Semialgebraic Representation of Monotone Deep Equilibrium Models and Applications to Certification

Anonymous Author(s)
Affiliation
Address
email

Abstract

Deep equilibrium models are based on implicitly defined functional relations and have shown competitive performance compared with the traditional deep networks. Monotone operator equilibrium networks (monDEQ) retain interesting performance with additional theoretical guaranties. Existing certification tools for classical deep networks cannot directly be applied to monDEQs for which much fewer tools exist. We introduce a semialgebraic representation for ReLU based monDEQs which allows to approximate the corresponding input output relation by semidefinite programming (SDP). We present several applications to network certification and obtain SDP models for the following problems : robustness certification, Lipschitz constant estimation, ellipsoidal uncertainty propagation. We use these models to certify robustness of monDEQs w.r.t. a general L_q norm. Experimental results show that the proposed models outperform existing approaches for monDEQ certification. Furthermore, our investigations suggest that monDEQs are much more robust to L_2 perturbations than L_∞ perturbations.

1 Introduction

With the increasing success of Deep Neural Networks (DNN) (e.g. computer vision, natural language processing), one witnesses a significant increase in size and complexity (topology and activation functions). This generates difficulties for theoretical analysis and a posteriori performance evaluation. This is problematic for applications where robustness issues are crucial, for example inverse problems (IP) in scientific computing. Indeed such IPs are notoriously ill-posed and as stressed in the March 2021 issue of SIAM News [1], *"Yet DL has an Achilles' heel. Current implementations can be highly unstable, meaning that a certain small perturbation to the input of a trained neural network can cause substantial change in its output. This phenomenon is both a nuisance and a major concern for the safety and robustness of DL-based systems in critical applications—like healthcare—where reliable computations are essential"*. Indeed, the Instability Theorem [1] predicts unavoidable lower bound on Lipschitz contants, which may explain the lack of stability of some DNNs, over-performing on training sets. This underlines the need to evaluate precisely a posteriori critical indicators, such as Lipschitz constants of DNNs. However, obtaining an accurate upper bounds on the Lipschitz constant of a DNN is a hard problem, it reduces to *certifying globally* an inequality "$\Psi(\mathbf{x}) \geq 0$ for all $\mathbf{x}$ in a domain", i.e., to provide a *certificate of positivity* for a function Ψ, which has no simple explicit expression. Even for modest size DNNs this task is practically challenging, previous successful attempts [6, 21] were restricted in practice to no more than two hidden layers with less than a hundred nodes. More broadly existing attempts to DNN certification rely either on zonotope calculus, linear programming (LP) or hierarchies of SDP based on positivity certificates from algebraic geometry [25], which may suffer from the curse of dimensionality.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Recently, *Deep Equilibrium Models (DEQ)* [2] have emerged as a potential alternative to classical DNNs. With their much simpler layer structure, they provide competitive results on machine learning benchmarks [2, 3]. The training of DEQs involves solving fix-point equations for which algorithmic success requires conditions. Fortunately, *Monotone operator equilibrium network (monDEQ)* introduced in [35] satisfies such conditions. Moreover, the authors in [24] provide explicit bounds on global Lipschitz constant of monDEQs (w.r.t. the L_2-norm) which can be used for robustness certification.

From a certification point of view, DEQs have the definite advantage of being relatively small in size compared to DNNs and therefore potentially more amenable to sophisticated techniques (e.g. algebraic certificates of positivity) which rapidly face their limit even in modest size classical DNNs. Therefore monDEQs constitute a class of DNNs for which robustness certification, uncertainty propagation or Lipschicity could potentially be investigated in a more satisfactory way than classical networks. Contrary to DNNs, for which a variety of tools have been developped, certification of DEQ modeld is relatively open, the only available tool is the Lipschitz bound in [24].

Contribution

We present three general semialgebraic models of ReLU monDEQ for certification ($p \in \mathbb{Z}_+ \cup \{+\infty\}$):

• *Robustness Model* for network L_p robustness certification.

• *Lipschitz Model* for network Lipschitz constant estimation with respect to any L_p norm.

• *Ellipsoid Model* for ellipsoidal outer-approximation of the image by the network of a polyhedra or an ellipsoid.

All these models can be used for robustness certification, a common task which we consider experimentally. Both Lipschitz and Ellipsoid models can in addition be used for further a posteriori analyses. Interestingly, all three models are given by solutions of semidefinite programs (SDP), obtained by Shor relaxation of a common semialgebraic representation of ReLU monDEQs. Our models are all evaluated to simple ReLU monDEQs on MNIST dataset similar as [35, 24] on the task of robustness certification. We demonstrate that all three models ourperform the approach of [24] and the Robustness Model being the most efficient. Our experiments also suggest that DEQs are much less robust to L^∞ perturbations than L^2 perturbations, in contrast with classical DNNs [27].

Related works

Neural network certification is a challenging topic in machine learning, contributions include:

Robustness certification of DNNs Even with high test accuracy, DNNs are very sensitive to tiny input perturbations, see e.g. [32, 12]. Robustness to input perturbation has been investigated in many different works with various techniques, including SDP relaxation in SDP-cert [27], abstract interpretation with ERAN [11, 31], LP relaxation in Reluplex [18], analytical certification with Fast-lin [34] and CROWN [37], and their extension to convolutional neural networks with CNN-Cert [5]. All these methods are restricted to DNNs or CNNs and do not directly apply to DEQs.

Lipschitz constant estimation of DNNs Lipschitz constant of DNNs can be used for robustness certification [32, 14]. Existing contributions include naive layer-wise product bound [17, 33], the LP relaxations for CNNs [38] and DNNs [21], as well as SDP relaxations [26, 10, 6].

Lipschitz constant estimation of DEQs The authors in [24] provide an optimization-free upper bound of the Lipschitz constant of monDEQs depending on network weights. This bound is valid for L^2 norm and we present a general model for arbitrary L^p norm.

2 Preliminary Background and Notations

We consider the *monotone operator equilibrium network (monDEQ)* [35], with a single implicit hidden layer. The main difference between monDEQ and *deep equilibrium network (DEQ)* [2] is that *strong monotonicity* is enforced on weight matrix and activation functions to guarantee the convergence of fixed point iterations. The authors in [35] proposed various structures of implicit

layer, we only consider fully-connected layers, investigation of more advanced convolutional layers is in our list of future works.

Network description: Denote by $F : \mathbb{R}^{p_0} \to \mathbb{R}^K$ a fully-connected monDEQ for classification, where p_0 is the input dimension and K is the number of labels. Let $\mathbf{x}_0 \in \mathbb{R}^{p_0}$ be the input variable and $\mathbf{z} \in \mathbb{R}^p$ be the variable in the implicit layer. We consider the ReLU activation function, which is simply defined as $\mathrm{ReLU}(x) = \max\{0, x\}$, and the output of the monDEQs can be written as

$$F(\mathbf{x}_0) = \mathbf{C}\mathbf{z} + \mathbf{c}, \mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x}_0 + \mathbf{u}), \tag{monDEQ}$$

where $\mathbf{W} \in \mathbb{R}^{p\times p}, \mathbf{U} \in \mathbb{R}^{p\times p_0}, \mathbf{u} \in \mathbb{R}^p, \mathbf{C} \in \mathbb{R}^{K\times p}, \mathbf{c} \in \mathbb{R}^K$ are parameters of the network. The vector-valued function $F(\mathbf{x}_0)$ provides a score for each label $i \in \{1, \dots, K\}$ associated to the input $\mathbf{x}_0$, the prediction corresponds to the highest score, i.e., $y_{\mathbf{x}_0} = \arg\max_{i=1,\dots,K} F(\mathbf{x}_0)_i$. As in [35], the matrix $\mathbf{I}_p - \mathbf{W}$ is *strongly monotone*: there is a known $m > 0$ such that $\mathbf{I}_p - \mathbf{W} \succeq m\mathbf{I}_p$, this constraint can be enforced by specific parametrization of the matrix $\mathbf{W}$. With the monotonicity assumption, the solution to equation $\mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x}_0 + \mathbf{u})$ is unique and can be evaluated using convergent algorithms, see [35] for more details.

Robustness of monDEQs: Given an input $\mathbf{x}_0 \in \mathbb{R}^{p_0}$, a norm $\|\cdot\|$, and a network $F : \mathbb{R}^{p_0} \to \mathbb{R}^K$, let y_0 be the label of input $\mathbf{x}_0$, i.e., $y_0 = \arg\max_{i=1,\dots,K} F(\mathbf{x}_0)_i$. For $\varepsilon > 0$, denote by $\mathcal{E} = \mathbb{B}(\mathbf{x}_0, \varepsilon, \|\cdot\|)$ the ball centered at $\mathbf{x}_0$ with radius ε for norm $\|\cdot\|$. If for all inputs $\mathbf{x} \in \mathcal{E}$, the label of $\mathbf{x}$ equals y_0, i.e., $y = \arg\max_{i=1,\dots,K} F(\mathbf{x})_i = y_0$, then we say that the network F is *ε-robust* at input $\mathbf{x}_0$ for norm $\|\cdot\|$. An equivalent way to verify whether the network F is ε-robust is to check that for all labels $i \neq y_0$, $F(\mathbf{x})_i - F(\mathbf{x})_{y_0} < 0$.

Semialgebraicity of ReLU function: The key reason why neural networks with ReLU activation function can be tackled using polynomial optimization techniques is *semialgebraicity* of the ReLU function, i.e., it can be expressed with a system of polynomial (in)equalities. For $x, y \in \mathbb{R}$, we have $y = \mathrm{ReLU}(x) = \max\{0, x\}$ if and only if $y(y-x) = 0, y \geq x, y \geq 0$. For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we denote by $\mathrm{ReLU}(\mathbf{x})$ the coordinate-wise evaluation of ReLU function, and by $\mathbf{x}\mathbf{y}$ the coordinate-wise product of $\mathbf{x}$ and $\mathbf{y}$. A subset of $\mathbb{R}^n$ defined by a finite conjunction of polynomial (in)equalities is called a *basic closed semialgebraic set*. The graph of the ReLU function is a basic closed semialgebraic set.

Going back to equation (monDEQ), we have the following equivalence:

$$\mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x}_0 + \mathbf{u}) \Leftrightarrow \mathbf{z}(\mathbf{z} - \mathbf{W}\mathbf{z} - \mathbf{U}\mathbf{x}_0 - \mathbf{u}) = 0,\ \mathbf{z} \geq \mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x}_0 + \mathbf{u},\ \mathbf{z} \geq 0, \tag{1}$$

where the right hand side is a system of polynomial (in)equalities. For the rest of the paper, mention of the ReLU function will refer to the equivalent polynomial system in (1).

POP and Putinar's positivity certificate: In general, a *polynomial optimization problem (POP)* has the form

$$\rho = \max_{\mathbf{x}\in\mathbb{R}^n} \{f(\mathbf{x}) : g_i(\mathbf{x}) \geq 0, i = 1, \dots, p\}, \tag{POP$_0$}$$

where f and g_i are polynomials whose degree is denoted by $\deg$. The robustness certification model (3.1), Lipischitz constant model (3.2) and ellipsoid model (3.3) are all POPs.

In most cases, the POPs are non-linear and non-convex problems, which makes them NP-hard. A typical approach to reduce the complexity of these problems is replacing the positivity constraints by Putinar's positivity certificate [25]. The problem (POP$_0$) is equivalent to

$$\rho = \min_{\lambda\in\mathbb{R}} \{\lambda : \lambda - f(\mathbf{x}) \geq 0, g_i(\mathbf{x}) \geq 0, i = 1, \dots, p, \forall \mathbf{x} \in \mathbb{R}^n\}. \tag{POP}$$

In order to reduce the size of the feasible set of problem (POP), we replace the positivity constraint $\lambda - f(\mathbf{x}) \geq 0$ by a weighted *sum-of-square (SOS)* polynomial decomposition, involving the polynomials g_i. Let d be a non-negative integer. Denote by $\sigma_0^d(\mathbf{x}), \sigma_i^d(\mathbf{x})$ some SOS polynomials of degree at most $2d$, for each $i = 1, \dots, p$. Note that if $d = 0$, such polynomials are non-negative real numbers. Then the positivity of $\lambda - f(\mathbf{x})$ is implied by the following decomposition

$$\lambda - f(\mathbf{x}) = \sigma_0^d(\mathbf{x}) + \sum_{i=1}^{p} \sigma_i^{d-\omega_i}(\mathbf{x}) g_i(\mathbf{x}), \quad \omega_i = \lceil \deg g_i/2 \rceil, \quad \forall \mathbf{x} \in \mathbb{R}^n, \tag{2}$$

for any $d \geq \max_i \omega_i$. Equation (2) is called the *order-d Putinar's certificate.* By replacing the positivity constraint $f(\mathbf{x}) - \lambda \geq 0$ in problem (POP) by Putinar's certificate (2), we have for

$d \geq \max_i \omega_i$,

$$\rho_d = \min_{\lambda \in \mathbb{R}} \{\lambda : \lambda - f(\mathbf{x}) = \sigma_0^d(\mathbf{x}) + \sum_{i=1}^{p} \sigma_i^{d-\omega_i}(\mathbf{x}) g_i(\mathbf{x}) \,,\ \omega_i = \lceil \deg g_i/2 \rceil, \forall \mathbf{x} \in \mathbb{R}^n\} \,. \quad \text{(POP-}d\text{)}$$

It is obvious that $\rho_d \geq \rho$ for all $d \geq \max_i \omega_i$. Under certain conditions (slightly stronger than compactness of the set of constraints), it is shown that $\lim_{d\to\infty} \rho_d = \rho$ [20]. The main advantage of relaxing problem (POP) to (POP-d) is that problem (POP-d) can be efficiently solved by *semidefinite programming (SDP)*. Indeed a polynomial f of degree at most $2d$ is SOS if and only if there exists a *positive semidefinte (PSD)* matrix $\mathbf{M}$ (called a *Gram matrix*) such that $f(\mathbf{x}) = \mathbf{v}(\mathbf{x})^T \mathbf{M} \mathbf{v}(\mathbf{x})$, for all $\mathbf{x} \in \mathbb{R}^p$, where $\mathbf{v}(\mathbf{x})$ is the vector of monomials of degree at most d.

Problem (POP-d) is also called the *order-d Lasserre's relaxation*. When the input polynomials are quadratic, the order-1 Lasserre's relaxation is also known as *Shor's relaxation* [30]. All our models are obtained using variations of Shor's relaxation applied to different POPs, see Section 3.3 for more details.

3 Semialgebraic Models for Certifying Robustness of Neural Networks

In this section, we introduce several models for certification of monDEQs. All the models are based on semialgebraicity of ReLU and $\partial\mathrm{ReLU}$ (the *subgradient* of ReLU, see Section 3.2) to translate our targeted problems to POPs. Then we use Putinar's certificates, defined in Section 2 to relax the non-convex problems to convex SDPs which can be solved efficiently using modern solvers. Each model can be eventually used to certify robustness but they also have their own independent interest.

Notations: Throughout this section, we consider a monDEQ for classification, denoted by F, with fixed, given parameters, $\mathbf{W} \in \mathbb{R}^{p\times p}, \mathbf{U} \in \mathbb{R}^{p\times p_0}, \mathbf{u} \in \mathbb{R}^p, \mathbf{C} \in \mathbb{R}^{K\times p}, \mathbf{c} \in \mathbb{R}^K$, where p_0 is the number of input neurons, p is the number of hidden neurons, and K is the number of labels. For $q \in \mathbb{Z}_+ \cup \{+\infty\}$, $\|\cdot\|_q$ is the L_q norm defined by $\|\mathbf{x}\|_q := (\sum_{i=1}^{p_0} |x_i|^q)^{1/q}$ for all $\mathbf{x} \in \mathbb{R}^{p_0}$. Throughout this section $\epsilon > 0$ and $\mathbf{x}_0 \in \mathbb{R}^{p_0}$ are fixed, we denote by $\mathcal{E} := \mathbb{B}(\mathbf{x}_0, \varepsilon, \|\cdot\|_q) = \{\mathbf{x} \in \mathbb{R}^{p_0} : \|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon\}$ the ball centered at $\mathbf{x}_0$ with radius ε for L_q norm, a perturbation region. If $q < +\infty$, i.e., q is a positive integer, $\|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon$ is equivalent to the polynomial inequality $\|\mathbf{x} - \mathbf{x}_0\|_q^q \leq \varepsilon^q$; if $q = \infty$, $\|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon$ is equivalent to $|\mathbf{x} - \mathbf{x}_0|^2 \leq \epsilon^2$ (where $|\mathbf{x}|$ denotes the vector of absolute values of coordinates of $\mathbf{x}$) which is a system of p_0 polynomial inequalities. Hence the input set $\mathcal{E}$ is a semialgebraic set for all considered L_q norms. For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, its *operator norm* induced by the norm $\|\cdot\|$ is given by $|||\mathbf{A}||| := \inf\{\lambda : \|\mathbf{A}\mathbf{x}\| \leq \lambda\|\mathbf{x}\|, \forall \mathbf{x} \in \mathbb{R}^n\}$.

3.1 Robustness Model

Let y_0 be the label of $\mathbf{x}_0$ and let $\mathbf{z} \in \mathbb{R}^p$ be the variables in the monDEQ implicit layer. The proposed model directly estimates upper bounds on the gap between the score of label y_0 and the score of labels different from y_0. Precisely, for $i \in \{1, \ldots, K\}$ such that $i \neq y_0$, denote by $\xi_i = (\mathbf{C}_{i,:} - \mathbf{C}_{y_0,:})^T$. For $\mathbf{x} \in \mathcal{E}$, the gap between its score of label i and label y_0 is $F(\mathbf{x})_i - F(\mathbf{x})_{y_0} = \xi_i^T \mathbf{z}$. The Robustness Model for monDEQ reads:

$$\delta_i := \max_{\mathbf{x}\in\mathbb{R}^{p_0}, \mathbf{z}\in\mathbb{R}^p} \{\xi_i^T \mathbf{z} : \mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x} + \mathbf{u}),\ \mathbf{x} \in \mathcal{E}\} \,. \quad \text{(CertMON-}i\text{)}$$

Using the semialgebraicity of both ReLU in (1), and set $\mathcal{E}$, problem (CertMON-i) is a POP for all i. As discussed in Section 2, one is able to derive a sequence of SDPs (Lasserre's relaxation) to obtain a converging serie of upper bounds of the optimal solution of (CertMON-i). For Robustness Model, we consider only the order-1 Lasserre's relaxation (Shor's relaxation), and denote by $\tilde{\delta}_i$ the upper bound of δ_i by Shor's relaxation, i.e., $\delta_i \leq \tilde{\delta}_i$. Recall that if for all label i different from y_0, we have $F(\mathbf{x})_i < F(\mathbf{x})_{y_0}$, then the label of $\mathbf{x}$ is still y_0. This justifies the following claim:

Certification criterion: If $\tilde{\delta}_i < 0$ for all $i \neq y_0$, then the network F is ε-robust at $\mathbf{x}_0$.

Robustness Model for DNNs has already been investigated in [27], where the authors also use Shor's relaxation as we do. Different from DNNs, we only have one implicit layer in monDEQ. Therefore, the number of variables in problem (CertMON-i) only depends on the number of input neurons p_0 and hidden neurons p.

3.2 Lipschitz Model

We bound the Lipschitz constant of monDEQ with respect to input perturbation. Recall that the *Lipschitz constant* of the vector-valued function F (resp. $\mathbf{z}$) w.r.t. the L_q norm and input ball $\mathcal{S} \supset \mathcal{E}$, denoted by $L^q_{F,\mathcal{S}}$ (resp. $L^q_{\mathbf{z},\mathcal{S}}$), is the smallest value of L such that $\|F(\mathbf{x}) - F(\mathbf{y})\|_q \leq L\|\mathbf{x} - \mathbf{y}\|_q$ (resp. $\|\mathbf{z}(\mathbf{x}) - \mathbf{z}(\mathbf{y})\|_q \leq L\|\mathbf{x} - \mathbf{y}\|_q$) for all $\mathbf{x}, \mathbf{y} \in \mathcal{S}$. For $\mathbf{x}, \mathbf{x}_0 \in \mathcal{S}$, with $\|\mathbf{x} - \mathbf{x}_0\|_q \leq \epsilon$, we can estimate the perturbation of the output as follows:

$$\|F(\mathbf{x}) - F(\mathbf{x}_0)\|_q \leq L^q_{F,\mathcal{S}} \cdot \|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon L^q_{F,\mathcal{S}}\,, \tag{3}$$

$$\|F(\mathbf{x}) - F(\mathbf{x}_0)\|_q \leq |||\mathbf{C}|||_q \cdot L^q_{\mathbf{z},\mathcal{S}} \cdot \|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon |||\mathbf{C}|||_q \cdot L^q_{\mathbf{z},\mathcal{S}}\,. \tag{4}$$

The authors in [24] use inequality (4) with $q = 2$, as they provide an upper bound of $L^2_{\mathbf{z},\mathcal{S}}$. In contrast, our model provides upper bounds on Lipschitz constants of F or $\mathbf{z}$ for arbitrary L_q norms. We directly focus on estimating the value of $L^q_{F,\mathcal{S}}$ instead of $L^q_{\mathbf{z},\mathcal{S}}$.

Since the ReLU function is non-smooth, we define its *subgradient*, denoted by $\partial\mathrm{ReLU}$, as the set-valued map $\partial\mathrm{ReLU}(x) = 0$ for $x < 0$, $\partial\mathrm{ReLU}(x) = 1$ for $x > 0$, and $\partial\mathrm{ReLU}(x) = [0, 1]$ for $x = 0$. Similar to ReLU function, $\partial\mathrm{ReLU}$ is also semialgebraic.

Semialgebraicity of $\partial\mathrm{ReLU}$: If $x, y \in \mathbb{R}$, we have $y \in \partial\mathrm{ReLU}(x)$, if and only if $y(y-1) \leq 0, xy \geq 0, x(y-1) \geq 0$. If $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, then $\partial\mathrm{ReLU}(\mathbf{x})$ denotes the coordinate-wise evaluation of $\partial\mathrm{ReLU}$. Going back to monDEQ, let $\mathbf{s}$ be any subgradient of the implicit variable: $\mathbf{s} \in \partial\mathrm{ReLU}(\mathbf{W}\mathbf{z}+\mathbf{U}\mathbf{x}+\mathbf{u})$. We can write equivalently a system of polynomial inequalities:

$$\mathbf{s}(\mathbf{s} - 1) \leq 0,\ \mathbf{s}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x} + \mathbf{u}) \geq 0,\ (\mathbf{s} - 1)(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x} + \mathbf{u}) \geq 0\,. \tag{5}$$

For the following discussion, $\partial\mathrm{ReLU}$ will refer to the equivalent polynomial systems (5).

With the semialgebraicity of ReLU in (1) and $\partial\mathrm{ReLU}$ in (5), one is able to compute upper bounds of the Lipschitz constant of F via POPs. The proof of the next Lemma is postponed to Appendix A.1.

Lemma 1 *Define*

$$\begin{aligned}\tilde{L}^q_{F,\mathcal{S}} = \max\{&\mathbf{t}^T\mathbf{U}^T\mathbf{y} : \mathbf{t}, \mathbf{x} \in \mathbb{R}^{p_0},\ \mathbf{s}, \mathbf{z}, \mathbf{y}, \mathbf{r} \in \mathbb{R}^p,\ \mathbf{v}, \mathbf{w} \in \mathbb{R}^K,\ \mathbf{x} \in \mathcal{S},\\ &\|\mathbf{t}\|_q \leq 1,\ \mathbf{w}^T\mathbf{v} \leq 1,\ \|\mathbf{w}\|_q \leq 1,\ \mathbf{r} - \mathbf{W}^T\mathbf{y} = \mathbf{C}^T\mathbf{v},\ \mathbf{y} = \mathrm{diag}(\mathbf{s}) \cdot \mathbf{r};\\ &\mathbf{s} \in \partial\mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x} + \mathbf{u}),\ \mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{U}\mathbf{x} + \mathbf{u})\}\,. \qquad \text{(LipMON)}\end{aligned}$$

Then $\tilde{L}^q_{F,\mathcal{S}}$ is an upper bound of the Lipschitz constant of F w.r.t. the L_q norm, i.e., $L^q_{F,\mathcal{S}} \leq \tilde{L}^q_{F,\mathcal{S}}$.

Since problem (LipMON) is a POP, we also consider Shor's relaxation and denote by $\hat{L}^q_{F,\mathcal{S}}$ the upper bound of $\tilde{L}^q_{F,\mathcal{S}}$ by Shor's relaxation, i.e., $\tilde{L}^q_{F,\mathcal{S}} \leq \hat{L}^q_{F,\mathcal{S}}$. Define $\delta := \varepsilon\hat{L}^q_{F,\mathcal{S}}$. By equations (3), (4), if $\mathcal{E} \subset \mathcal{S}$, using Lemma 1 and the fact that $\|\cdot\|_\infty \leq \|\cdot\|_q$, we have $\|F(\mathbf{x}) - F(\mathbf{x}_0)\|_\infty \leq \delta$, yielding the following criterion:

Certification criterion: Let y_0 be the label of $\mathbf{x}_0$. Define $\tau := F(\mathbf{x}_0)_{y_0} - \max_{k \neq y_0} F(\mathbf{x}_0)_k$. If $2\delta < \tau$, then the network F is ε-robust at $\mathbf{x}_0$.

Remark: In order to avoid some possible numerical issues, we add some bound constraints and redundant constraints to problem (LipMON), see Appendix A.2 for details. Shor's relaxation of Lipschitz Model for DNNs has already been extensively investigated in [10, 6]. If one want to certify robustness for several input test examples, then one may choose S to be a big ball containing all such examples with an additional margin of ϵ.

3.3 Ellipsoid Model

As above, the input region $\mathcal{E}$ is a neighborhood of input $\mathbf{x}_0 \in \mathbb{R}^{p_0}$ with radius ε for the L_q norm, i.e., $\mathcal{E} = \{\mathbf{x} \in \mathbb{R}^p : \|\mathbf{x} - \mathbf{x}_0\|_q \leq \varepsilon\}$. More generally, $\mathcal{E}$ could be other general semialgebraic sets, such as polytopes or zonotopes. Denote by $F(\mathcal{E})$ the image of $\mathcal{E}$ by F. In this section, we aim at finding a semialgebraic set $\mathcal{C}$ such that $F(\mathcal{E}) \subseteq \mathcal{C}$. We choose $\mathcal{C}$ to be an ellipsoid which can in turn be used for robustness certification. Our goal is to find such outer-approximation ellipsoid with minimum volume. Let $\mathcal{C} := \{\xi \in \mathbb{R}^K : \|\mathbf{Q}\xi + \mathbf{b}\|_2 \leq 1\}$ be an ellipsoid in the output space $\mathbb{R}^K$ parametrized

by $\mathbf{Q} \in \mathbb{S}^K$ and $\mathbf{b} \in \mathbb{R}^K$, where $\mathbb{S}^K$ is the set of PSD matrices of size $K \times K$. The problem of finding the minimum-volume ellipsoid containing the image $F(\mathcal{E})$ can be formulated as

$$\max_{\mathbf{Q}\in\mathbb{S}^K, \mathbf{b}\in\mathbb{R}^K} \{\det(\mathbf{Q}) : \mathbf{x} \in \mathcal{E},\ \|\mathbf{Q}(\mathbf{C}\mathbf{z}+\mathbf{c})+\mathbf{b}\|_2 \leq 1,\ \mathbf{z} = \mathrm{ReLU}(\mathbf{W}\mathbf{z}+\mathbf{U}\mathbf{x}+\mathbf{u})\}\,. \tag{EllipMON-POP}$$

By semialgebraicity of both ReLU in (1) and $\mathcal{C}$, problem (EllipMON-POP) can be cast as a POP (the determinant being a polynomial). We no longer apply Shor's relaxation, we rather replace the non-negativity output constraint $1 - \|\mathbf{Q}(\mathbf{C}\mathbf{z}+\mathbf{c})+\mathbf{b}\|_2 \geq 0$ by a stronger Putinar's certificate related to both ReLU and input constraints. We can then relax the non-convex problem (EllipMON-POP) to a problem with SOS constraints, which can be reformulated by (convex) SDP constraints. This is due to the fact that a polynomial f of degree at most $2d$ is SOS if and only if there exists a PSD matrix $\mathbf{M}$ (called a *Gram matrix*) such that $f(\mathbf{x}) = \mathbf{v}(\mathbf{x})^T\mathbf{M}\mathbf{v}(\mathbf{x})$, for all $\mathbf{x} \in \mathbb{R}^p$, with $\mathbf{v}(\mathbf{x})$ being the vector containing all monomials of degree at most d. In summary, we relax problem (EllipMON-POP) to an SOS constrained problem keeping the determinant unchanged:

$$\begin{aligned}\max_{\mathbf{Q}\in\mathbb{S}^K, \mathbf{b}\in\mathbb{R}^K} \{\det(\mathbf{Q}) : 1 - \|\mathbf{Q}(\mathbf{C}\mathbf{z}+\mathbf{c})+\mathbf{b}\|_2^2 &= \sigma_0(\mathbf{x},\mathbf{z}) + \sigma_1(\mathbf{x},\mathbf{z})^T g_q(\mathbf{x}-\mathbf{x}_0) \\ &+ \tau(\mathbf{x},\mathbf{z})^T(\mathbf{z}(\mathbf{z}-\mathbf{W}\mathbf{z}-\mathbf{U}\mathbf{x}-\mathbf{u})) + \sigma_2(\mathbf{x},\mathbf{z})^T(\mathbf{z}-\mathbf{W}\mathbf{z}-\mathbf{U}\mathbf{x}-\mathbf{u}) + \sigma_3(\mathbf{x},\mathbf{z})^T\mathbf{z}\}\,.\end{aligned} \tag{EllipMON-SOS-d}$$

where $g_q(\mathbf{x}) = \varepsilon^q - \|\mathbf{x}\|_q^q$ for $q < +\infty$ and $g_q(\mathbf{x}) = \varepsilon^2 - |\mathbf{x}|^2$ for $q = +\infty$, σ_0 is a vector of SOS polynomials of degree at most $2d$, σ_1 is a vector of SOS polynomials of degree at most $2(d - \lceil q/2 \rceil)$ for $q < +\infty$ and $2d-2$ for $q = +\infty$, σ_2, σ_3 are vectors of SOS polynomials of degrees at most $2d-2$, τ is a vector of polynomials of degree at most $2d-2$. Problem (EllipMON-SOS-d) provides an ellipsoid feasible for (EllipMON-POP), that is an ellipsoid which contains $F(\mathcal{E})$. In practice, the determinant is replaced by a log-det objective because there exist efficient solver dedicated to optimize such objectives on SDP constraints. By increasing the relaxation order d in problem (EllipMON-SOS-d), one is able to obtain a hierarchy of log-det objected SDP problems for which the outer-approximation ellipsoids have decreasing volumes.

In this paper, we only consider the case $p = 2, \infty$, and order-1 relaxation ($d = 1$). Therefore, σ_i and τ are all (vectors of) real (non-negative) numbers for $i = 1, 2, 3$, except that σ_0 is an SOS polynomial of degree at most 2. In this case, problem (EllipMON-SOS-d) is equivalent to a problem with log-det objective and SDP constraints, as the following lemma states (proof postponed to Appendix A.3):

Lemma 2 *For $p = 2$ or $p = \infty$, problem* (EllipMON-SOS-d) *with $d = 1$ is equivalent to*

$$\max_{\mathbf{Q}\in\mathbb{S}^K, \mathbf{b}\in\mathbb{R}^K, \sigma_1,\sigma_2,\sigma_3\geq 0, \tau\in\mathbb{R}^p} \{\log\det(\mathbf{Q}) : -\mathbf{M} \succeq 0\}\,. \tag{EllipMON-SDP}$$

where $\mathbf{M} \in \mathbb{S}^{(p_0+p+1)\times(p_0+p+1)}$ is a symmetric matrix parametrized by the decision variables $(\mathbf{Q}, \mathbf{b})$, the coefficients $(\sigma_1, \sigma_2, \sigma_3, \tau)$, and the parameters of the network $(\mathbf{W}, \mathbf{U}, \mathbf{u}, \mathbf{C}, \mathbf{c})$.

Since the outer-approximation ellipsoid $\mathcal{C} = \{\xi \in \mathbb{R}^K : \|\mathbf{Q}\xi + \mathbf{b}\|_2 \leq 1\}$ contains the image $F(\mathcal{E})$, i.e., all possible outputs of the input region $\mathcal{E}$, one is able to certify robustness by solving the following optimization problems.

Certification criterion: Let y_0 be the label of $\mathbf{x}_0$. For $i \neq y_0$, define $\delta_i := \max_{\xi\in\mathbb{R}^K}\{\xi_i - \xi_{y_0} : \|\mathbf{Q}\xi + \mathbf{b}\|_2 \leq 1\}$. If $\delta_i < 0$ for all $i \neq y_0$, then the network F is ε-robust at $\mathbf{x}_0$.

The certification criterion for Ellipsoid Model has a geometric explanation: for $i \neq y_0$, denote by $\mathcal{P}_i$ the projection map from output space $\mathbb{R}^K$ to its 2-dimensional subspace $\mathbb{R}_{y0} \times \mathbb{R}_i$, i.e., $\mathcal{P}_i(\xi) = [\xi_{y_0}, \xi_i]^T$ for all $\xi \in \mathbb{R}^K$. Let $\mathcal{L}_i$ be the line in subspace $\mathbb{R}_{y0} \times \mathbb{R}_i$ defined by $\{[\xi_{y_0}, \xi_i]^T \in \mathbb{R}_{y0} \times \mathbb{R}_i : \xi_{y_0} = \xi_i\}$. Then the network F is ε-robust if and only if the projection $\mathcal{P}_i(\mathcal{C})$ lies strictly below the line $\mathcal{L}_i$ for all $i \neq y_0$. We give an explicit example in Section 4.3 to visually illustrate this.

3.4 Summary of the Models

We have already presented three models which can all be dedicated to certify robustness of neural networks. However, the size and complexity of each model are different. We summarize the number of variables in each model and the maximum size of PSD matrices in the resulting Shor's relaxation, see Table 1. The complexity of our models only depends on the number of neurons in the input layer

and implicit layer. The size of PSD matrices is a limiting factor for SDP solvers, our models are practically restricted to network for which such size can be handled by SDP solvers. For the popular dataset MNIST [36], whose input dimension is $28 \times 28 = 784$, we are able to apply our model on monDEQs with moderate size implicit layers (87) and report the corresponding computation time.

Table 1: Summary of the number of variables the three models

	Robustness Model	Lipschitz Model	Ellipsoid Model
Num. of variables	$p_0 + p$	$2p_0 + 4p + 2K$	$p_0 + 3p + K + K^2$
Max. size of PSD matrices	$1 + p_0 + p$	$1 + 2p_0 + 4p + 2K$	$1 + p_0 + p$

4 Experiments

In this section, we present the experimental results of Robustness Model, Lipschitz Model and Ellipsoid Model described in Section 3 for a pretrained monDEQ on MNIST dataset. The network we use consists of a fully-connected implicit layer with 87 neurons and we set its monotonicity parameter m to be 20. The training hyperparameters are set to be the same as in Table D1 of [35], where the training code (in Python) is available at `https://github.com/locuslab/monotone_op_net`. Training is based on the normalized MNIST database in [35], we use the same normalization setting on each test example with mean $\mu = 0.1307$ and standard deviation $\sigma = 0.3081$, which means that each input is an image of size 28×28 with entries varying from -0.42 to 2.82. And for every perturbation ε, we also take the normalization into account, i.e., we use the normalized perturbation ε/σ for each input.

Since all our three models can be applied to certify robustness of neural networks, we first compare the performance of each model in certification of the first 100 test MNIST examples. Then we compare the upper bounds of Lipschitz Model with the upper bounds proposed in [24]. Finally we show that Ellipsoid Model can also be applied for reachability analysis. For Certification model and Lipschitz model, we implement them in Julia [4] with JuMP [8] package; for Ellipsoid model, we implement it in Matlab [29] with CVX [13] package. For all the three models, we use Mosek [22] as a backend to solve the targeted POPs. All experiments are performed on a personal laptop with an Intel 8-Core i7-8665U CPU @ 1.90GHz Ubuntu 18.04.5 LTS, 32GB RAM. The code of all our models is available at `https://github.com/NeurIPS2021Paper4075/SemiMonDEQ`.

4.1 Robustness certification

We consider $\varepsilon = 0.1$ for the L_2 norm and $\varepsilon = 0.1, 0.05, 0.01$ for the L_∞ norm. For each model, we compute the ratio of certified test examples among the first 100 test inputs. Following [24], we also compute the *projected gradient descent (PGD)* attack accuracy using Foolbox library [28], which indicates the ratio of non-successful attacks among our 100 inputs. Note that the ratio of certified examples should always be less or equal than the ratio of non-successful attacks. The gaps between them shows how many test examples there are for which we are neither able to certify robustness nor find adversarial attacks.

Remark: For Lipschitz Model, we use inequality (3) to test robustness, i.e., we compute directly the upper bound of the Lipschitz constant of F rather than $\mathbf{z}$ (seen as a function of $\mathbf{x}$) where $\mathcal{S}$ is a big ball containing all test examples.

From Table 2, we see that the monDEQ is robust to all the 100 test examples for the L_2 norm and $\varepsilon = 0.1$ (the only example that we can not certify is because the label itself is wrong). However, it is not robust for the L_∞ norm at the same level of perturbation (all our three models cannot certify any examples, and the PGD algorithm finds adversarial examples for 85% of the inputs. The network becomes robust again for the L_∞ norm when we reduce the perturbation ε to 0.01. Overall, we see that Robustness Model is the best model as it provides the highest ratio, Ellipsoid Model is the second best model compared to Robustness Model, and Lipschitz Model provides the lowest ratio. As a trade-off, for each test example, Robustness Model requires to consider at most 9 optimization problems, each one being solved in around 150 seconds, while Ellipsoid Model requires to consider only one problem, which is solved in around 500 seconds. We only need to calculate one (global)

Lipschitz constant, which takes around 1500 seconds, so that we are able to certify any number of inputs. Each model we propose provide better or equal certification accuracy compared to [24], and significant improvements for L^∞ perturbations.

Table 2: Ratio of certified test examples and running time per example by different methods. We consider L_2 norm with $\varepsilon = 0.1$ and L_∞ norm with $\varepsilon = 0.1, 0.05, 0.01$. The ratio is based on the first 100 MNIST test examples, and we count the average computation time (with unit second) for one example of each method. The ratio in parentheses of the column "Lipschitz Model" are computed by the Lipschitz constant given in [24] (see Section 4.2 for details). Exact binomial 95% confidence intervals are given in bracket.

Norm	ε	Robustness Model (1350s / example)	Lipschitz Model (1500s in total)	Ellipsoid Model (500s / example)	PGD Attack
L_2	0.1	99% [>94]	91% (91%) [>83]	99%[>94]	99%[>94]
L_∞	0.1	0% [<4]	0% (0%) [<4]	0%[<4]	15% [8, 24]
	0.05	24% [16, 34]	0% (0%) [<4]	0% [<4]	82% [73, 89]
	0.01	99% [>94]	24% [16, 34] (0% [<4])	92% [>84]	99% [>94]

Figure 2 in Appendix A.4 shows the original image of the first test example (2a) and an adversarial attack (2b) for the L_∞ norm with $\varepsilon = 0.1$ found by the PGD algorithm in [28].

4.2 Comparison with Lipschitz constants

In this section, we compare the upper bounds of Lipschitz constants computed by Lipschitz Model with the upper bounds proposed in [24]. Notice that the upper bounds in [24] only involve the function $\mathbf{z}(\mathbf{x})$, hence we are only able to use inequality (4) to test robustness. In fact the quantity $|||\mathbf{C}|||_q \cdot L^q_{\mathbf{z},\mathcal{S}}$ can be regarded as an upper bound of $L^q_{F,\mathcal{S}}$, the Lipschitz constant of F. We denote by $\mathbf{UB}^2_{\mathbf{z}}$ the upper bound of the Lipschitz constant of $\mathbf{z}$ w.r.t. the L_2 norm, given by $\mathbf{UB}^2_{\mathbf{z}} = |||\mathbf{U}|||_2/m$ according to [24], where $\mathbf{U}$ is the parameter of the network and m is the monotonicity factor. We can then compute the upper bound w.r.t. the L_∞ norm by $\mathbf{UB}^\infty_z = \sqrt{p_0} \cdot \mathbf{UB}^2_z$ where p_0 is the input dimension. The upper bound of Lipschitz constant of F is computed via the upper bound of $\mathbf{z}$: $\mathbf{UB}^q_F = |||\mathbf{C}|||_q \cdot \mathbf{UB}^q_{\mathbf{z}}$. Denote similarly by $\mathbf{SemiUB}^q_F$ the upper bounds of Lipschitz constants of F provided by Lipschitz Model, w.r.t. the L_q norm.

Table 3: Comparison of upper bounds of Lipschitz constant for L_2 and L_∞ norm, and the corresponding computation time (with unit second).

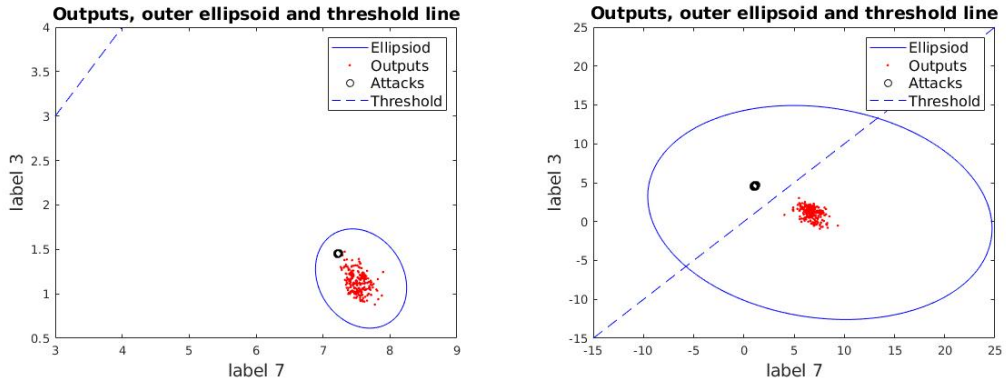
	$q = 2$		$q = \infty$	
	bound	time (s)	bound	time (s)
$\mathbf{UB}^q_F$	4.80	-	824.14	-
$\mathbf{SemiUB}^q_F$	4.67	1756.58	108.84	1898.65

From Table 3, we see that Lipschitz Model provides consistently tighter upper bounds than the ones in [24]. Especially for L_∞ norm, the upper bound computed by $|||\mathbf{C}|||_\infty \cdot \mathbf{UB}^\infty_{\mathbf{z}}$ is rather crude compared to the bound obtained directly by Lipschitz Model. Therefore, we are able to certify more examples using $\mathbf{SemiUB}^q_F$ than $\mathbf{UB}^q_F$, see Table 2.

4.3 Outer ellipsoid approximation

In this section, we provide a visible illustration of how Ellipsoid Model can be applied to certify robustness of neural networks. Recall that Ellipsoid Model computes a minimum-volume ellipsoid $\mathcal{C} = \{\xi \in \mathbb{R}^K : \|\mathbf{Q}\xi + \mathbf{b}\|_2 \leq 1\}$ in the output space $\mathbb{R}^K$ that contains the image of the input region $\mathcal{E}$ by F, i.e., $F(\mathcal{E}) \subseteq \mathcal{C}$. The certification criterion for Ellipsoid Model claims that the network F is ε-robust at input $\mathbf{x}_0$ if and only if the projection of $\mathcal{C}$ onto $\mathbb{R}_{y0} \times \mathbb{R}_i$ lies below the line $\mathcal{L}_i$ for all labels $i \neq y_0$.

Take the first MNIST test example (which is classified as 7) for illustration. For $\varepsilon = 0.1$, this example is certified to be robust for the L_2 norm but not for the L_∞ norm. We show the landscape of the projections onto $\mathbb{R}_7 \times \mathbb{R}_3$, i.e., the x-axis indicates label 7 and the y-axis indicates label 3. In Figure 1, the red points are projections of points in the image $F(\mathcal{E})$, for $\mathcal{E}$ an L_2 or L_∞ norm perturbation zone, the black circles are projections of some (successful and unsuccessful) adversarial examples found by the PGD algorithm. Notice that the adversarial examples also lie in the image $F(\mathcal{E})$. The blue curve is the boundary of the projection of the outer-approximation ellipsoid (which is an ellipse), and the blue dashed line plays the role of a certification threshold. Figure 1a shows the landscape for the L_2 norm, we see that the ellipse lies strictly below the threshold line, which means that for all points $\xi \in \mathcal{C}$, we have $\xi_3 < \xi_7$. Hence for all $\xi \in F(\mathcal{E})$, we also have $\xi_3 < \xi_7$. On the other hand, for the L_∞ norm, we see from Figure 1b that the threshold line crosses the ellipse, which means that we are not able to certify robustness of this example by Ellipsoid Model. Indeed, we can find adversarial examples with the PGD algorithm, as shown in Figure 1b by the black circles that lie above the threshold line. The visualization of one of the attack examples is shown in Figure 2 in Appendix A.4.



(a) Certified example for the L_2 norm

(b) Non-certified example for the L_∞ norm

Figure 1: Visualization of the outer-approximation ellipsoids and outputs with $\varepsilon = 0.1$ for L_2 norm (left) and L_∞ norm (right). The red points are image of the input region, the blue curve is the ellipsoid we compute, the blue dashed line is the threshold line used for certifying robustness of inputs, and the black circles are attack examples found by PGD algorithm.

5 Conclusion and Future Works

In this paper, we introduce semialgebraic representations of monDEQ and propose several POP models that are useful for certifying robustness, estimating Lipschitz constants and computing outer-approximation ellipsoids. For each model, there are several hierarchies of relaxations that allow us to improve the results by increasing the relaxation order. Even though we simply consider the order-1 relaxation, we obtain tighter upper bounds of Lipschitz constants compared to the results in [24]. Consequently, we are able to certify robustness of more examples.

Our models are based on SDP relaxation, hence requires an efficient SDP solver. However, the stat-of-the-art SDP solver Mosek (by interior-point method) can only handle PSD matrices of moderate size (smaller than 5000). This is the main limitation of our method if the dimension of the input gets larger. Moreover, we only consider the fully-connected monDEQ based on MNIST datasets for illustration. One important and interesting future work is to generalize our model to single and multi convolutional monDEQ, and to other datasets such as CIFAR [19] and SVHN [23]. A convolutional layer can be regarded as a fully-connected layer with a larger sparse weight matrix. Hence one is able to build similar models via sparse polynomial optimization tools.

The authors in [24] showed that we can train DEQs with small Lipschitz constants for the L_2 norm, by controlling the monotonicity of the weight matrix. This guarantees the robustness of monDEQ w.r.t. the L_2 norm but not for the L_∞ norm. A natural investigation track is to adapt this training technique to the L_∞ norm with a better control of the associated Lipschitz constant.

References

[1] Vegard Antun, Nina M. Gottschling, Anders C. Hansen, and Ben Adcock. Deep learning in scientific computing: Understanding the instability mystery. *SIAM NEWS MARCH 2021*, 2021.

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[3] Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Multiscale deep equilibrium models. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5238–5250. Curran Associates, Inc., 2020.

[4] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[5] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3240–3247, 2019.

[6] Tong Chen, Jean B Lasserre, Victor Magron, and Edouard Pauwels. Semialgebraic optimization for lipschitz constants of relu networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19189–19200. Curran Associates, Inc., 2020.

[7] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Wiley New York, 1983.

[8] Iain Dunning, Joey Huchette, and Miles Lubin. Jump: A modeling language for mathematical optimization. *SIAM review*, 59(2):295–320, 2017.

[9] Mahyar Fazlyab, Manfred Morari, and George J. Pappas. Probabilistic verification and reachability analysis of neural networks via semidefinite programming. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2726–2731, 2019.

[10] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[11] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2018.

[12] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[13] Michael Grant and Stephen Boyd. Cvx: Matlab software for disciplined convex programming, version 2.1, 2014.

[14] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[15] Didier Henrion, Jean-Bernard Lasserre, and Johan Löfberg. Gloptipoly 3: moments, optimization and semidefinite programming. *Optimization Methods & Software*, 24(4-5):761–779, 2009.

[16] Haimin Hu, Mahyar Fazlyab, Manfred Morari, and George J. Pappas. Reach-sdp: Reachability analysis of closed-loop systems with neural network controllers via semidefinite programming. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 5929–5934. IEEE, 2020.

[17] Todd Huster, Cho-Yu Jason Chiang, and Ritu Chadha. Limitations of the lipschitz constant as a defense against adversarial examples. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 16–29. Springer, 2018.

[18] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[20] Jean B Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on optimization*, 11(3):796–817, 2001.

[21] Fabian Latorre, Paul Rolland, and Volkan Cevher. Lipschitz constant estimation of neural networks via sparse polynomial optimization. In *International Conference on Learning Representations*, 2020.

[22] ApS Mosek. The mosek optimization toolbox for matlab manual, 2015.

[23] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[24] Chirag Pabbaraju, Ezra Winston, and J. Zico Kolter. Estimating lipschitz constants of monotone deep equilibrium models. In *International Conference on Learning Representations*, 2021.

[25] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993.

[26] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

[27] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[28] Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020.

[29] Gaurav Sharma and Jos Martin. Matlab®: a language for parallel computing. *International Journal of Parallel Programming*, 37(1):3–36, 2009.

[30] Naum Z. Shor. Quadratic optimization problems. *Soviet Journal of Computer and Systems Sciences*, 25:1–11, 1987.

[31] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[32] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[33] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[34] Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285. PMLR, 2018.

[35] Ezra Winston and J. Zico Kolter. Monotone operator equilibrium networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 10718–10728. Curran Associates, Inc., 2020.

[36] LeCun Yann, Cortes Corinna, and Burges Christopher J. C. Mnist handwritten digit database. 2010. [ATT Labs Online].

[37] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[38] Dongmian Zou, Radu Balan, and Maneesh Singh. On lipschitz bounds of general convolutional neural networks. *IEEE Transactions on Information Theory*, 66(3):1738–1759, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] All the claims made in the abstract are justified in methodological sections 2 and 3 or experimental section 4.
   (b) Did you describe the limitations of your work? [Yes] We restrict ourselves to fully-connected ReLU monDEQs as presented in Sections 2 and 5. We also discuss the limitation due to the size of PSD matrices which can be handled by SDP solvers in Section 3.4.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work is methodological and we do not foresee any direct negative societal impact of this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] The paper conforms to the guidelines.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] As presented in Section 2 we consider monDEQ models, which is our main assumption, no further assumption is required for all our models.
   (b) Did you include complete proofs of all theoretical results? [Yes] The methodological sections 2 and 3 describe all proof arguments for the soundness of our models. Proofs of some technical lemmas are postponed to the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code is available at `https://github.com/NeurIPS2021Paper4075/SemiMonDEQ` which is included in the main text.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We discuss training details in Section 4, we report the code used for training at `https://github.com/locuslab/monotone_op_net`.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report exact 95% binomial confidence intervals for proportions reported in Table 2.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We report in Section 4 the number of experiments, the average computation time and the type of personal computer used to perform all these experiments.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cited references for all the software used in our experiments.
   (b) Did you mention the license of the assets? [Yes] We report licenses for all assets used in the paper in Appendix A.5.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include code to reproduce our experiments as a separate URL.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We did not obtain any kind of new data.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We did not obtain any kind of new data.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not use crowdsourcing.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] We did not use crowdsourcing.

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] We did not use crowdsourcing.