2022 IEEE 61st Conference on Decision and Control (CDC)
December 6-9, 2022. Cancún, Mexico

Chordal Sparsity for Lipschitz Constant Estimation of Deep Neural Networks

Anton Xue, Lars Lindemann, Alexander Robey, Hamed Hassani, George J. Pappas, and Rajeev Alur †

2022 IEEE 61st Conference on Decision and Control (CDC) | 978-1-6654-6761-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CDC51059.2022.9993136

***Abstract*— Computing Lipschitz constants of neural networks allows for robustness guarantees in image classification, safety in controller design, and generalization beyond the training data. As calculating Lipschitz constants of neural networks is NP-hard, techniques for estimating Lipschitz constants must navigate the trade-off between scalability and accuracy. In this work, we significantly push the scalability frontier of a semidefinite programming technique known as LipSDP while achieving zero accuracy loss. We first show that LipSDP has *chordal sparsity*, which allows us to derive a chordally sparse formulation that we call Chordal-LipSDP. The key benefit is that the main computational bottleneck of LipSDP, a large linear matrix inequality, can be decomposed into an *equivalent* collection of smaller ones — allowing Chordal-LipSDP to outperform LipSDP particularly as the network depth grows. Moreover, our formulation uses a tunable sparsity parameter that enables one to gain tighter estimates without incurring a significant computational cost. We illustrate the scalability of our approach through extensive numerical experiments.**

I. Introduction

Neural networks are arguably the most common choice of function approximators used in machine learning and artificial intelligence. Their success is well documented in the literature and showcased in various applications, e.g., in solving the game Go [1] and in handwritten character recognition [2]. However, many neural networks are known to be non-robust, i.e., their outputs may be sensitive to small changes in the inputs and result in large deviations in the outputs [3]. It is hence often unclear exactly what a neural network learns and how it can generalize to previously unseen data. This is a particular concern in safety critical applications like perception in autonomous driving, where one would like to be robust and even obtain a robustness certificate. A way to measure the robustness of a neural network $f : \mathbb{R}^{n_1} \to \mathbb{R}^m$ is to calculate its Lipschitz constant L that satisfies

$$\|f(x) - f(y)\| \leq L\|x - y\| \quad \text{for all } x, y \in \mathbb{R}^{n_1}.$$

† The authors are with the School of Engineering and Applied Sciences, University of Pennsylvania, Philadelphia, PA, USA. {antonxue,larsl,arobey1,hassani,pappasg,alur}@seas.upenn.edu

AX is supported by the NSF Graduate Research Fellowship Program. LL, AR, and GJP are supported by NSF award CPS-2038873 and AFOSR grant FA9550-19-1-0265 (Assured Autonomy in Contested Environments). HH and AR are supported by NSF grants 1837253, 1943064, AFOSR grant FA9550-20-1-0111, DCIST-CRA, and the AI Institute for Learning-Enabled Optimization at Scale (TILOS). GJP and AR are supported by the NSF-Simons Foundation's Mathematical and Scientific Foundations of Deep Learning (MoDL) program on Transferable, Hierarchical, Expressive, Optimal, Robust, Interpretable NETworks (THEORINET). AX and RA are supported by the DARPA Assured Autonomy grant and the Office of Naval Research (ONR) award N00014-20-1-2115.

The exact calculation of L is NP-hard and hence poses computational challenges [4], [5]. Therefore, past effort has been on estimating upper bounds on the Lipschitz constant L in computationally efficient ways. A key difficulty here is to appropriately model the nonlinear activation functions within a neural network. For feedforward neural networks, the authors in [6] abstract activation functions using incremental quadratic constraints [7]. These are then formed into a convex semidefinite program (SDP), referred to as LipSDP, whose solution yields tight upper bounds on L. As the size of the neural network grows, however, the general formulation of LipSDP becomes computationally intractable. One may partially alleviate this issue by selectively reducing the number of optimization variables, which will induce sparsity into LipSDP at the cost of a looser bound. Still, this does not address the core computational bottleneck of LipSDP, which is that the solver must process a large linear matrix inequality (LMI) whose dimension scales with the number of neurons.

In this paper we study *computationally efficient formulations of LipSDP*. In particular, we introduce a variant of LipSDP that exhibits *chordal sparsity* [8], [9], which allows us to decompose a large LMI into an equivalent collection of smaller ones. Moreover, our formulation has a *tunable sparsity parameter*, enabling one to trade-off between efficiency and accuracy. We call our decomposed program *Chordal-LipSDP*, and study its theoretical properties and computational performance in this paper. The contributions of our work are as follows:

- We introduce a variant of LipSDP formulated in terms of a sparsity parameter τ and precisely characterize its chordal sparsity pattern. This allows us to decompose LipSDP, which is a large LMI, into a collection of smaller ones, yielding an equivalent problem that we call Chordal-LipSDP.
- We present numerical evaluations and observe that Chordal-LipSDP is significantly faster than LipSDP, especially for deeper networks, without accuracy loss relative to LipSDP. Furthermore, adjusting τ allows Chordal-LipSDP to obtain rapidly tightening bounds on L without incurring a high performance penalty.
- We make an open-source implementation available at `github.com/AntonXue/chordal-lipsdp`.

A. Related Work

There has been a great interest in the machine learning and control communities towards efficiently and accurately estimating Lipschitz constants of neural networks. Indeed,

it has been shown that there is a close connection between the Lipschitz constant of a neural network and its ability to generalize [10]. The authors in [11] were among the first to normalize weights of a neural network based on the Lipschitz constant. In control, Lipschitz constants of neural network-based control laws can be used to obtain stability or safety guarantees [12], [13]. Training neural networks with a desired Lipschitz constant is, however, difficult. In practice, one has to either solve constrained optimization problems, e.g., [14], or iteratively bootstrap training parameters. As a consequence, one is interested in obtaining Lipschitz certificates of neural networks. In [4], [5], it was shown that exact calculation of the Lipschitz constant is NP-hard. As estimating Lipschitz constants is computationally challenging, we are hereby motivated to efficiently estimate Lipschitz constants of neural networks.

Broadly, there are two common ways for estimating Lipschitz constants of deep neural networks, either sampling-based as in [15] and [16], or using optimization techniques [6], [17]. A naive approach is to calculate the product of the norm of the weights of each individual layer. The authors in [4] follow a similar idea, and obtain tighter Lipschitz constants using singular value decomposition and maximization over the unit cube. This, however, still becomes quickly computationally intractable for large neural networks. Tighter bounds have been obtained in [18] capturing cross-layer dependencies using compositions of nonexpansive averaged operators. However, again this approach does not scale well with the number of layers. While these works estimate global Lipschitz constants, it was shown in [19] that estimating local Lipschitz constants can be done more efficiently.

In this paper, we build on the LipSDP framework presented in [6], which amounts to solving a SDP. LipSDP abstracts activation functions into quadratic constraints and allows to encode rich layer-to-layer relations allowing to trade-off accuracy and efficiency. While LipSDP considers the l_2-norm, general l_p-norms on the input output relation of a neural network can be conservatively obtained using the equivalence of norms. The authors in [17] present LiPopt, which is a polynomial optimization framework that allows to calculate tight estimates of Lipschitz constants for l_2 and l_∞-norms. However, for l_2-norms LipSDP empirically shows to have tighter bounds. Exact computation of the Lipschitz constant under l_1 and l_∞ norms was presented in [5] by solving a mixed integer linear program. Lipschitz continuity of a neural network with respect to its training parameters has been analyzed in [20].

We show that a particular formulation of LipSDP satisfies *chordal sparsity* [8], [9], from which we apply chordal decomposition to obtain Chordal-LipSDP. The key benefit of exploiting chordal sparsity is that a large LMI is decomposed into an equivalent collection of smaller ones, in particular allowing us to scale to deeper networks. This equivalence also means that LipSDP and Chordal-LipSDP will compute *identical* estimates of the Lipschitz constant.

Most similar to our work are [21], [22]. The authors of [21] induce chordal sparsity in a special case of the DeepSDP [23] framework — which is similar to LipSDP — but only consider the case of ReLU activations and do not allow for efficiency-accuracy trade-offs. The authors of [22] uses sum-of-squares optimization [24] to study feedforward networks with ReLU activations, and observe, but do not formalize nor exploit, similar sparsity patterns as we do. A survey of scalability methods for SDPs is given in [25].

II. Background and Problem Formulation

In this section, we state the problem formulation and provide background on LipSDP and chordal sparsity.

A. Lipschitz Constant Estimation of Neural Networks

We consider feedforward neural networks $f : \mathbb{R}^{n_1} \to \mathbb{R}^m$ with $K \geq 2$ layers, i.e., $K-1$ hidden layers and one linear output layer. From now on, let $x_1 \in \mathbb{R}^{n_1}$ denote the input of the neural network. The output of the neural network is recursively computed for layers $k = 1, \dots, K-1$ as

$$f(x_1) \coloneqq W_K x_K + b_K, \quad x_{k+1} \coloneqq \phi(W_k x_k + b_k), \tag{1}$$

where W_k and b_k are the weight matrices and bias vectors of the kth layer, respectively, that are assumed to be of appropriate size. We denote the dimensions of $x_2, \dots, x_K$ by $n_2, \dots, n_K \in \mathbb{N}$. The function $\phi(u) \coloneqq \operatorname{vcat}(\varphi(u_1), \varphi(u_2) \dots)$ is the stack vector of activation functions φ, e.g., ReLU or tanh activation functions, that are applied element-wise. We assume throughout the paper that the same type of activation function is used across all layers.

B. LipSDP

We now present LipSDP [6] in a way that enables us later to conveniently characterize the chordal sparsity pattern of LipSDP. First, let $\mathbf{x} \coloneqq \operatorname{vcat}(x_1, \dots, x_K) \in \mathbb{R}^N$ be a stack of the state vectors with $N \coloneqq \sum_{k=1}^K n_k$. By defining

$$A \coloneqq \begin{bmatrix} W_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & W_{K-1} & 0 \end{bmatrix}, \quad B \coloneqq \begin{bmatrix} 0 & I_{n_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_{n_K} \end{bmatrix}$$

and $b \coloneqq \operatorname{vcat}(b_1, b_2, \dots, b_{K-1})$ we can rewrite the dynamics of (1) as $B\mathbf{x} = \phi(A\mathbf{x} + b)$, where $\phi : \mathbb{R}^{N_f} \to \mathbb{R}^{N_f}$ is a N_f-height stack of φ with $N_f \coloneqq n_2 + \cdots + n_K$. To deal with the nonlinear activation function ϕ in an efficient way, the key idea in LipSDP is to abstract ϕ using incremental quadratic constraints [7]. In particular, LipSDP considers a family of symmetric indefinite matrices $\mathcal{Q}$ such that any matrix $Q \in \mathcal{Q}$ satisfies

$$\begin{bmatrix} u - v \\ \phi(u) - \phi(v) \end{bmatrix}^\top Q \begin{bmatrix} u - v \\ \phi(u) - \phi(v) \end{bmatrix} \geq 0 \tag{2}$$

for all $u, v \in \mathbb{R}^{N_f}$. In the case where each element φ of ϕ is $[\underline{s}, \overline{s}]$-*sector-bounded*, i.e., where its subgradients satisfy $\partial\varphi \subseteq [\underline{s}, \overline{s}]$, then one possible parameterization of $\mathcal{Q}$ is

$$\mathcal{Q} \coloneqq \left\{ \begin{bmatrix} A \\ B \end{bmatrix}^\top \begin{bmatrix} -2\underline{s}\overline{s}T & (\underline{s} + \overline{s})T \\ (\underline{s} + \overline{s})T & -2T \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} : \gamma_\alpha \geq 0 \right\}$$

where T is a dense matrix that is parametrized by γ_α. In this paper, we fix an integer $\tau \geq 0$ and define T as follows [1]

$$T \coloneqq \sum_{i=1}^{N_f} (\gamma_\alpha)_{ii} e_i e_i^\top + \sum_{(i,j)\in\mathcal{I}_\tau} (\gamma_\alpha)_{ij} (e_i - e_j)(e_i - e_j)^\top,$$
$$I_\tau \coloneqq \{(i,j) = 1 \leq i < j \leq N_f,\ \ j - i \leq \tau\}.$$

By tuning the value of τ, we obtain different formulations of $\mathcal{Q}$ that all provide over-approximations of ϕ as in (2) while allowing us to trade-off on the spectrum of sparsity and accuracy. In the sparsest case, i.e., $\tau = 0$, the matrix T is a nonnegative diagonal matrix and $\gamma_\alpha \in \mathbb{R}_+^{N_f}$, while in the densest case, i.e., $\tau = N_f - 1$, the matrix T is fully dense and parameterized by $\gamma_\alpha \in \mathbb{R}_+^{1+\cdots+N_f}$.

To formulate our variant of LipSDP, we define the linearly-parametrized matrix-valued functions

$$Z_\alpha(\gamma_\alpha) \coloneqq \begin{bmatrix} A \\ B \end{bmatrix}^\top \begin{bmatrix} -2\underline{s}\overline{s}T & (\underline{s}+\overline{s})T \\ (\underline{s}+\overline{s})T & -2T \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix},$$
$$Z_\ell(\gamma_\ell) \coloneqq E_K^\top (W_K^\top W_K) E_K - \gamma_\ell E_1^\top E_1,$$
$$E_k \coloneqq \begin{bmatrix} \cdots & 0 & I_{n_k} & 0 & \cdots \end{bmatrix} \in \mathbb{R}^{n_k \times N},$$

where T is defined as above, $\gamma_\ell \in \mathbb{R}_+$, and E_k is the kth block-index selector such that $x_k = E_k \mathbf{x}$. Now combine the above terms as

$$Z(\gamma) \coloneqq Z_\alpha(\gamma_\alpha) + Z_\ell(\gamma_\ell) \in \mathbb{S}^N, \tag{3}$$

then LipSDP is the following semidefinite program:

$$\underset{\gamma \geq 0}{\text{minimize}}\ \ \gamma_\ell \quad \text{subject to}\ \ Z(\gamma) \preceq 0 \tag{4}$$

If $\gamma_\ell^\star$ is the optimal value of (4), then the Lipschitz constant of f is upper-bounded by $(\gamma_\ell^\star)^{1/2}$, see [6]. That is,

$$\|f(x) - f(y)\| \leq (\gamma_\ell^\star)^{1/2} \|x - y\| \quad \text{for all } x, y \in \mathbb{R}^{n_1}.$$

C. Chordal Sparsity

Chordal sparsity establishes a connection between graph theory and sparse matrix decomposition [26], [8]. In the context of this paper, we aim to solve the potentially large-scale SDP (4) using chordal sparsity in $Z(\gamma)$. This is done by decomposing the LMI $Z(\gamma) \preceq 0$ into an equivalent collection of smaller $Z_k \preceq 0$ LMIs, which we show in Section III.

1) Chordal Graphs and Sparse Matrices: A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ consists of vertices $\mathcal{V} \coloneqq \{1, \ldots, n\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We assume that $\mathcal{E}$ is symmetric, i.e. $(i,j) \in \mathcal{E}$ implies $(j,i) \in \mathcal{E}$, and so $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is an undirected graph. We say that the vertices $\mathcal{C} \subseteq \mathcal{V}$ form a *clique* if $u, v \in \mathcal{C}$ implies $(u,v) \in \mathcal{E}$, and let $\mathcal{C}(i)$ be the ith vertex of $\mathcal{C}$ under the natural ordering. A *maximal clique* is a clique that is not strictly contained within another clique. A *cycle* of length l is a sequence of vertices $v_1, \ldots, v_l$ with $(v_l, v_1) \in \mathcal{E}$ and adjacent connections $(v_i, v_{i+1}) \in \mathcal{E}$. A *chord* is any edge that connects two nonadjacent vertices in a cycle, and we say that a graph is *chordal* if every cycle of length four has at least one chord [8].

An edge set $\mathcal{E}$ can dually describe the sparsity pattern of a matrix. Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, define the set of symmetric matrices of size n with sparsity pattern $\mathcal{E}$ as

$$\mathbb{S}^n(\mathcal{E}) \coloneqq \{X \in \mathbb{S}^n : X_{ij} = X_{ji} = 0 \ \text{ if } \ (i,j) \notin \mathcal{E}\}. \tag{5}$$

If in addition $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is chordal and $X \in \mathbb{S}^n(\mathcal{E})$, then we say that X has *chordal sparsity* or is *chordally sparse*. For X with sparsity $\mathcal{E}$, we say that X_{ij} is *dense* if $(i,j) \in \mathcal{E}$, and that it is *sparse* otherwise.

2) Chordal Decomposition of Sparse Matrices: For a chordally sparse $X \in \mathbb{S}^n(\mathcal{E})$, useful decompositions can be analyzed through the cliques of $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Given a clique $\mathcal{C}_k \subseteq \mathcal{V}$, define its block-index matrix as follows:

$$(E_{\mathcal{C}_k})_{ij} = 1 \text{ if } \mathcal{C}_k(i) = j \text{ else } 0, \quad E_{\mathcal{C}_k} \in \mathbb{R}^{|\mathcal{C}_k| \times n}.$$

By decomposing a chordally sparse matrix with respect to its maximal cliques, a key result in sparse matrix analysis allows us to deduce the semidefiniteness of a large matrix with respect to a collection of smaller matrices.

Lemma 1 (Theorem 2.10 [9])**.** *Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a chordal graph and let $\{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$ be the set of its maximal cliques. Then $X \in \mathbb{S}^n(\mathcal{E})$ and $X \succeq 0$ if and only if there exists $X_k \in \mathbb{S}^{|\mathcal{C}_k|}$ such that each $X_k \succeq 0$ and*

$$X = \sum_{k=1}^{p} E_{\mathcal{C}_k}^\top X_k E_{\mathcal{C}_k}. \tag{6}$$

We say that (6) is a *chordal decomposition* of X by $\mathcal{C}_1, \ldots, \mathcal{C}_p$, and such a decomposition allows us to solve a large LMI using an equivalent collection of smaller ones.

III. Chordal Decomposition of LipSDP

In this section, we present Chordal-LipSDP which is a chordally sparse formulation of LipSDP. We first identify the sparsity pattern for $Z(\gamma)$ in Theorem 1 and then present Chordal-LipSDP in Theorem 2 as a chordal decomposition of LipSDP. An equivalence result is then stated in Theorem 3. The proofs of our results can be found in the appendix.

Our goal is to construct the edge set $\mathcal{E}$ of a chordal graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} \coloneqq \{1, \ldots, N\}$ such that $Z(\gamma) \in \mathbb{S}^N(\mathcal{E})$. To gain intuition for $\mathcal{E}$, we plot the dense entries of $Z(\gamma)$ in Figure 1 where the (i,j) square is dark if $(Z(\gamma))_{ij}$ is dense, i.e., $(i,j) \in \mathcal{E}$.

Fig. 1. The sparsity of $Z(\gamma)$ for $\tau = 0, 2, 4$ with dimensions $(3,3,3,3,3)$. For each increment of τ, each block grows by one unit on the bottom and right, and corresponds to a maximal clique of $\mathcal{G}(\mathcal{V}, \mathcal{E})$. As τ increases the number of blocks (maximal cliques) will decrease as the lower-right blocks become overshadowed. At $\tau = 0$ we have what [6] refers to as "LipSDP-neuron"; at $\tau = N_f - 1$ we have the completely dense "LipSDP-network".

[1] We use matrix-like subscripts on γ_α even though it is a vector. I_τ is a τ-banded index set, which makes T a τ-banded matrix. In general T may be dense, but restricting its structure will induce chordal sparsity in LipSDP.

In order to compactly present our results, we first define a notation for summation as follows:

$$S(k) \coloneqq \sum_{l=1}^{k} n_l, \;\; n_{K+1} \coloneqq m, \;\; S(0) \coloneqq 0, \;\; S(K) \coloneqq N.$$

Our main results are then stated in the following theorems.

Theorem 1. *Let $Z(\gamma)$ be defined as in* (3)*. It holds that $Z(\gamma) \in \mathbb{S}^N(\mathcal{E})$, where $\mathcal{E} \coloneqq \bigcup_{k=1}^{K-1} \mathcal{E}_k$ such that*

$$\mathcal{E}_k \coloneqq \big\{(i,j) : \; S(k-1)+1 \leq i,j \leq S(k+1)+\tau\big\}.$$

Note that the set $\mathcal{E}$ defined in Theorem 1 is already illustrated in Fig. 1. Also, we implicitly assume that all (i,j) in the definition of $\mathcal{E}$ are within $1 \leq i,j \leq N$. From this construction of $\mathcal{E}$ it is then straightforward to prove chordality of $\mathcal{G}(\mathcal{V},\mathcal{E})$ and identify its maximal cliques.

Theorem 2. *Let $\mathcal{V} \coloneqq \{1,\ldots,N\}$ and define $\mathcal{E}$ as in Theorem 1. Then $\mathcal{G}(\mathcal{V},\mathcal{E})$ is chordal and the set of its maximal cliques is $\{\mathcal{C}_1,\ldots,\mathcal{C}_p\}$, where*

$$p \coloneqq \min\{k : S(k+1)+\tau \geq N\}$$

and each clique $\mathcal{C}_k$ for $k<p$ has size and elements

$$|\mathcal{C}_k| \coloneqq n_k + n_{k+1} + \tau, \quad \mathcal{C}_k(i) \coloneqq S(k-1)+i$$

for $1 \leq i \leq |\mathcal{C}_k|$. The final clique $\mathcal{C}_p$ has elements

$$\mathcal{C}_p(i) \coloneqq S(p-1)+i, \quad 1 \leq i \leq N - S(p-1).$$

Using Lemma 1, the maximal cliques $\{\mathcal{C}_1,\ldots,\mathcal{C}_p\}$ from Theorem 2 now give a chordal decomposition of $Z(\gamma)$, and lets us formulate the following semidefinite program that we call Chordal-LipSDP:

$$\begin{aligned} \underset{\gamma\geq 0, Z_1,\ldots,Z_p}{\text{minimize}} \quad & \gamma_\ell \\ \text{subject to} \quad & Z(\gamma) = \sum_{k=1}^{p} E_{\mathcal{C}_k}^\top Z_k E_{\mathcal{C}_k}, \\ & Z_k \preceq 0 \;\text{ for }\; k=1,\ldots,p, \end{aligned} \tag{7}$$

We remark that solving Chordal-LipSDP is typically much faster than solving LipSDP, especially for deep neural networks as we impose a set of smaller LMIs instead of one large LMI. That is, the computational benefit of (7) over (4) is that each $Z_k \preceq 0$ constraint is a significantly smaller LMI than $Z(\gamma) \preceq 0$, which is the case for deeper networks.

In the next theorem, we show that LipSDP and Chordal-LipSDP compute Lipschitz constants that are in fact *identical*, i.e., a chordal decomposition of LipSDP gives no loss of accuracy over the original formulation.

Theorem 3. *The SDPs* (4) *and* (7) *are equivalent: γ is a solution for* (4) *iff $\gamma, Z_1, \ldots, Z_p$ is a solution for* (7)*. Moreover, their optimal objective values are identical.*

Note that Theorem 3 assumes the same T is used for both LipSDP and Chordal-LipSDP, and that this T is τ-banded.

IV. Experiments

In this section we evaluate the effectiveness of Chordal-LipSDP. Our aim is to answer the following questions:

(Q1) How well does Chordal-LipSDP scale in comparison to the baseline methods?
(Q2) How does the computed Lipschitz constant vary as the sparsity parameter τ increases?

(**Dataset**) We use a randomly generated batch of neural networks with random weights from $\mathcal{N}(0,1/2)$, with depth $K=d$, widths $n_2 = \cdots = n_K = w$, and input-output $n_1 = m = 2$ for $w \in \{10,\ldots,50\}$, and $d \in \{5,10,\ldots,50\}$. As a naming convention, for instance, W30-D20 would be the random network with $w=30$ and $d=20$. In total there are 50 such random networks.

(**Baseline Methods**) We compare Chordal-LipSDP against the following baselines:

- LipSDP: as in (4), using the same values of τ
- Naive-Lip: by taking $L = \prod_{k=1}^{K} \|W_k\|_2$
- CP-Lip [18], which scales exponentially with depth. To the best of our knowledge this is the only[2] other method that can handle general activation functions while yielding a non-trivial bound.

(**System**) All experiments were run on an Intel i9-9940X with 28 cores and 125 GB of RAM. Our codebase was implemented in Julia 1.7.2, and we used MOSEK 9.3 as our convex solver with a tolerance of $\varepsilon = 10^{-6}$.

A. (Q1) Runtime of Chordal-LipSDP vs Baselines

We first evaluate the runtime of Chordal-LipSDP against the baselines of LipSDP, Naive-Lip, and CP-Lip. For each random network we ran both Chordal-LipSDP and LipSDP with sparsity parameter values of $\tau = 0,\ldots,6$ and record their respective runtimes in Figure 2. Because Naive-Lip and CP-Lip do not depend on the sparsity parameter τ, they therefore appear as constant times for all sparsities; we omit plotting the Naive-Lip times because they are < 0.1 seconds for all networks. Moreover, because the runtime of CP-Lip scales exponentially with the number of layers, we only ran CP-Lip for networks of depth ≤ 25.

Figure 2 gives a general comparison for scalability between LipSDP and Chordal-LipSDP. We further record the runtimes of these two methods when the width of the network is fixed and the depth is varied in Figure 3, as well as when the depth is fixed and the width is varied in Figure 4.

We see that as the network depth increases, Chordal-LipSDP significantly out-scales LipSDP, especially for networks of depth $\geq$ 20. Moreover, Chordal-LipSDP also achieves better scaling for higher values of τ compared to LipSDP. In general, Naive-Lip is consistently the fastest method, while CP-Lip is initially fast, but quickly falls off on deep networks due to exponential scaling with depth.

[2]The method of [4] is not a true upper-bound of the Lipschitz constant, although it is often such in practice as demonstrated in [6]. The method of [5] assumes piecewise linear activations.

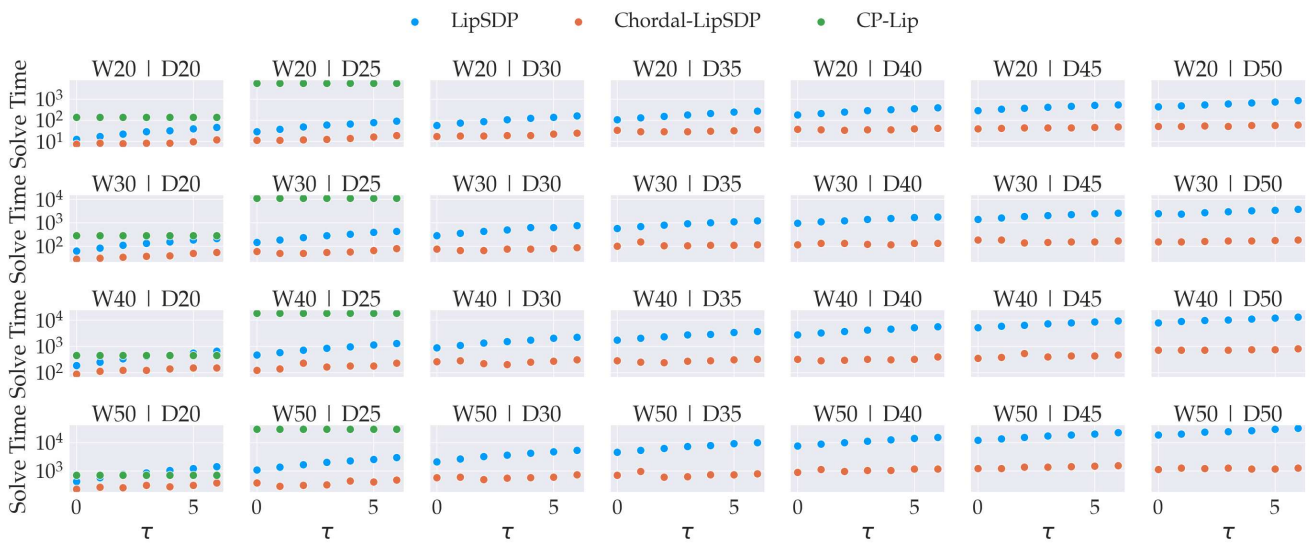

Fig. 2. The runtimes (seconds) of Chordal-LipSDP, LipSDP, and CP-Lip on a subset of the networks. The times for Naive-Lip are omitted because it finishes in < 0.1 seconds on all instances. We ran Chordal-LipSDP and LipSDP for $\tau = 0, \ldots, 6$. Because CP-Lip is independent of τ, it is a constant line. Moreover, due to the scaling exponentially with respect to the number of layers, we only ran CP-Lip for networks of depth ≤ 25.

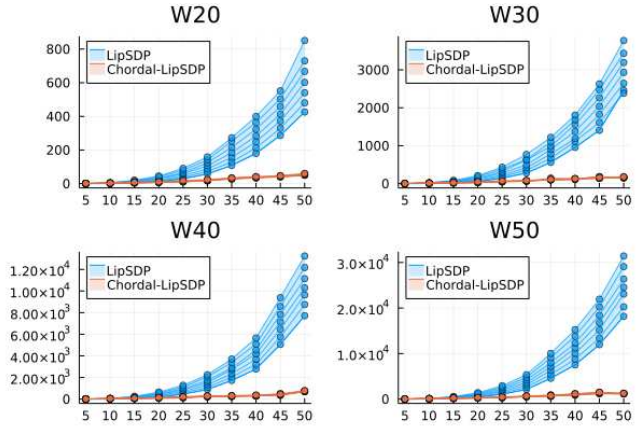

Fig. 3. The runtimes (seconds) of LipSDP and Chordal-LipSDP as the depth varies. The networks of each plot share the same width, but vary by depth on the x-axis. Each curve shows the runtimes for a different value of $\tau = 0, \ldots, 6$, where higher curves denote higher runtimes — and also higher values of τ. The region between $\tau = 0$ and $\tau = 6$ are shaded.

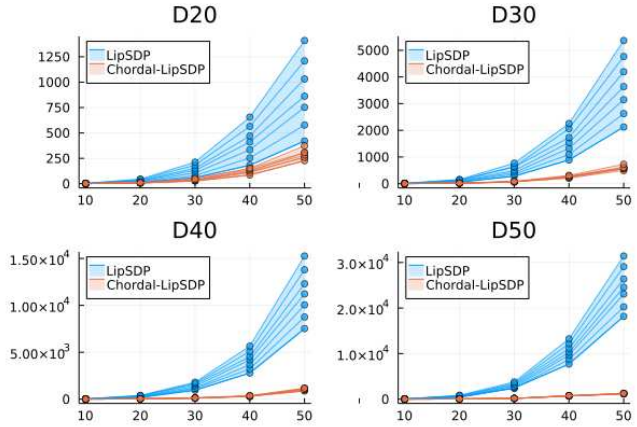

Fig. 4. The runtimes (seconds) of LipSDP and Chordal-LipSDP as the width varies. Similar to Figure 3, but the x-axis now shows varying widths.

B. (Q2) Lipschitz Constant vs Sparsity Parameter

We also studied how the value of τ affects the resulting Lipschitz constant and plot the results in Figure 5. In particular, as τ increases, the estimate rapidly improves by at least an order of magnitude. Moreover, the Lipschitz constant estimate is also better than Naive-Lip and CP-Lip — when the runtime would be reasonable (depth ≤ 25).

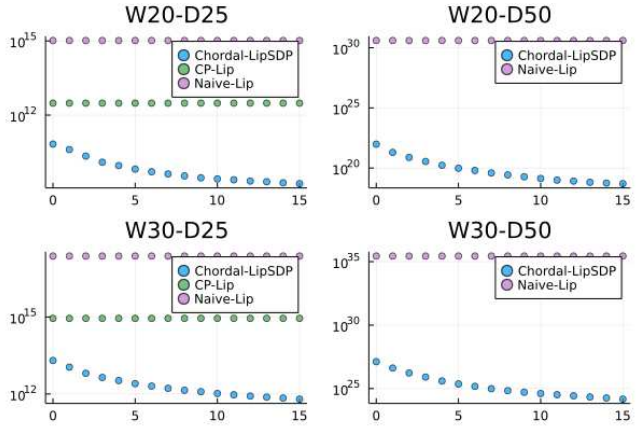

Fig. 5. The Lipschitz constant estimate given by Chordal-LipSDP (the same as LipSDP) on some networks, with τ on the x-axis. On the left we also plot the estimates given by CP-Lip (green) and Naive-Lip (purple).

C. Discussion

Our experiments show that Chordal-LipSDP out-scales LipSDP on deeper networks, but this is not necessarily the case for shallower networks, e.g. when depth ≤ 10. This is likely because the overhead of creating many smaller constraints of the form $Z_k \preceq 0$, as well as a large equality constraint $Z(\gamma) = \sum_{k=1}^{p} E_{\mathcal{C}_k}^\top Z_k E_{\mathcal{C}_k}$ may only be worthwhile when there are sufficiently many maximal cliques, i.e., when the network is deep. This also means that Chordal-LipSDP is likely more resource intensive than LipSDP.

We found that using `Dualization.jl` to preprocess LipSDP resulted in a significantly (often $\times 10$) faster solve time. This preprocessing did not yield noticeable benefits for Chordal-LipSDP, however, and so are not shown.

Additionally, we found it helpful to scale the weights of W_k in order to make sure that the solver receives a sufficiently well-conditioned problem, especially for larger problem instances. To ensure scaling correctness, we require that the scaled network $\widehat{f}$ must satisfy $f(x) = c_1 \widehat{f}(c_0 x)$ for all $x \in \mathbb{R}^{n_1}$ for some known $c_0, c_1 \in \mathbb{R}_+$, which is possible for ReLU activations.

V. Conclusions

We present Chordal-LipSDP, a chordally sparse variant of LipSDP for estimating the Lipschitz constant of a feedforward neural network. We give a precise characterization of the sparsity structure present, and using this we decompose the large LMI of LipSDP — which is its main computational bottleneck — into an equivalent collection of smaller constraints.

Our numerical experiments show that Chordal-LipSDP significantly out-scales LipSDP, especially on deeper networks. Moreover, our formulation introduces a tunable sparsity parameter that allows the user to finely trade-off accuracy and scalability: in fact it is often possible to gain rapidly tightening estimates of the Lipschitz constant without incurring a major performance penalty.

Appendix

A. Proof of Theorem 1

To simplify and formalize the proof, we first need to introduce some useful notation. We extend the definition of sparsity patterns to general matrices. Let $\mathcal{E} \subseteq \{1, \dots, m\} \times \{1, \dots, n\}$ and define analogously to (5):

$$\mathbb{M}^{m\times n}(\mathcal{E}) \coloneqq \{M \in \mathbb{R}^{m\times n} : M_{ij} = 0 \text{ if } (i,j) \notin \mathcal{E}\}, \quad (8)$$

and for $(i,j) \in \mathcal{E}$ associated with an $m \times n$ matrix we will assume that $1 \leq i \leq m$ and $1 \leq j \leq n$. When $m = n$, we simply write $\mathbb{M}^n$. Let $\mathcal{E}^\top$ be the transpositioned (inverse) pairs of $\mathcal{E}$, and note that $\mathcal{E} = \mathcal{E}^\top$ iff $\mathcal{E}$ is symmetric. We will explicitly distinguish between symmetric and nonsymmetric $\mathcal{E}$ when necessary. Whenever we write $\mathbb{S}^n(\mathcal{E})$ it is implied that $\mathcal{E}$ is symmetric, and that the undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is therefore well-defined.

In the remainder, we also use the following notation

$$k_i \coloneqq \min\{k : S(k) \geq i\}, \quad 1 \leq i \leq N.$$

There are a few useful properties for k_i that we remark:

- k_i is the index of $(n_1, \dots, n_K)$ that $1 \leq i \leq N$ falls in.
- If $i \leq j$, then $k_i \leq k_j$.
- $S(k_i - 1) \leq i \leq S(k_i)$ for all $1 \leq i \leq N$.

Also, some rules of sparse matrix arithmetics are as follows:

$$\begin{aligned} A \in \mathbb{M}^{m\times n}(\mathcal{E}) &\implies A^\top \in \mathbb{M}^{n\times m}(\mathcal{E}^\top) \\ A \in \mathbb{M}^n(\mathcal{E}_A), B \in \mathbb{M}^n(\mathcal{E}_B) &\implies A + B \in \mathbb{M}^n(\mathcal{E}_A \cup \mathcal{E}_B) \\ A \in \mathbb{M}^n(\mathcal{E}) &\implies A + A^\top \in \mathbb{S}^n(\mathcal{E} \cup \mathcal{E}^\top) \end{aligned}$$

To prove Theorem 1, we need to show that $Z(\gamma) \in \mathbb{S}^N(\mathcal{E})$. Note first that $Z(\gamma)$ can be expressed as:

$$\begin{aligned} Z(\gamma) = A^\top T A + B^\top T B + A^\top T B + B^\top T A \\ + E_K^\top W_K^\top W_K E_K - \gamma_l E_1^\top E_1. \end{aligned}$$

The proof of Theorem 1 follows five steps and analyzes sparsity of each term in $Z(\gamma)$ separately. For better readability, we summarize these five steps next and provide detailed proofs for each step in separate lemmas.

Step 1. We construct the edge set $\mathcal{E}_B \coloneqq \bigcup_{k=1}^{K-1} \mathcal{E}_{B,k}$ where

$$\begin{aligned} \mathcal{E}_{B,k} \coloneqq \big\{(i,j) :\ & S(k-1)+1 \leq j \leq S(k), \\ & S(k) - \tau + 1 \leq i \leq S(k+1) + \tau \big\}. \end{aligned}$$

In Lemma 2, we show that $B^\top T A \in \mathbb{M}^N(\mathcal{E}_B)$. By symmetry, it then also holds that $A^\top T B \in \mathbb{M}^N(\mathcal{E}_B^\top)$.

Step 2. By construction, each $\mathcal{E}_{B,k}$ has dense entries only in the column range $S(k-1)+1 \leq j \leq S(k)$, which means that $\mathcal{E}_{B,k} \cap \mathcal{E}_{B,k'} = \emptyset$ when $k \neq k'$. The goal now is to show that $B^\top T A + A^\top T B$ is in a sense the "frontier" of growth for $Z(\gamma)$ as τ increases, as seen in Figure 1. To more easily analyze the growth pattern of $\mathcal{E}_B \cup \mathcal{E}_B^\top$, we define an over-approximation $\mathcal{E}_C \coloneqq \bigcup_{k=1}^{K} \mathcal{E}_{C,k} \supseteq \mathcal{E}_B$ where

$$\begin{aligned} \mathcal{E}_{C,k} \coloneqq \big\{(i,j) :\ & S(k-1)+1 \leq j \leq S(k), \\ & 1 \leq i \leq S(k+1) + \tau \big\}. \end{aligned}$$

Each $\mathcal{E}_{C,k}$ is similar to $\mathcal{E}_{B,k}$, but with the i index range relaxed. In addition the union is up to K, which is the depth of the neural network. $\mathcal{E}_C$ is then a stair-case like sparsity pattern where the top side is dense (resp. the left side of $\mathcal{E}_C^\top$ is dense), and so $\mathcal{E}_C \cap \mathcal{E}_C^\top$ is an overlapping block diagonal structure. Our goal is now to show that each term of $Z(\gamma)$ has sparsity $\mathcal{E}_C \cap \mathcal{E}_C^\top$, beginning with $B^\top T A + A^\top T B$. In Lemma 3, we show that $\mathcal{E}_B \cup \mathcal{E}_B^\top \subseteq \mathcal{E}_C \cap \mathcal{E}_C^\top$. By Step 1, it consequently follows that

$$B^\top T A + A^\top T B \in \mathbb{S}^N(\mathcal{E}_B \cup \mathcal{E}_B^\top) \subseteq \mathbb{S}^N(\mathcal{E}_C \cap \mathcal{E}_C^\top).$$

Step 3. Let us next define the edge set $\mathcal{E}_A$ as

$$\begin{aligned} \mathcal{E}_A \coloneqq \big\{(i,j) :\ & S(k_j) - \tau + 1 \leq S(k_i + 1), \\ & S(k_i) - \tau + 1 \leq S(k_j + 1) \big\}. \end{aligned}$$

In Lemma 4, we show that

$$A^\top T A + E_K^\top W_K^\top W_K E_K - \gamma_\ell E_1^\top E_1 \in \mathbb{S}^N(\mathcal{E}_A),$$

while we show that $\mathcal{E}_A \subseteq \mathcal{E}_C \cap \mathcal{E}_C^\top$ in Lemma 5.

Step 4. For the remaining term $B^\top TB$ of $Z(\gamma)$, we show that $B^\top TB \in \mathbb{S}^N(\mathcal{E}_C \cap \mathcal{E}_C^\top)$ in Lemma 6.

Step 5. The previous steps imply that $Z(\gamma) \in \mathbb{S}^N(\mathcal{E}_C \cap \mathcal{E}_C^\top)$. Particularly, by Lemmas 3, 5, and 6, each term has sparsity $\mathcal{E}_C \cap \mathcal{E}_C^\top$, and therefore so does their sum. Finally, we show that $\mathcal{E}_C \cap \mathcal{E}_C^\top \subseteq \mathcal{E}$ in Lemma 7. This therefore means that $Z(\gamma) \in \mathbb{S}^N(\mathcal{E})$ and concludes the proof. □

B. Statement and proof of Lemma 2

Lemma 2. *It holds that* $B^\top TA \in \mathbb{M}^N(\mathcal{E}_B)$.

Proof. We analyze the action of $B^\top T$ on each block column $A_k \in \mathbb{R}^{N_f \times n_k}$ of A separately, where we have the partition $A = \begin{bmatrix} A_1 & \dots & A_{K-1} & 0 \end{bmatrix}$ and $A = \sum_{k=1}^{K-1} A_k E_k$. Since $W_k \in \mathbb{R}^{n_{k+1} \times n_k}$, the entry $(A_k)_{ij}$ is dense iff

$$S(k) - n_1 + 1 \le i \le S(k+1) - n_1,$$

and there is no condition on j since each column of A_k has at least one dense entry. Note that T is τ-banded, and so has the same sparsity as $R + R^\top$, where $R \coloneqq I + U + \cdots + U^\tau$ and U is the upper-shift matrix. Thus $(TA_k)_{ij}$ is dense iff

$$S(k) - n_1 - \tau + 1 \le i \le S(k+1) - n_1 + \tau.$$

Finally, left-multiplication by $B^\top$ pads a zero block of height n_1 at the top, and so $(B^\top TA_k)_{ij}$ is dense iff

$$S(k) - \tau + 1 \le i \le S(k+1) + \tau.$$

Right-multiplication by E_k puts A_k into the kth block column of $(n_1, \dots, n_K)$, so $(B^\top TA_k E_k)_{ij}$ is dense iff

$$\begin{aligned} S(k-1) + 1 \le j \le S(k), \\ S(k) - \tau + 1 \le i \le S(k+1) + \tau \end{aligned}$$

and so $B^\top TA_k E_k$ has sparsity $\mathcal{E}_{B,k}$. Since $B^\top TA$ is the sum of $B^\top TA_k E_k$, we have that $B^\top TA \in \mathbb{M}^N(\mathcal{E}_B)$. □

By symmetry we also have that $A^\top TB \in \mathbb{M}^N(\mathcal{E}_B^\top)$; the dense blocks of $B^\top TA$ grow vertically with τ, and those of $A^\top TB$ grow horizontally.

C. Statement and proof of Lemma 3

Lemma 3. *It holds that* $\mathcal{E}_B \cup \mathcal{E}_B^\top \subseteq \mathcal{E}_C \cap \mathcal{E}_C^\top$.

Proof. We show that $\mathcal{E}_{B,k} \subseteq \mathcal{E}_C$ and $\mathcal{E}_{B,k} \subseteq \mathcal{E}_C^\top$ for any $1 \le k \le K-1$. It suffices to consider only $\mathcal{E}_{B,k}$ because $\mathcal{E}_C \cap \mathcal{E}_C^\top$ is symmetric, and would therefore also contain $\mathcal{E}_{B,k}^\top$.

To show that $\mathcal{E}_{B,k} \subseteq \mathcal{E}_C$, observe that $\mathcal{E}_{B,k} \subseteq \mathcal{E}_{C,k}$. To show that $\mathcal{E}_{B,k} \subseteq \mathcal{E}_C^\top$, consider $(i,j) \in \mathcal{E}_{B,k}$, and we claim that $(i,j) \in \mathcal{E}_{C,k_i}^\top$, for which we need to satisfy

$$S(k_i - 1) + 1 \le i \le S(k_i), \quad 1 \le j \le S(k_i + 1) + \tau.$$

The LHS inequalities follow from the previously stated properties of k_i. For the RHS inequalities deduce the following from the definition of $\mathcal{E}_{B,k}$:

$$j \le S(k), \;\; S(k) - \tau + 1 \le i \implies j \le S(k) \le i + \tau - 1,$$

and because $i \le S(k_i + 1)$ we have

$$j \le i + \tau - 1 \le S(k_i + 1) + \tau,$$

meaning that $(i,j) \in \mathcal{E}_{C,k_i}^\top \subseteq \mathcal{E}_C^\top$. □

D. Statement and proof of Lemma 4

Lemma 4. *It holds that*

$$A^\top TA + E_K^\top W_K^\top W_K E_K - \gamma_\ell E_1^\top E_1 \in \mathbb{S}^N(\mathcal{E}_A).$$

Proof. Note that $(\gamma_\ell E_1^\top E_1)_{ij}$ being dense implies that $(A^\top TA)_{ij}$ is dense, and therefore it suffices to show that

$$\begin{aligned} A^\top TA + E_K^\top W_K^\top W_K E_K = W^\top \widehat{T} W \in \mathbb{S}^N(\mathcal{E}_A), \\ \widehat{T} \coloneqq \operatorname{blockdiag}(T, I), \quad W \coloneqq \operatorname{blockdiag}(W_1, \dots, W_K), \end{aligned}$$

where it is assumed that each $W_k \in \mathbb{R}^{n_{k+1} \times n_k}$ is dense. Because $\widehat{T}$ has more sparse entries than a τ-banded matrix of the same size, it is therefore less dense than $R^\top R$, where $R \coloneqq I + U + \cdots + U^\tau$ and U is the upper-shift matrix. Let $V \coloneqq RW$, it then suffices to show that $V^\top V \in \mathbb{S}^N(\mathcal{E}_A)$ because $V^\top V$ includes all the dense entries of $W^\top \widehat{T} W$. Observe that $(V^\top V)_{ij} = \sum_l V_{li} V_{lj}$ is dense iff the ith and jth columns of V share a row ℓ at which V_{li} and V_{lj} are both dense. Let $V_k \in \mathbb{R}^{(n_2 + \cdots + n_K + m) \times n_k}$ be the kth block column of V, then $(V_k)_{ij}$ is dense iff

$$S(k) - n_1 - \tau + 1 \le i \le S(k+1) - n_1.$$

Thus V_k and $V_{k'}$ have rows at which they are both dense iff

$$\begin{aligned} S(k) - n_1 - \tau + 1 \le S(k'+1) - n_1 \\ S(k') - n_1 - \tau + 1 \le S(k+1) - n_1, \end{aligned}$$

which are equivalent to the conditions described in $\mathcal{E}_A$. □

E. Statement and proof of Lemma 5

Lemma 5. *It holds that* $\mathcal{E}_A \subseteq \mathcal{E}_C \cap \mathcal{E}_C^\top$.

Proof. By symmetry of $\mathcal{E}_A$, it suffices to prove that $\mathcal{E}_A \subseteq \mathcal{E}_C$. Consider $(i,j) \in \mathcal{E}_A$, we claim that $(i,j) \in \mathcal{E}_{C,k_j}$ — for which a sufficient condition is

$$S(k_j - 1) + 1 \le j \le S(k_j), \quad 1 \le i \le S(k_j + 1) + \tau,$$

The LHS inequalities follow from the properties of k_j. For the RHS inequalities, recall that $i \le S(k_i)$, and rewrite with the second condition of $\mathcal{E}_A$ to yield

$$1 \le i \le S(k_i) \le S(k_j + 1) + \tau - 1,$$

and so $(i,j) \in \mathcal{E}_{C,k_j} \subseteq \mathcal{E}_C$. □

F. Statement and proof of Lemma 6

Lemma 6. *It holds that* $B^\top TB \in \mathbb{S}^N(\mathcal{E}_C \cap \mathcal{E}_C^\top)$.

Proof. By symmetry, it suffices to show $B^\top TB \in \mathbb{M}^N(\mathcal{E}_C)$. Because left (resp. right) multiplication by $B^\top$ (resp. B) consists of padding n_1 zeros on the top (resp. left), we may treat $B^\top TB$ as a τ-banded matrix. First suppose that $j \le i$, then $(B^\top TB)_{ij}$ is dense iff $i \le j + \tau$. Since $j \le S(k_j + 1)$,

$$1 \le i \le j + \tau \le S(k_j + 1) + \tau,$$

which shows that $(i,j) \in \mathcal{E}_{C,k_j} \subseteq \mathcal{E}_C$.

Now suppose that $i \le j$, then $(B^\top TB)_{ij}$ is dense iff $j \le i + \tau$. Furthermore, $S(k_i) \le S(k_j) \le S(k_j + 1)$, so

$$1 \le j \le i + \tau \le S(k_i) + \tau \le S(k_j + 1) + \tau$$

which again shows that $(i,j) \in \mathcal{E}_{C,k_j} \subseteq \mathcal{E}_C$. □

G. Statement and proof of Lemma 7

Lemma 7. *It holds that* $\mathcal{E}_C \cap \mathcal{E}_C^\top \subseteq \mathcal{E}$.

Proof. Consider $(i,j) \in \mathcal{E}_C \cap \mathcal{E}_C^\top$ and suppose without loss of generality that $i \leq j$. Then $(i,j) \in \mathcal{E}_{C,k_j}$ and $(i,j) \in \mathcal{E}_{C,k_i}^\top$, meaning that the following conditions hold:

$$S(k_j - 1) + 1 \leq j \leq S(k_j), \quad 1 \leq i \leq S(k_j + 1) + \tau,$$
$$S(k_i - 1) + 1 \leq i \leq S(k_i), \quad 1 \leq j \leq S(k_i + 1) + \tau.$$

Tightening the bounds for k_i and by monotonicity of S,

$$S(k_i - 1) + 1 \leq i \leq S(k_i) \leq S(k_i + 1) + \tau,$$
$$S(k_i - 1) + 1 \leq S(k_j - 1) + 1 \leq j \leq S(k_i + 1) + \tau,$$

which together imply that $(i,j) \in \mathcal{E}_{k_i} \subseteq \mathcal{E}$. □

H. Proof of Theorem 2

The structure of $\mathcal{E}$ results in a lengthy proof of Theorem 1. However, it is easy to guess each $\mathcal{E}_k$ by simple experiments, and leads to a straightforward proof of Theorem 2.

Note that $\mathbb{S}^N(\mathcal{E}_k)$ are the block diagonal matrices whose (i,j) entry is dense iff $S(k-1)+1 \leq i,j \leq S(k+1)+\tau$. Because $\mathcal{E}$ is a union of the $\mathcal{E}_k$ sparsities, $\mathbb{S}^N(\mathcal{E})$ is therefore the set of matrices with overlapping block diagonals, which are known to be chordal [8, Section 8.2].

It remains to identify the maximal cliques of $\mathcal{G}(\mathcal{V},\mathcal{E})$. First consider $k < p$ with $k \geq 1$, and observe that $(N,N) \notin \mathcal{E}_k$. By construction each $\mathcal{E}_k$ is the edges of a clique, and when $k < p$ such $\mathcal{E}_k$ is also not contained by any other $\mathcal{E}_{k'}$ because

$$\begin{cases} S(k-1)+1 < S(k'-1)+1, & \text{if } k < k' \\ S(k'+1)+\tau < S(k+1)+\tau, & \text{if } k > k', \end{cases}$$

so there exists $(i,j) \in \mathcal{E}_k \setminus \mathcal{E}_{k'}$ with $i = j = S(k-1)+1$ when $k < k'$, and $i = j = S(k+1)+\tau$ when $k > k'$. Thus, $\mathcal{E}_k$ is in fact the edges of a maximal clique containing indices i that satisfy $S(k-1)+1 \leq i \leq S(k+1)+\tau$, which are exactly the conditions of $\mathcal{C}_k$ for $k < p$.

Now consider $k > p$ with $k \leq K-1$, and observe that $\mathcal{E}_k \subseteq \mathcal{E}_p$ because any $(i,j) \in \mathcal{E}_k$ will satisfy

$$S(p-1)+1 < S(k-1)+1 \leq i,j \leq N \leq S(p+1)+\tau,$$

and so $(i,j) \in \mathcal{E}_p$ as well. $\mathcal{E}_p$ is therefore the edges of a clique that contains all other cliques that contain N, and is thus maximal — corresponding to the description of $\mathcal{C}_p$. □

I. Proof of Theorem 3

Because $Z(\gamma) \in \mathbb{S}^N(\mathcal{E})$ and $\mathcal{G}(\mathcal{V},\mathcal{E})$ is chordal with maximal cliques $\{\mathcal{C}_1,\ldots,\mathcal{C}_p\}$, conclude from Lemma 1 that $Z(\gamma) \preceq 0$ iff each $Z_k \preceq 0$. □

REFERENCES

[1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

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

[4] A. Virmaux and K. Scaman, "Lipschitz regularity of Deep Neural Networks: Analysis and Efficient Estimation," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 31, Montreal, Canada, December 2018, p. 3839–3848.

[5] M. Jordan and A. G. Dimakis, "Exactly Computing the Local Lipschitz Constant of ReLU Networks," in *Proc. of the Adv. in Neural Info. Processing Systems*, vol. 33, December 2020, pp. 7344–7353.

[6] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas, "Efficient and Accurate Estimation of Lipschitz Constants for Deep Neural Networks," in *Proc. of the Adv. in Neural Info. Processing Sys.*, vol. 32, Vancouver, Canada, December 2019, pp. 11 427–11 438.

[7] B. Açıkmeşe and M. Corless, "Observers for systems with nonlinearities satisfying incremental quadratic constraints," *Automatica*, vol. 47, no. 7, pp. 1339–1348, 2011.

[8] L. Vandenberghe and M. S. Andersen, "Chordal graphs and semidefinite optimization," *Foundations and Trends in Optimization*, vol. 1, no. 4, pp. 241–433, 2015.

[9] Y. Zheng, "Chordal Sparsity in Control and Optimization of Large-scale Systems," Ph.D. dissertation, University of Oxford, 2019.

[10] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," in *Proceedings of the Conference on Neural Information Processing Systems*, vol. 30, Long Beach, California, USA, December 2017.

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

[12] M. Jin and J. Lavaei, "Stability-Certified Reinforcement Learning: A Control-Theoretic Perspective," *IEEE Access*, vol. 8, pp. 229 086–229 100, 2020.

[13] L. Lindemann, A. Robey, L. Jiang, S. Tu, and N. Matni, "Learning Robust Output Control Barrier Functions from Safe Expert Demonstrations," *arXiv preprint arXiv:2111.09971*, 2021.

[14] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree, "Regularisation of neural networks by enforcing lipschitz continuity," *Machine Learning*, vol. 110, no. 2, pp. 393–416, 2021.

[15] G. Wood and B. Zhang, "Estimation of the Lipschitz constant of a function," *J. of Global Optimization*, vol. 8, no. 1, pp. 91–103, 1996.

[16] A. Chakrabarty, D. K. Jha, G. T. Buzzard, Y. Wang, and K. G. Vamvoudakis, "Safe Approximate Dynamic Programming via Kernelized Lipschitz Estimation," *IEEE Transactions On Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 405–419, 2020.

[17] F. Latorre, P. T. Y. Rolland, and V. Cevher, "Lipschitz constant estimation for Neural Networks via sparse polynomial optimization," in *Proceedings of the International Conference on Learning Representations*, April 2020.

[18] P. L. Combettes and J.-C. Pesquet, "Lipschitz Certificates for Layered Network Structures Driven by Averaged Activation Operators," *SIAM J. on Mathematics of Data Science*, vol. 2, no. 2, pp. 529–557, 2020.

[19] T. Avant and K. A. Morgansen, "Analytical bounds on the local Lipschitz constants of ReLU networks," *arXiv:2104.14672*, 2021.

[20] C. Herrera, F. Krach, and J. Teichmann, "Estimating Full Lipschitz Constants of Deep Neural Networks," *arXiv preprint arXiv:2004.13135*, 2020.

[21] M. Newton and A. Papachristodoulou, "Exploiting Sparsity for Neural Network Verification," in *Proceedings of Learning for Dynamics and Control*, June 2021, pp. 715–727.

[22] T. Chen, J. B. Lasserre, V. Magron, and E. Pauwels, "Semialgebraic Optimization for Lipschitz Constants of ReLU Networks," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 33, December 2020, pp. 19 189–19 200.

[23] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Trans. on Automatic Control*, 2020.

[24] P. A. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology, 2000.

[25] A. Majumdar, G. Hall, and A. A. Ahmadi, "Recent scalability improvements for semidefinite programming with applications in machine learning, control, and robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 331–360, 2020.

[26] A. Griewank and P. L. Toint, "On the existence of convex decompositions of partially separable functions," *Mathematical Programming*, vol. 28, no. 1, pp. 25–49, 1984.