# Abstract

Neural networks are highly versatile and broadly applicable, often outperforming traditional methods in areas such as image processing in terms of accuracy, flexibility, and scalability. Despite these advantages, they are black-box models, lacking rigorous robustness and stability guarantees, which makes them prone to adversarial attacks. This limits the use of neural networks in safety-critical applications, such as autonomous driving or surgical robots. This doctoral thesis addresses this problem by developing system-theoretic methods for the robustness analysis of neural networks and the design of neural networks with robustness guarantees. In addition, we propose a stability analysis of control systems that include neural components such as neural network controllers.

**Robustness analysis of neural networks**

Neural networks are often susceptible to small changes in the input. The Lipschitz constant, a key measure of sensitivity to input variations, is widely used to evaluate the robustness of neural networks. However, determining the exact Lipschitz constant of a neural network is an NP-hard problem, necessitating the estimation of reasonably accurate upper bounds on the Lipschitz constant. In this thesis, we propose a method for Lipschitz constant estimation for general feedforward neural networks based on semidefinite programming (SDP). The method leverages the layer-wise structure of neural networks as well as the structure of the individual layers. To this end, we employ state space representations for convolutional layers, which significantly enhance the scalability of the method. Moreover, our approach encompasses a broad range of layer types, including general convolutional layers, diagonally repeated and gradient norm preserving activation functions, and pooling layers.

**Synthesis of robust neural networks**

In addition to the robustness analysis of neural networks, this dissertation deals with the training of robust neural networks, in particular the training of neural networks with a user-specified bound on the Lipschitz constant. To this end, we are the first to incorporate semidefinite constraints into the training problem. To address the resulting constrained optimization problem, we first apply the alternating direction method of multipliers, which is effective, but involves a significant computational overhead, as it requires solving an SDP at each training iteration. To mitigate this computational burden, we develop a more efficient interior point method for the training of Lipschitz-bounded neural networks, which scales to larger problems like the training of Wasserstein generative adversarial networks. To further leverage

existing first-order methods for the training, we finally propose a neural network parameterization that guarantees the satisfaction of linear matrix inequalities at all times during training. These matrix inequalities in turn ensure that the neural network complies with a specified Lipschitz bound.

**Stability analysis of closed-loop systems with neural components**

Thanks to their fast evaluation capabilities, neural network approximations can be used to replace costly controllers, such as optimization-based controllers, and thereby enable their use in real-time applications. In addition, neural networks are utilized to identify nonlinear systems or specific components within them. In this context, it is crucial to ensure the stability of the resulting closed-loop systems with neural network components. In this thesis, we consider closed-loop systems consisting of a linear time-invariant system in feedback with a neural network. The stability of these closed-loop systems is analyzed using robust control techniques leveraging SDPs. For this purpose, we use a set of dynamic integral quadratic constraints to describe the nonlinear activation functions, thereby reducing the conservatism of the analysis compared to approaches from the literature based on static quadratic constraints. In addition, we consider the problem of offset-free setpoint tracking using neural network controllers and propose global and local stability analyses for piecewise constant references. Finally, we propose the use of a reference governor to significantly expand the region of attraction.

The main goal of this doctoral thesis is to develop analysis and synthesis methods for robust neural networks and to provide closed-loop stability guarantees for systems incorporating neural components. To this end, we show that control tools provide scalable and accurate robustness certificates, such as the Lipschitz constant, for general feedforward neural networks. Furthermore, these methods facilitate the training of expressive, robust neural networks, and enable the safe use of neural networks in closed-loop systems.

# Chapter 1

# Introduction

## 1.1 Motivation

Deep learning architectures such as deep neural networks, convolutional neural networks, recurrent neural networks, and transformers have revolutionized numerous fields within engineering and computer science and are ubiquitous in our everyday lives [117]. Prominent applications of such neural networks include image and video processing tasks, natural language processing tasks, nonlinear system identification, learning-based control, and reinforcement learning [28, 51, 113, 125]. In these applications, neural networks offer advantages over traditional methods in terms of flexibility, accuracy, and scalability. This is due to their capability to handle large datasets and model highly complex nonlinear relationships as universal function approximators [53].

Despite their strengths and broad applicability, neural networks also face limitations. As black-box models, they generally lack robustness and stability guarantees, which restricts their use in safety-critical applications such as autonomous driving [137] and surgical robotics [134]. A significant vulnerability of neural networks is their sensitivity to small input perturbations [178]. This sensitivity can be quantified by the Lipschitz constant of the network's input-output mapping. A lower Lipschitz constant corresponds to higher robustness, as it ensures that small input changes result in proportionally small output variations. Consequently, Lipschitz constants of neural networks have generated considerable interest in the field of robust neural networks, inspiring a substantial body of research on Lipschitz constant estimation. Determining the exact Lipschitz constant is an NP-hard problem [107, 188]. Therefore, there is a demand for methods that provide accurate upper bounds for large-scale neural networks, see, e.g., [52, 69, 116, 188]. These Lipschitz bounds serve not only as standalone sensitivity measures but are also utilized to compute more sophisticated robustness measures such as certified robust accuracies [102, 103, 104, 122, 127].

While the problem of determining Lipschitz bounds enables robustness quantification of a given neural network after training, it naturally raises the question of designing robust neural networks. This can be achieved by regularizing or imposing guaranteed Lipschitz bounds during training [82]. Such robust neural networks, especially those with guaranteed Lipschitz bounds, are valuable in a range of applications. One example are generative adversarial networks (GANs), which can generate fake data that closely resemble real-world data [80]. Original GAN training faced

the issues of vanishing gradients, which prompted the development of Wasserstein GANs (WGANs) [11]. WGAN training optimizes over 1-Lipschitz neural networks to estimate the Wasserstein-1 distance. Furthermore, Lipschitz-bounded neural networks can enhance robustness and safety in reinforcement learning [18, 22, 106] and in neural network-based controller design with closed-loop stability guarantees [65, 173, 210].

Some control methods, such as model predictive control (MPC), rely on solving optimization problems at each time step. However, in real-time or resource-constrained applications, such optimization-based controllers may be too slow or computationally expensive. In such cases, neural networks can approximate and replace these controllers, providing a faster alternative [97, 115, 151, 179, 209]. In particular, explicit MPC, which precomputes control outputs for all polytopic regions in the state space, can be effectively approximated using neural networks with rectified linear unit (ReLU) activation functions [44, 58, 131, 170]. Although explicit MPC does not rely on an optimization problem at each time step, its computational complexity measured by the number of regions grows rapidly with the problem size and the number of constraints. Additionally, identifying the specific region containing the current system state may require too much processing power or memory storage for real-time evaluation. Furthermore, in the context of system identification, neural networks can be used to model nonlinear systems or specific nonlinear components within them [43, 47, 199]. However, when neural networks are employed to approximate control laws or identify nonlinear systems, performance and stability guarantees may not be preserved. To enable safe operation, it is crucial to establish closed-loop stability guarantees for feedback systems that incorporate neural components [32, PP1, 97, 99][1]. An alternative approach, proposed by [17, 75, 191], circumvents *a posteriori* testing and instead involves designing neural network controllers by learning directly over stable and robust closed-loop configurations that include neural network nonlinearities.

This motivates the purpose of this thesis: We analyze and design neural networks with robustness guarantees and verify stability of closed-loop systems that incorporate neural components using various control tools. Building on the above discussion, we derive accurate upper bounds on the Lipschitz constant for a general class of feedforward neural networks. Additionally, we propose and evaluate different training schemes designed to enforce upper bounds on the Lipschitz constant during training. Furthermore, we provide a stability analysis and an inner approximation of the region of attraction (ROA) for dynamical systems with neural network components. The subsequent section provides a detailed review of the literature relevant to this thesis, followed by a discussion of the key contributions and an outline of the structure of this thesis.

[1] Throughout this thesis, publications (co-)authored by the author of this thesis are marked with PP and listed separately in the bibliography.

## 1.2 Related work

In this section, we briefly survey selected topics on robust neural networks which are relevant for this thesis. Further details on the relation of our results to existing works are discussed in the respective sections later in the thesis.

### 1.2.1 Adversarial robustness

Adversarial robustness deals with the neural network's ability to withstand input attacks designed to deceive the neural network [178]. This led to the development of multiple adversarial attacks, see, e.g., [38, 132]. Adversarial attacks involve subtle modifications to the input data that are often imperceptible to humans but can significantly degrade the model's performance. In classification tasks, adversarially perturbed data cause misclassifications, as illustrated in Figure 1.1. Examples of imperceptible perturbations in image classification can be found in [81], while similar examples in the context of image segmentation for autonomous driving are presented in [137]. Robustness certification for neural networks largely focuses on analyzing attacks and defenses against perturbations, particularly those measured using the $\ell_2$ and $\ell_\infty$ norms [38]. The choice of norm depends on the application and the specific use case.

Defenses against adversarial attacks include adversarial training [81, 132, 171], which involves training the network on adversarial examples generated from the training data to increase robustness locally around the training samples. Another approach is the use of more complex architectures, such as ensemble methods [180]. Moreover, defensive distillation [150] entails transferring knowledge from a larger, more complex model to a smaller, simpler one. This approach enhances the robustness of the smaller model while preserving the performance of the original model. While these defenses mitigate adversarial attacks, they often lack rigorous robustness guarantees. Moreover, the introduction of a new defense has often been followed by a new, more powerful attack that breaks this defense method. For example, defensive distillation and adversarial training were shown to be ineffective against stronger attacks [37, 180].

Other approaches use robustness certificates for analysis and training of provably robust neural networks. Such certificates typically mean a certified robustness region for all training data points [198, 214]. For example, [94] derive lower bounds on the input perturbations required to change classifier decisions based on local Lipschitz constants and then introduce a regularizer to encourage robustness. Moreover, [158] derive a worst-case loss bound for some given dataset using a semidefinite relaxation and a given attack size and jointly optimize this bound with the neural network parameters and [132] solve a min-max optimization problem, maximizing over all adversaries from a set and minimizing the resulting worst-case loss. Local Lipschitz constants are also leveraged in [104] to train robust models through clipping of the activation functions. Additional robustness certification techniques use convex outer approximation [202], or interval bound propagation [83] to relax the activation functions, while [15] use robustness certificates to generate adversarial examples for adversarial training. More Lipschitz-based techniques such

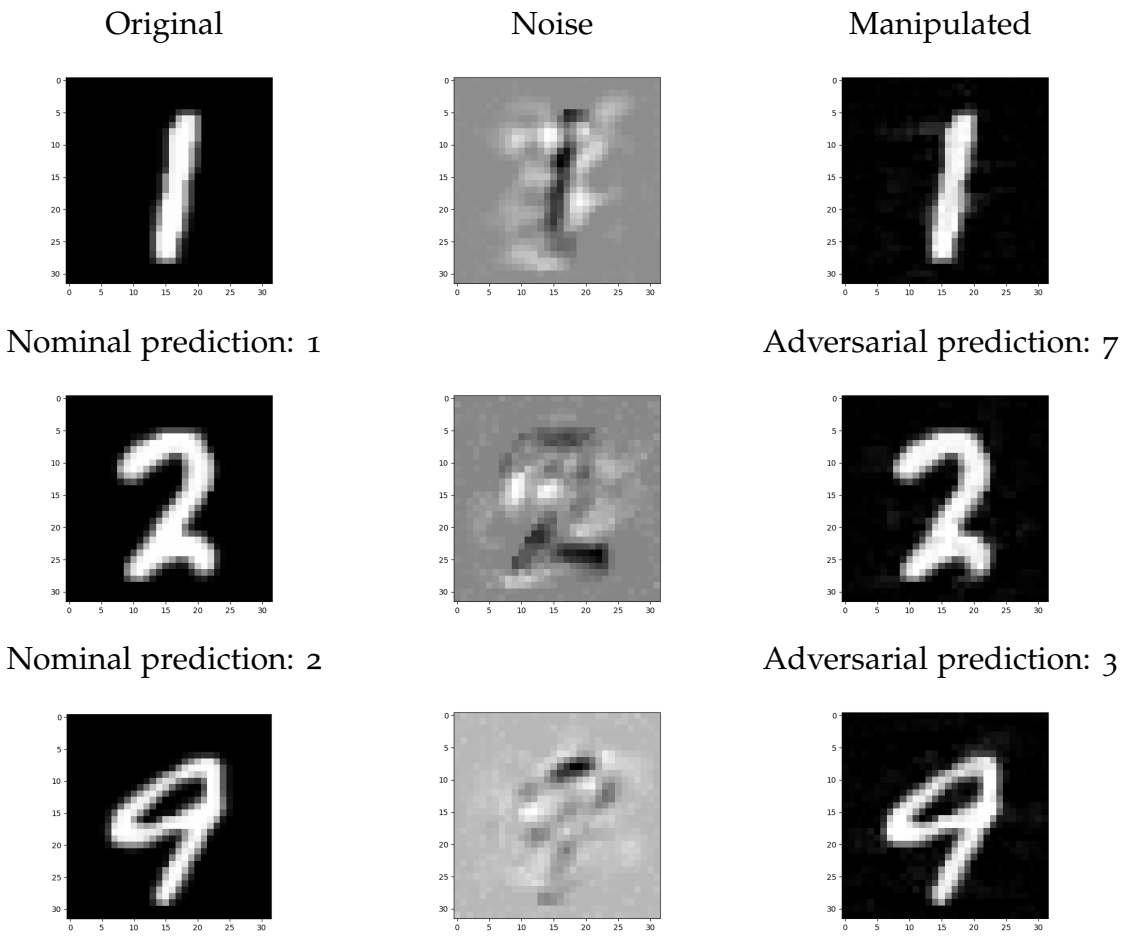


Figure 1.1: Adversarial attacks on images from MNIST dataset [56].

as Lipschitz Margin Training (LMT) [182] and Globally-Robust Neural Networks (GloRo) [103, 122] focus on bounding worst-case margins using Lipschitz constants to define robustness enhancing objectives. Another prominent method is randomized smoothing, which provides probabilistic robustness guarantees [49, 120, 154, 168]. This approach involves adding random noise to the input data and averaging over its predictions. Finally, [2] explore the interplay between neural network architectures and robustness, highlighting the role of architectural choices in enhancing model resilience.

### 1.2.2 Lipschitz constant estimation

It is well-established that determining the exact Lipschitz constant is NP-hard [107, 188]. While computationally inexpensive methods, such as using the product of spectral norms of weight matrices, provide upper bounds [178], these bounds are often overly conservative. As a result, significant effort has been devoted to developing techniques for efficiently computing tighter upper bounds.

Lipschitz constant estimation techniques include gradient approximation based methods by using the chain rule and automatic differentiation [188]. A global Lipschitz upper bound is obtained by optimizing over all activation patterns, i.e., all combinations of active/inactive neurons, including those that do not occur [197]. This problem can be further relaxed using sparse polynomial optimization [116]. Additionally, the structure and properties of the neural network and its components are often identified and relaxed. For example, [52] propose an approach for Lipschitz constant estimation with exponential scalability that treats activation functions as non-expansive averaged operators. In contrast, [69] introduce a semidefinite programming (SDP) based method named LipSDP that scales polynomially, using the property of *slope restriction* satisfied by common activation functions, e.g., ReLU and tanh. These activation functions are monotonically increasing functions with bounded slopes. They can be captured using quadratic constraints [68], which is the key ingredient to derive an SDP for accurate Lipschitz constant estimation [69]. An SDP is a convex optimization problem with an objective function that is linear in the decision variables subject to linear matrix inequality (LMI) constraints [184].

While many works focus on global Lipschitz bounds, e.g., [52, 69], in practice, local Lipschitz constants that hold on a problem-specific region are often significantly tighter than global ones [94]. RecurJac [215] and FastLip [198] compute local Lipschitz bounds by bounding the entries of the gradient of the Jacobian. For ReLU networks, local Lipschitz bounds can be determined by solving a mixed integer linear programming (MILP) problem [107] or using local quadratic constraints [92]. In this context, [26, 107] present approaches to exactly compute the Lipschitz constant of ReLU networks based on an MILP. These MILP-based approaches are, however, inherently limited in scalability.

The spectral norm product mentioned above [178] is computationally inexpensive and has better scalability properties than optimization-based methods. To compute upper bounds in this way, layer-wise Lipschitz constants are oftentimes determined using the power iteration method [78], which is applicable to any linear layer type, including fully connected and convolutional layers. Efficient computations of Lip-

schitz constants for convolutional layers are further discussed in [10, 14, 55, 218]. Another optimization-free method for Lipschitz constant estimation based on loop transformation and power iteration was recently presented in [67].

### 1.2.3 Training of Lipschitz-bounded neural networks

Several approaches have been developed to ensure that the Lipschitz constant of a neural network remains small during training. Some methods achieve this by incorporating regularization terms on the spectral norms of the network weights [72, 82] or on a data-based estimate of the actual Lipschitz constant [33]. In the context of regression tasks, approaches such as nonlinear set membership prediction, kinky inference [35], and Lipschitz interpolation [36, 130] provide guaranteed and optimized upper bounds on the Lipschitz constant.

Besides these approaches, there exist many works that impose a 1-Lipschitz constraint on all layers, e.g., [7, 48, 66, 139, 157, 204]. The feedforward composition of these 1-Lipschitz layers remains 1-Lipschitz, provided 1-Lipschitz activation functions are used. This follows directly from the calculation of the product of the layer-wise Lipschitz constants, which yields the end-to-end Lipschitz bound equal to one. This can be realized by spectral normalization [48, 66, 139, 157]. Alternatively, [7, 204] parameterize all layers to be orthogonal, i.e., constrain all singular values to be equal to one, and consequently also to be 1-Lipschitz, by an orthogonal parameterization. In this respect, the parameterization of orthogonal convolutional layers [124, 174, 181, 190, 211] is significantly more involved than the one of fully connected layers.

In addition to the gradient norm preserving linear layers obtained by orthogonalization, [7] and [175] suggest gradient norm preserving activation functions, namely GroupSort and Householder activation functions. These activation functions in combination with orthogonal linear layers yield gradient norm preserving neural networks that do not suffer from vanishing or exploding gradients [7]. In [91, 98], proximal neural networks consisting of averaged, hence 1-Lipschitz, operators are suggested. The linear operators of parseval proximal neural networks [91] are furthermore in a Stiefel manifold. In [24, 122], the use of robustness promoting loss functions is suggested and [13, 63] optimize over activation functions consisting of learnable 1-Lipschitz linear splines to improve expressivity while maintaining a Lipschitz bound. In a similar spirit, [138] suggest convex-potential layers, 1-Lipschitz layers derived from a discretization of continuous dynamical systems, and [9] propose SDP-based Lipschitz layers that by design satisfy layer-wise Lipschitz conditions adopted from an SDP. Their approach further embeds previous methods including spectral normalization into the SDP-based framework as special cases. An alternate method that is also based on SDP constraints uses so-called direct parameterizations for equilibrium networks [161], recurrent equilibrium networks [162], and feedforward neural networks [193], respectively. Neural networks parameterized by direct parameterization satisfy the underlying LMI constraints by design.

As mentioned before, one interesting application of 1-Lipschitz neural networks is for the parameterization of the discriminator network of a WGAN. In this appli-

cation, the Lipschitz bound of 1 was first realized by weight clipping [11], which is highly conservative, resulting in neural networks with unnecessarily low Lipschitz constants. A gradient penalty method was suggested in [89] that in turn provides no guarantees and, in [139], spectral normalization is applied to the training of GANs.

### 1.2.4 Quadratic constraints for neural network analysis and design

The feedback interconnection of a linear time-invariant (LTI) system and a diagonal static sector-bounded nonlinearity is called Lur'e system [128] and the problem of determining the stability of a Lur'e system for any nonlinearity within the defined sector bounds is called the problem of absolute stability, see [129] and references therein. In the 1960s, many well-known researchers including Popov, Yakubovich and Brockett [31, 155, 207] studied the problem in the frequency domain, yielding the Circle, the Popov and the Yakubovich criterion. O'Shea, Zames and Falb then extended on these criteria, deriving the so-called Zames-Falb multipliers [212, 213], whose use decreases the conservatism over previous methods. See [40, 183] for more information on Zames-Falb multipliers. As early as in the 1990s, Suykens et al. conceptualized neural networks as an alternating sequence of linear and nonlinear operators that satisfy a sector condition [176, 177]. They recognized the potential of analyzing and synthesizing nonlinear dynamical systems with neural network components by viewing them as Lur'e systems.

In 1997, Megretski and Rantzer introduced integral quadratic constraints (IQCs), proposing a framework for system analysis with respect to nonlinearities, uncertainties, or time variations [136]. This framework enabled the analysis of nonlinear systems with rigorous robustness and performance guarantees, particularly in the presence of troublesome components, whose input-output characteristics can be modeled by an IQC. The majority of activation functions constituting the nonlinear components of neural networks are both sector-bounded and slope-restricted, properties that can be captured by IQCs. Based on the IQC framework [136], [68] discussed quadratic constraints to describe the sector-bounded, slope-restricted, and bounded activation functions of neural networks, popularizing the use of quadratic constraints and SDP methods for the analysis and design of neural networks. Thereafter, a growing body of research has emerged, which utilizes robust control techniques for the analysis and design of neural networks, see, e.g., [69, 92, 101, 142, 160, 208].

Besides its application to Lipschitz constant estimation [69, 92] and robustness analysis [8, PP6], the quadratic constraint description of nonlinear parts of a neural network has led to new results in reachability analysis [101, 144], neural network verification [142, 143, 145], and closed-loop stability analysis [146, 208]. The problem of checking how much the output of a neural network changes given some perturbation set on the input can be approached by propagating input uncertainties through the neural network to compute an outer bound on the reachability set of the output. Neural network verification in this context refers to ensuring that all possible inputs map to a safe set of outputs. This problem was described by an SDP using quadratic constraints in [68], chordal sparsity of the underlying constraints was exploited in [142], and the problem was solved more efficiently using polynomial optimization

in [143, 145]. SDP methods have also been applied to the reachability analysis of closed-loop systems with neural components [101, 144], for which [144] also apply sparse polynomial optimization. Additionally, sparse matrix decomposition is utilized for Lipschitz constant estimation in [206] and [189] suggest an SDP-based model-free verification for neural network controlled systems.

The stability of closed-loop systems with neural network controllers was tackled by [208] and later extended to imitation learning [209], bounding errors between an MPC controller and its approximation [PP2], and the synthesis of recurrent neural network controllers for partially observed systems [88, 108, 109]. Furthermore, SDP-based parameterizations of recurrent equilibrium networks with robustness and stability guarantees [162, 163] have been used in several applications where a parameterization of stable operators is needed [74, 133].

Efforts have also been made to derive new and tighter quadratic constraints for various activation functions, with a particular focus on common ReLU activation functions [68, 92]. In [60], multiple properties beyond slope restriction and sector boundedness satisfied by typical activation functions are listed, including boundedness, positivity of the activation function and its complement, leading to tighter over-approximations of the nonlinearity. In [164, 165], strengthened Circle and Popov criteria for ReLU neural networks are provided and [147] derive a complete set of quadratic constraints for the repeated ReLU activation function.

## 1.3 Contributions and outline of the thesis

In the following, we provide details on the outline of the thesis and state the contributions.

### Chapter 2: Background

In Chapter 2, we provide the necessary background for the main results presented in this thesis. In Section 2.1, we introduce neural networks, defining the key layer types and relevant architectures considered throughout the work, and streamline the training problem. Next, in Section 2.2, we review the IQC framework, with a particular focus on quadratic constraints related to slope restriction and sector boundedness. In Section 2.3, we discuss robustness in neural networks, summarize the LipSDP method [69] for Lipschitz constant estimation and introduce common adversarial attacks.

### Chapter 3: Robustness analysis of neural networks

The main contribution of Chapter 3 is an accurate and scalable SDP-based method for Lipschitz constant estimation for a general class of feedforward neural networks. We first present two intermediate results and subsequently extend them to the main result.

Existing works on SDP-based Lipschitz constant estimation, such as LipSDP [69], and neural network analysis in general have primarily focused on diagonally re-