, cf. Figure 1. Let us denote the respective subnetwork by $W_j(z_j;\theta_j)$. By applying (5) we obtain for $j \in [s]$

$$\inf_{\theta_j} \|F_j(\cdot) - W_j(\cdot;\theta_j)\|_{\infty,K_d} \leq r\mu_d M^{-\frac{1}{d}} \max\{R,1\} =: \rho$$

Finally, invoking (8) gives us for $x \in K_n$

$$\begin{aligned} \inf_\theta \|F(x) - W(x;\theta)\| &= \inf_\theta \Big\| \sum_{j=1}^s F_j(z_j) - W_j(z_j;\theta_j) \Big\| \\ &\leq \sum_{j=1}^s \inf_{\theta_j} \|F_j(z_j) - W_j(z_j;\theta_j)\| \leq s\rho. \end{aligned}$$

Since $s \leq n$, this shows the claim. □

In the proof of Theorem 4, we have used the first (linear) layer of the network displayed in Figure 1 to compute the decomposition of the state x into vectors z_j, $1 \leq j \leq s$, according to Definition 3. We can thus identify the first layer with the mapping $x \mapsto W^1 x$, where $W^1 \in \mathbb{R}^{nd \times n}$ denotes the matrix that represents the corresponding decomposition of x. However, since the weights $w_{k,i}^1$, $k \in [nd]$, $i \in [n]$, can take on any real value, the first layer of the NN depicted in Figure 1 can in fact express any matrix $W^1 \in \mathbb{R}^{nd \times n}$. This observation motivates the following definition.

Definition 5 *Let $d \in [n]$, $F \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R})$, and $T \in \mathbb{R}^{n \times n}$ be invertible. Then F is called linearly d-separable with respect to T if the mapping $x \mapsto F(Tx)$ is (strictly) d-separable. Further, a function $G \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R}^l)$ is called linearly d-separable if each of its l component functions is linearly d-separable.*

Definition 5 extends the class of separable functions to all functions that are separable after a suitable linear transformation of the state space. The following corollary generalizes Theorem 4 to the case of linearly d-separable functions. To this end, for $c \in \mathbb{R}_{>0}$ we define GL_n^c as the space of invertible matrices $T \in \mathbb{R}^{n \times n}$ such that $\|T\|_\infty \leq c$ and $\|T^{-1}\|_\infty \leq c$. Note that after rescaling with $c/\|T\|_\infty$, every $T \in \mathbb{R}^{n \times n}$ with condition number $\leq c^2$ lies in GL_n^c.

Corollary 6 *Let $d \in \mathbb{N}$, $c, r, R \in \mathbb{R}_{>0}$, and $\sigma \in \mathcal{C}^\infty(\mathbb{R}, \mathbb{R})$ be not a polynomial. Define for $n \in \mathbb{N}$ the sets $K_n := [-R, R]^n$ and*

$$\mathcal{F}_{r,d,c}^{(n)} := \Big\{ F \in W_{1,r}(K_n) \;\Big|\; F \text{ is linearly } d\text{-separable w.r.t. some } T \in GL_n^c, F(0) = 0 \Big\}.$$

Then there exists a constant $\mu_d > 0$ such that for all $n \in \mathbb{N}$ and $M \in \mathbb{N}$ the NN $W(x;\theta)$ depicted in Figure 1 with $n(d+M)$ neurons and activation functions $\sigma^1 = I_1$ and $\sigma^2 = \sigma$ satisfies for all $F \in \mathcal{F}_{r,d,c}^{(n)}$

$$\inf_\theta \|F(\cdot) - W(\cdot;\theta)\|_{\infty,K} \leq cnr\mu_d \max\{cR, 1\} M^{-\frac{1}{d}}.$$

PROOF. Let $F \in \mathcal{F}_{r,d,c}^{(n)}$. Consider the mapping $G\colon T^{-1}K_n \to \mathbb{R}, x \mapsto F(Tx)$. By assumption, G is a d-separable function. Further, note that $G(0) = 0$ and $T^{-1}K_n \subset cK_n = [-cR, cR]^n$. Moreover, it holds that

$$\begin{aligned}
\|G\|_{W_1(T^{-1}K_n)} &= \sum_{0 \leq |\alpha| \leq 1} \|D_\alpha F(T\cdot)\|_{\infty, T^{-1}K_n} \\
&= \|F(T\cdot)\|_{\infty,T^{-1}K_n} + \sum_{j=1}^n \|\frac{\partial}{\partial x_j} F(T\cdot)\|_{\infty,T^{-1}K_n} \\
&\leq \|F\|_{\infty,K_n} + c\sum_{j=1}^n \|\frac{\partial}{\partial x_j} F\|_{\infty,K_n} \leq c\|F\|_{W_1(K_n)}.
\end{aligned}$$

Hence, applying Theorem 4 yields for $M \in \mathbb{N}$

$$\inf_\theta \|G(\cdot) - W(\cdot;\theta)\|_{\infty,T^{-1}K_n} \leq ncr\mu_d \max\{cR, 1\} M^{-\frac{1}{d}},$$

where $W(x;\theta)$ is the NN constructed in the proof of Theorem 4. Recall that the output of the first hidden layer is obtained as $W^1 x$ for some matrix $W^1 \in \mathbb{R}^{nd \times n}$. Hence, having $T^{-1}K_n$ as input space and the matrix W^1 representing the first hidden layer can equivalently be replaced by using K as input space and redefining $W^1 := W^1 T^{-1}$. With this, we obtain an approximation of F on K. Since the linear transformation of the weights in the first layer is already included in the infimum over all parameters θ, we obtain the claim. □

We would like to point out that Corollary 6 can also be proven by leveraging Theorem 4.10 in [23]. To this end, one represents the given separable function as a particular instance of a compositional function as defined in [23, Definition 3.1] and estimates the constants appearing in [23, Remark 4.12]. This approach results in estimates that are similar to the argumentation presented in the proofs of Theorem 4 and Corollary 6. As an immediate consequence of Corollary 6 we obtain that the number of neurons needed to approximate linearly d-separable functions grows only polynomially in the state dimension n.

Corollary 7 *Let $\varepsilon > 0$ and consider the setting from Corollary 6. Then for $n \in \mathbb{N}$ the number of neurons $N \in \mathbb{N}$ needed to ensure*

$$\sup_{F \in \mathcal{F}_{r,d,c}^{(n)}} \inf_\theta \|F(\cdot) - W(\cdot;\theta)\|_{\infty,[-R,R]^n} \leq \varepsilon$$

is given by $N = \mathcal{O}\big(nd + \frac{n^{d+1}}{\varepsilon^d}\big)$.

PROOF. Applying Corollary 6 yields

$$M \geq (n\, c\, r \mu_d \max\{cR, 1\})^d \varepsilon^{-d}.$$

The claim is obtained by counting the total number of neurons in the hidden layers in Figure 1. □

Remark 8 *In the setting of Theorem 4 as well as in the corollaries 6 and 7 we have worked with $\mathcal{C}^1$ target functions and thus applied Theorem 2 for the case $p = 1$. If one is in a position to apply Theorem 2 for higher values of p, one still obtains a number of neurons that is polynomially increasing in n, albeit with the lower exponent $d/p + 1$ in place of $d + 1$.*

4 Existence of separable control Lyapunov functions

In this section, we use of methods from nonlinear systems theory for providing conditions for the existence of (linearly) separable clfs. Thus, by invoking the results from Section 3 we can identify classes of systems that allow for a curse-of-dimensionality-free approximation of clfs by NNs.

4.1 Separability via small-gain theory and active nodes

This subsection proves the existence of separable clfs based on small gain theory, leveraging the notion of active nodes from [8,9]. We consider a control system (1) and assume that it can be decomposed into $s \in \mathbb{N}$ subsystems denoted by

$$\Sigma_j : \quad \dot{z_j} = f_j(x, \tilde{u}_j) = f_j(z_j, z_{-j}, \tilde{u}_j), \quad j \in [s], \quad (9)$$

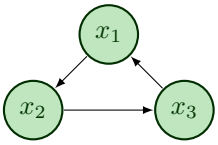
where

$$x = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_s \end{pmatrix}, \quad u = \begin{pmatrix} \tilde{u}_1 \\ \tilde{u}_2 \\ \vdots \\ \tilde{u}_s \end{pmatrix}, \quad f(x,u) = \begin{pmatrix} f_1(x, \tilde{u}_1) \\ f_2(x, \tilde{u}_2) \\ \vdots \\ f_s(x, \tilde{u}_s) \end{pmatrix}$$

with $z_j \in \mathbb{R}^{d_j}$, $U = U_1 \times U_2 \times \cdots \times U_s$, $\tilde{u}_j \in U_j$, $f_j \colon \mathbb{R}^n \times U_j \to \mathbb{R}^{d_j}$, and

$$z_{-j} := \Big(z_1, \dots, z_{j-1}, z_{j+1}, \dots, z_s\Big)^T \in \mathbb{R}^{n-d_j}.$$

We explicitly allow for the case that some subsystems Σ_j are independent of the control u, which corresponds to the case $U_j = \{0\}$. In the following, we investigate whether there exist functions V_j defined on the respective subspaces $\mathbb{R}^{d_j}$ such that their sum constitutes a clf for the whole system. To this end, in addition to a stability property for each subsystem, one needs to impose a condition on the coupling of the subsystems. For this purpose, we represent the decomposition as a directed graph that consists of s nodes. Each node belongs to one subsystem and there exists an edge from node i to node j, $j \neq i$, if the subsystem i influences the subsystem j, i.e., if the function f_j depends on the vector z_i. Figure 2 illustrates the graph corresponding to a decomposition into 1-dimensional subsystems of the control system (10) from Section 4 in [8].



$$\begin{aligned} \dot{x}_1 &= x_3 + u, \\ \dot{x}_2 &= x_1 - x_2 + x_1^2, \\ \dot{x}_3 &= x_2 - x_3. \end{aligned} \tag{10}$$

Fig. 2. A control system and its corresponding graph.

The following assumption imposes a stability condition on each subsystem:

Assumption 9 *For each $j \in [s]$ there exists a feedback function $F_j \colon \mathbb{R}^{d_j} \to U_j$, comparison functions $\alpha_j \in \mathcal{K}_\infty$, $\gamma_{i,j} \in \mathcal{K}_\infty$, $i \neq j$, as well as a positive-definite and radially unbounded function $V_j \in \mathcal{C}^1(\mathbb{R}^{d_j}, \mathbb{R})$ such that*

$$\begin{aligned} &DV_j(z_j) f_j(z_j, z_{-j}, F_j(z_j)) \\ &\leq -\alpha_j(V_j(z_j)) + \sum_{i \neq j} \gamma_{i,j}(V_i(z_i)). \end{aligned} \tag{11}$$

Note that for a subsystem Σ_j that is not influenced by the control, the left-hand side in (11) does not depend on any feedback function F_j. In particular, Assumption 9 states that for all $j \in [s]$, the function V_j is an ISS-Lyapunov function (see [41]) for the system

$$\dot{z_j} = f_j(z_j, z_{-j}, F_j(z_j)), \tag{12}$$

where z_{-j} is seen as the external input. Given such a stability assumption on each of the subsystems, small-gain theory can be used to obtain a stability property of the overall system, see, for instance, [14,29,36]. In the following, we focus on the theory developed in [9] that allows us to formulate a graph-based criterion regarding the existence of a separable clf. Note that we do not impose regularity conditions on F_j in Assumption 9 since this is not necessary in order to apply the results from [9], whereas regularity of F_j is of course required for the existence of solutions of the control system (12).

Definition 10 (cf. Definition 4 in [9]) *Let $j \in [s]$ and consider a subsystem Σ_j as in (9). The subsystem is called active if there exist $\bar{\alpha}_j$, $\gamma_{i,j} \in \mathcal{K}_\infty$, $i \neq j$, and a function $V_j \in \mathcal{C}^1(\mathbb{R}^{d_j}, \mathbb{R})$ such that for all $\alpha_j > \bar{\alpha}_j$ there exists $F_j \colon \mathbb{R}^{d_j} \to U_j$ such that (11) holds.*

Intuitively, Definition 10 implies that, for given gain functions $\gamma_{i,j}$, the rate of decrease of V_j along the direction of the vector field can be made as steep as desired by applying an appropriate feedback F_j. Using this notion of active subsystems (or active nodes) in the graph, the results of [9] yield the following proposition, where we call a tuple of nodes $(v_1, v_2, \dots, v_l)$, $l \geq 3$, cycle if there exists an edge from v_i to v_{i+1} for all $i \in [l-1]$, and $v_1 = v_l$, that is, the starting node equals the last node.

Proposition 11 *Consider a control system of the form (1) given through subsystems of the form (9) and let Assumption 9 hold. Moreover, assume that in each cycle of the directed graph corresponding to the decomposition (9) there is at least one active subsystem. Then there exists a d-separable clf for the system (1).*

PROOF. Let V_j, $j \in [s]$, denote the ISS-Lyapunov functions obtained from Assumption 9. Applying Theorem 4 and Theorem 5 in [9], respectively, yields the existence of continuous, positive definite functions $\lambda_j \colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, $j \in [s]$, such that

$$V(\cdot) := \sum_{j=0}^{s} \int_0^{V_j(\cdot)} \lambda_j(s)\, ds$$

is a Lyapunov function for

$$\dot{z_j} = f_j(z_j, z_{-j}, F_j(z_j)), \quad j \in [s].$$

This implies that V satisfies condition (2b), whence V is a clf for (1). This gives us the decomposition of V as d-separable function as in Definition 3. □

Revisiting the control system in (10), we can check that $V_j(x_j) = x_j^2$ is an ISS-Lyapunov function for each subsystem and that the first subsystem is active. Thus, Proposition 11 yields the existence of a 1-separable clf for Example 10. Overall, by invoking Corollary 7 we can conclude that Proposition 11 identifies a class of control systems, where a clf can be approximated by a NN without the curse of dimensionality.

4.2 Linear separability via linearization

In this subsection, the discussion of Subsection 4.1 is extended to the existence of *linearly* d-separable clfs according to Definition 5. We motivate this extension by considering a variation of (10), namely the control system

$$\begin{aligned} \dot{x}_1 &= x_3 + u, \\ \dot{x}_2 &= x_1 - x_2 + x_2^2, \\ \dot{x}_3 &= x_2 + x_3. \end{aligned} \tag{13}$$

Applying the backstepping procedure (cf. Section 6.1 in [37]) yields the existence of a clf for the system (13). However, the obtained clf is not 1-separable. In fact, it follows from Lemma 5 in [19] that there does not exist a 1-separable clf for (13). However, one can show that the control system falls into a class of systems that possess a *linearly* 1-separable clf, at least on some neighborhood of the origin. In order to formulate the corresponding assertion, we first show that stabilizable linear systems always admit a linearly 1-separable clf.

Proposition 12 *Consider a linear control system of the form*

$$\dot{x} = Ax + Bu, \tag{14}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. Assume that (A, B) is stabilizable. Then there exists a linearly 1-separable clf V for the system (14). The function V is quadratic, i.e., $V(x) = x^T Px$ for some $P \in \mathbb{R}^{n\times n}$ and satisfies

$$\inf_{u\in\mathbb{R}^m} DV(x)(Ax + Bu) \le DV(x)(A + BF)x \le -c\|x\|_2^2 \tag{15}$$

for a suitable feedback matrix $F \in \mathbb{R}^{n\times m}$ and a constant $c > 0$.

PROOF. It is known that for a linear and stabilizable system, there always exists a clf of the form $V(x) = x^T Px$ for a suitable symmetric and positive definite matrix $P \in \mathbb{R}^{n\times n}$. The inequality (15) then follows from the usual matrix Lyapunov inequality. As P is symmetric and positive-definite, there exists an orthogonal matrix $T \in \mathbb{R}^{n\times n}$ such that

$$\widetilde{P} := T^{-1}PT = T^T PT = \operatorname{diag}(p_1, \dots, p_n)$$

is a diagonal matrix. Thus,

$$V(Tx) = (Tx)^T P(Tx) = x^T \widetilde{P}x = \sum_{i=1}^n p_i x_i^2$$

is a 1-separable function. □

Proposition 12 implies that linearizable control systems locally possess linearly 1-separable clfs.

Corollary 13 *Consider a control system (1) with a $\mathcal{C}^1$-function f and assume that its linearization at the origin is stabilizable. Then the control system (1) possesses a linearly 1-separable clf on some neighborhood of the origin.*

PROOF. Write $f(x, u) = Ax + Bu + g(x, u)$ with

$$\lim_{\|(x,u)\|\to 0} \frac{\|g(x,u)\|}{\|(x,u)\|} = 0.$$

Since (A, B) is stabilizable, Proposition 12 yields the existence of $c \in \mathbb{R}_{\ge 0}$, $F \in \mathbb{R}^{n\times m}$, and a linearly 1-separable function $V(x) = x^T Px$ such that (15) holds. Following the proof of [40, Theorem 19], we obtain

$$\inf_u DV(x)f(x, u) \le -c\|x\|_2^2 + 2xPg(x, F(x)) < 0$$

for x sufficiently small, since $\frac{\|2xPg(x,F(x))\|}{\|x\|^2} \to 0$ for $x \to 0$. Hence, V is a clf for the nonlinear system (1) in a suitable neighborhood of the origin. □

4.3 Linear separability via feedback linearization

Next we explore a class of systems for which Proposition 12 can be employed to achieve linear separability through a potential nonlinear transformation. To this end, we extend the definition of feedback linearizability from [40, Section 5.3] to multi-input systems.

Definition 14 *An affine control system*

$$\dot{x} = f(x) + \sum_{j=1}^m g_j(x)u_j$$

with control input $u = (u_1, \dots, u_m)^T \in \mathbb{R}^m$ is called feedback linearizable, if there exists a diffeomorphism $S \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R}^n)$ as well as maps $a_j, b_j \colon \mathbb{R}^n \to \mathbb{R}$, $j \in [m]$, such that the transformed control system

$$\dot{\tilde{x}} = \tilde{f}(\tilde{x}) + \sum_{j=1}^m \tilde{g}_j(\tilde{x})v_j$$

with transformed state $\tilde{x} = S(x)$, new control input $v = (v_1, \dots, v_m)^T \in \mathbb{R}^m$ and

$$\begin{aligned} \tilde{f}(\tilde{x}) &= DS(x)\Big(f(x) + \sum_{j=1}^m a_j(x)g_j(x)\Big), \\ \tilde{g}_j(\tilde{x}) &= b_j(x)DS(x)g_j(x), \end{aligned}$$

is a linear control system, i.e., if there exist matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ such that $\tilde{f}(\tilde{x}) = A\tilde{x}$ and $(\tilde{g}_1(\tilde{x}), \dots, \tilde{g}_m(\tilde{x})) = B$ holds for all $\tilde{x} \in \mathbb{R}^n$.

Theorem 15 *Consider a feedback linearizable affine control system with transformation map S satisfying $S(0) = 0$, for which the pair (A, B) is stabilizable. Then the control system has a clf V of the form $V(x) = \widetilde{V}(S(x))$ with a linearly 1-separable function $\widetilde{V} : \mathbb{R}^n \to \mathbb{R}$.*

PROOF. According to Proposition 12, we have

$$\inf_{v\in\mathbb{R}^m} D\widetilde{V}(\tilde{x})(A\tilde{x}+Bv) \le D\widetilde{V}(\tilde{x})(A\tilde{x}+BF\tilde{x}) \le -c\|\tilde{x}\|_2^2$$

for suitable $c \in \mathbb{R}_{\ge 0}$, $F \in \mathbb{R}^{n\times m}$, and some linearly 1-separable mapping $\widetilde{V}$. For $V(x) = \widetilde{V}(S(x))$ and $u_j = a_j(x) + b_j(x)v_j$ we then obtain

$$\begin{aligned} & DV(x)\big(f(x) + \sum_{j=1}^m g_j(x)u_j\big) \\ =\ & D\widetilde{V}(S(x))DS(x)\big(f(x) + \sum_{j=1}^m g_j(x)(a_j(x)+b_j(x)v_j)\big) \\ =\ & D\widetilde{V}(\tilde{x})\big(\tilde{f}(\tilde{x}) + \sum_{j=1}^m \tilde{g}_j(\tilde{x})v_j\big) \ =\ D\widetilde{V}(\tilde{x})(A\tilde{x}+Bv). \end{aligned}$$

This implies

$$\inf_{u\in\mathbb{R}^m} DV(x)\Big(f(x) + \sum_{j=1}^m g_j(x)u_j\Big) \le -c\|S(x)\|_2^2.$$

Since S is a diffeomorphism with $S(0) = 0$, there exist α_1, $\alpha_2 \in \mathcal{K}_\infty$ with $\alpha_1(\|x\|_2) \le (\|S(x)\|_2) \le \alpha_2(\|x\|_2)$, see Lemma 1 in [24]. Thus, V satisfies all inequalities in (2), whence it is a clf. □

Corollary 16 *Consider the setting of Theorem 15 and assume that the transformation map S is linearly k-separable for some $k \in [n]$. Then the control system has a clf V that is a composition of a linearly 1-separable function with a linearly k-separable function.*

Note that Corollary 16 in particular applies to linear mappings S, as linear mappings are always 1-separable. We can conclude that for control systems that satisfy the requirements in Corollary 16 there exists a curse-of-dimensionality-free approximation with a NN that is built as in Figure 1, but has one additional hidden layer at the beginning, which is used to represent the k-separable transformation S.

5 Numerical illustration

5.1 Network structure and training algorithm

The structure of the NN that we use for the computation of a linearly separable clf is exactly the one depicted in Figure 1 with the modification of introducing a hyperparameter s for the number of sublayers, i.e., replacing the n sublayers in Figure 1 by s sublayers. An important feature of this network architecture is the fact that the decomposition of the state vector x into the vectors z_j, $1 \le j \le s$, is determined by the first hidden layer. Thus, the detection of a suitable splitting of the state space (see Definition 3) is part of the training process. This means that the numerical algorithm presented in this section does not need to know the splitting or coordinate transformation discussed in Section 4. Rather, this structure will be "learned" by the network in the training process.

It is possible to incorporate the linear transformation computed by the first hidden layer in Figure 1 into the second hidden layer, that is, to merge the two hidden layers into one hidden fully-connected layer. Since the NN in Figure 1 can be viewed as a fully-connected NN with some particular weights set to 0, a fully connected NN still preserves the property of mitigating the curse of dimensionality for separable clfs. However, in our numerical test cases, the NN with two hidden layers as depicted in Figure 1 frequently demonstrated an improved numerical performance. On the other hand, if no a priori estimates of the hyperparameters d and s are possible, the usage of a fully connected NN is more practical. A detailed comparison of these NN architectures, including different numbers of hidden layers, is of high importance but is deferred to future research due to space limitations.

In order to train the NN $x \mapsto W(x;\theta)$ towards a clf, we define a loss-function L that penalizes the violation of any of the three inequalities defining a smooth clf in Definition 1. For any point $x \in K$ we set

$$\begin{aligned} & L(x, W(x;\theta), DW(x;\theta)) := \\ & \quad ([W(x;\theta) - \alpha_1(\|x\|)]_-)^2 + ([W(x;\theta) - \alpha_2(\|x\|)]_+)^2 \\ & \quad + \eta\Big(\Big[\alpha_3(\|x\|) + \inf_{u\in U} DW(x;\theta)f(x,u)\Big]_+\Big)^2, \end{aligned} \tag{16}$$

where $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$, $\alpha_3 \in \mathcal{K}$, $[\cdot]_+ := \max(\cdot, 0)$, $[\cdot]_- := \min(\cdot, 0)$, and $\eta > 0$ is a weighting factor. Note that the functions α_1, α_2, and α_3, as well as the parameter η are hyperparameters of the algorithm. Their choice can significantly affect the training process, whence a system approach for selecting these hyperparameters is important and will be investigated in future research.

Note that L depends on the point x, the evaluation $W(x;\theta)$ and the orbital derivative $DW(x;\theta)f(x,u)$. We calculate this orbital derivative alongside the evaluation of $W(x;\theta)$ via automatic differentiation. This means that the orbital derivative is computed on the fly from the separable network, using the built-in differentiation via backpropagation in Tensorflow. Thus, the derivate does not need to be stored separately, whence separability of the orbital derivate, which cannot be expected

since f is not separable, is not needed. Moreover, we need to evaluate the expression $\inf_{u\in U} DW(x;\theta)f(x,u)$. This expression can be simplified for systems with $U = [-C,C]^m$ for some $C > 0$ and an affine linear control input of the form $\dot{x} = f(x,u) = h(x) + g(x)u$, since then we have

$$\begin{aligned}&\inf_{u\in U} DW(x;\theta)f(x,u)\\ =&DW(x;\theta)h(x) - C\|DW(x;\theta)g(x)\|_1,\end{aligned} \tag{17}$$

cf. [19, Lemma 6]. The training process of the NN is then performed by minimizing the value of the loss function (16) over a finite set of training data $\mathcal{D}_T \subset K$.

Remark 17 *Clfs can be characterized as solutions of Zubov's equation [6,20], i.e., as optimal value functions for suitable optimal control problems. For such problems, NN approaches have been proposed in the literature, see e.g. [3,27,34,44]. However, they are difficult to apply in our setting, because while we assume that a separable clf exists, we do not know its precise form and thus also not the corresponding optimal control problem.*

Furthermore, in our numerical tests it has turned out that the most significant error usually lies around the origin. We tackle this by adding the term $W(0;\theta)^2 + \|DW(0;\theta)\|^2$ to the loss-function used for the training of the network, cf. [7]. Adding these terms to the loss function encourages the used optimization routine to stay at $W(0;\theta) = 0$ and $DW(0;\theta) = 0$ during the training. While this approach produced the best results for us, different ways to address issues at the origin have successfully been implemented in the literature, for instance by transforming the NN output, cf. [15,32].

5.2 Numerical test case

Finally, we illustrate the presented algorithm on the following 10-dimensional control system

$$\dot{x} = f(x,u) = \begin{pmatrix} -x_1 + x_1x_2 - 0.1x_9^2\\ -x_2u_1\\ -x_3 + x_3x_4 - 0.1x_1^2\\ -x_4u_2\\ -x_5 + x_5x_6 + 0.1x_7^2\\ -x_6u_3\\ -x_7 + x_7x_8\\ -x_8u_4\\ -x_9 + x_9x_{10}\\ -x_{10}u_5 + 0.1x_2^2 \end{pmatrix} \tag{18}$$

with $U = [-1,1]^5$. It consists of 5 two-dimensional bilinear subsystems of the form $\dot{y} = -y + zy, \dot{z} = -uz$ coupled with small non-linearities. For $u = 1$ this recovers the ODE presented in [2], where it is shown that there does not exist a polynomial Lyapunov function for this system on $\mathbb{R}^2$. While there still exists a quadratic clf with appropriate coefficients on compact sets, enlarging the training domain makes it more difficult to recover it. This can cause the NN to defer from a quadratic influence of the variables, cf. Figure 3.

To illustrate the ability of our approach to determine subspaces that lead to separability, we consider the transformed system $\dot{x} = T^{-1}f(Tx,u)$, where $T = I_{10} + P \in \mathbb{R}^{10\times 10}$ with P being normally distributed around 0 with scale 0.1 leading to a condition number of $\|T\|\|T^{-1}\| \approx 2$ in the presented test case. Note that the subsystem that are computed during the training process are typically not the original subsystems from (18). We employed the hyperparameters $\alpha_1(r) = 0.5r^2, \alpha_2(r) = 10r^2, \alpha_3(r) = 0.01r^2$, as well as $d = 2$, $s = 5$, and $M = 64$ in a training process with $2e^5$ training data, a batch size of 64, and the softplus-function $\sigma(\cdot) = \ln(1+\exp(\cdot))$ as activation function in the second hidden layer. The training process was conducted to compute a clf on the domain $[-4,4]^{10}$, where we beforehand transformed $x \mapsto \frac{1}{4}x$ and performed the training on $[-1,1]^{10}$ for numerical reasons. Our computations are carried out with Python 3.10.6 and Tensorflow 2.11.0 (see [1]) on an NVIDIA GeForce RTX 3070 GPU. The optimization has been performed with the ADAM stochastic gradient descent method. After 30 epochs and a training time of 380 seconds, the algorithm reached an $\mathcal{L}_1$ error of $4.4e^{-5}$ in the training data. An evaluation at independently chosen $2e^5$ test data showed an $\mathcal{L}_1$ error of $1.1e^{-6}$ and an $\mathcal{L}_\infty$ error of $8.9e^{-2}$.

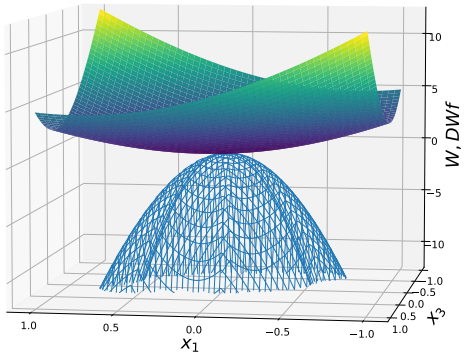


Fig. 3. Approximate clf (solid) and its corresponding orbital derivative (mesh) on the (x_1,x_3)-plane.

Figure 3 shows the computed NN output $W(x;\theta)$ as surface plot. Further, the directional derivative $DW(x;\theta)f(x,u^*)$ with

$$u^*(x) = \arg\min DW(x;\theta)f(x,u) \tag{19}$$

is calculated according to (17) and depicted as wireframe plot. Figure 4 depicts the evaluation of $W(x;\theta)$ alongside 10 trajectories with initial values randomly sampled

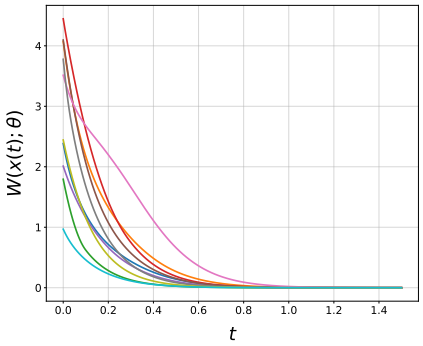


Fig. 4. Evaluation of $W(x;\theta)$ along trajectories.

in $[-0.5, 0.5]^{10}$ and control $u^*(x)$ as in (19). Note that the convergence of the trajectories in Figure 4 towards 0 as well as the plots in Figure 3 provide empirical evidence that the computed NN output might indeed by a clf. However, there is no formal guarantee that the Lyapunov conditions are met at every point. For verification techniques, we refer to the corresponding discussion in the introduction. Our Tensorflow code is available on `https://github.com/MarioSperl/SeparableCLF-NN`.

6 Conclusion

In this paper, we have discussed the capability of NNs to approximate clfs in high space dimensions. To this end, we have shown that NNs can mitigate the curse of dimensionality for approximating (linearly) separable functions and provided conditions for the existence of (linearly) separable clfs. Thus, we have identified control systems that allow for a curse-of-dimensionality-free approximation of clfs with NNs. Moreover, a numerical algorithm was presented and illustrated on two ten-dimensional control systems. For future research, we intend to systematically study the influence of the hyperparameters determining the NN architecture and the loss function. Afterwards, a comparison to other numerical methods is of interest, as it was for example done in [44]. Moreover, we aim to investigate the approximation of non-smooth clfs with NNs. A possible basis for this may be recent results on curse-of-dimensionality-free approximations of non-smooth compositional functions with ReLU NNs, see [16], which rely on the approximation result in [43].

Acknowledgements

The authors acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) in the frame of the priority programme SPP 2298 “Theoretical Foundations of Deep Learning” - project number 463912816.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] A. Ahmadi, M. Krstic, and P. Parrilo. A globally asymptotically stable polynomial vector field with no polynomial Lyapunov function. In *2011 IEEE 50th Conference on Decision and Control (CDC)*, pages 7579–7580. IEEE, 2011.

[3] G. Albi, S. Bicego, and D. Kalise. Gradient-augmented supervised learning of optimal feedback laws using state-dependent Riccati equations. *IEEE Control Syst. Lett.*, 6:836–841, 2022.

[4] C. Beck, A. Jentzen, K. Kleinberg, and T. Kruse. Nonlinear monte carlo methods with polynomial runtime for Bellman equations of discrete time high-dimensional stochastic optimal control problems. *preprint arXiv:2303.03390*, 2023.

[5] P. Braun, L. Grüne, and C. M. Kellett. *(In-) Stability of Differential Inclusions–Notions, Equivalences, and Lyapunov-like Characterizations*. SpringerBriefs in Mathematics. Springer, Cham, 2021.

[6] F. Camilli, L. Grüne, and F. Wirth. Control Lyapunov functions and Zubov's method. *SIAM J. Control Optim.*, 47(1):301–326, 2008.

[7] Y. Chang, N. Roohi, and S. Gao. Neural Lyapunov control. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[8] K. Chen and A. Astolfi. On the active nodes of network systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 5561–5566. IEEE, 2020.

[9] K. Chen and A. Astolfi. Active nodes of network systems with sum-type dissipation inequalities. *IEEE Trans. Automat. Control*, 69(6):3896–3911, 2024.

[10] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Systems*, 2(4):303–314, 1989.

[11] W. Dahmen. Compositional sparsity, approximation classes, and parametric transport equations. *preprint arXiv: 2207.06128*, 2023.

[12] H. Dai, B. Landry, S. Gao, L. Yang, M. Pavone, and R. Tedrake. Lyapunov-stable neural-network control. In *Proceedings of Robotics: Science and Systems*, 2021.

[13] J. Darbon, G. Langlois, and T. Meng. Overcoming the curse of dimensionality for some Hamilton-Jacobi partial differential equations via neural network architectures. *Res. Math. Sci.*, 7, 2020.

[14] S. Dashkovskiy, B. Rüffer, and F. Wirth. Small gain theorems for large scale systems and construction of ISS Lyapunov functions. *SIAM J. Control Optim.*, 48(6):4089–4118, 2010.

[15] N. Gaby, F. Zhang, and X. Ye. Lyapunov-net: A deep neural network architecture for Lyapunov function approximation. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2091–2096. IEEE, 2022.

[16] Q. Gong, W. Kang, and F. Fahroo. Approximation of compositional functions with ReLU neural networks. *Systems Control Lett.*, 175:105508, 2023.

[17] L. Gonon and C. Schwab. Deep ReLU neural networks overcome the curse of dimensionality for partial integrodifferential equations. *Anal. Appl.*, 21(01):1–47, 2023.

[18] L. Grüne. Computing Lyapunov functions using deep neural networks. *J. Comput. Dyn.*, 8(2):131–152, 2021.

[19] L. Grüne and M. Sperl. Examples for separable control Lyapunov functions and their neural network approximation. *IFAC-PapersOnLine*, 56(1):19–24, 2023.

[20] L. Grüne and F. Wirth. Computing control Lyapunov functions via a Zubov type algorithm. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 2129–2134. IEEE, 2000.

[21] M. Hutzenthaler, A. Jentzen, T. Kruse, T. Anh Nguyen, and P. von Wurstemberger. Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations. *Proceedings of the Royal Society A*, 476(2244):20190630, 2020.

[22] W. Kang and Q. Gong. Neural network approximations of compositional functions with applications to dynamical systems. *preprint arXiv:2012.01698*, 2020.

[23] W. Kang and Q. Gong. Feedforward neural networks and compositional functions with applications to dynamical systems. *SIAM J. Control Optim.*, 60(2):786–813, 2022.

[24] C. M. Kellett and P. M. Dower. Input-to-state stability, integral input-to-state stability, and $\mathcal{L}_2$-gain properties: Qualitative equivalences and interconnected systems. *IEEE Trans. Automat. Control*, 61(1):3–17, 2015.

[25] H. Khalil. *Nonlinear systems. Third Edition.* Prentice-Hall, Upper Saddle River, NJ, 2002.

[26] S. Khansari-Zadeh and A. Billard. Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems*, 62(6):752–765, 2014.

[27] J. Liu, Y. Meng, M. Fitzsimmons, and R. Zhou. Physics-informed neural network Lyapunov functions: PDE characterization, learning, and verification, 2023.

[28] J. Liu, Y. Meng, M. Fitzsimmons, and R. Zhou. Compositionally verifiable vector neural Lyapunov functions for stability analysis of interconnected nonlinear systems. *preprint arXiv:2403.10007*, 2024.

[29] T. Liu, D. Hill, and Z.-P. Jiang. Lyapunov formulation of ISS cyclic-small-gain in continuous-time dynamical networks. *Automatica J. IFAC*, 47(9):2088–2093, 2011.

[30] Y. Long and M. Bayoumi. Feedback stabilization: Control Lyapunov functions modelled by neural networks. In *Proceedings of 32nd IEEE Conference on Decision and Control*, pages 2812–2814. IEEE, 1993.

[31] H. Mhaskar. Neural networks for optimal approximation of smooth and analytic functions. *Neural Comput.*, 8(1):164–177, 1996.

[32] S. Mukherjee, J. Drgoňa, A. Tuor, M. Halappanavar, and D. Vrabie. Neural Lyapunov differentiable predictive control. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2097–2104. IEEE, 2022.

[33] S. Na, S. Shin, M. Anitescu, and V. M. Zavala. On the convergence of overlapping schwarz decomposition for nonlinear optimal control. *IEEE Trans. Autom. Control*, 67(11):5996–6011, 2022.

[34] T. Nakamura-Zimmerer, Q. Gong, and W. Kang. Neural network optimal feedback control with guaranteed local stability. *IEEE Open Journal of Control Systems*, 1:210–222, 2022.

[35] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, 2017.

[36] B. S. Rüffer. *Monotone dynamical systems, graphs, and stability of large-scale interconnected systems.* PhD thesis, Universität Bremen, 2007.

[37] R. Sepulchre, M. Janković, and P. Kokotović. *Constructive Nonlinear Control.* Springer, London, 1997.

[38] E. D. Sontag. A Lyapunov-like characterization of asymptotic controllability. *SIAM J. Control Optim.*, 21(3):462–471, 1983.

[39] E. D. Sontag. Feedback stabilization using two-hidden-layer nets. In *1991 American control conference*, pages 815–820. IEEE, 1991.

[40] E. D. Sontag. *Mathematical Control Theory.* Springer New York, 1998.

[41] E. D. Sontag and Y. Wang. On characterizations of the input-to-state stability property. *Systems Control Lett.*, 24(5):351–359, 1995.

[42] M. Sperl, L. Saluzzi, L. Grüne, and D. Kalise. Separable approximations of optimal value functions under a decaying sensitivity assumption. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 259–264. IEEE, 2023.

[43] D. Yarotsky. Optimal approximation of continuous functions by very deep ReLU networks. In *Conference on Learning Theory*, pages 639–649. PMLR, 2018.

[44] R. Zhou, M. Fitzsimmons, Y. Meng, and J. Liu. Physics-informed extreme learning machine lyapunov functions. *IEEE Control Systems Letters*, 8:1763–1768, 2024.