Tightening the SDP Relaxation Gap in Neural Network Verification

Ziye Ma

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-198
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-198.html

December 2, 2020

Acknowledgement

Special thanks goes to my advisor Somayeh Sojoudi, who have offered her unparalleled mentorship throughout my time at Berkeley. This thesis report wouldn't have existed without her. I also want to express my gratitudes towards my colleagues, Brendon Anderson and Jingqi Li for their discussions and collaborations, which served as a groundwork for this report. Lastly, I want to thank my second reader Professor Javad Lavaei for offering me insights into this work and my friends and parents who have always been my strongest support.

Tightening the SDP Relaxation Gap in Neural Network Verification

by Ziye Ma

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Somayeh Sojoudi
Research Advisor

2020-12-02

* * * * * * *

Professor Javad Lavaei
Second Reader

2020-12-02

Tightening the SDP Relaxation Gap in Neural Network Verification

Ziye Ma

In partial fulfillment of Master of Science in Electrical Engineering and Computer Science, University of California, Berkeley

Abstract

In this thesis report, we consider certifying the robustness of pre-trained ReLU neural networks against adversarial input perturbations. To be more specific, we only consider the semidefinite programming (SDP) approach of certification in which the original non-convex certification problem is relaxed as a SDP programming.

The SDP approach enjoys good certification percentage because the relaxation gap that it poses is naturally smaller than that of other methods, including the Linear Programming (LP) based approaches and the Mixed Integer Programming (MIP) based approaches. However, even SDP approaches suffer from large relaxation gaps when used on large networks with many hidden layers. Towards this end, this work proposes 2 theoretically sound and empirically proven techniques to further bridge this gap when the network is large or when the current SDP approach is unsatisfactory. Both techniques can achieve exact certification in the asymptotic regime with exponential complexity. Moreover we show that even with polynomial complexity both techniques can greatly reduce the relaxation gap when compared to the original method.

The first technique is partitioning the input uncertainty set and solving the convex relaxations on each part separately. We show that this approach guarantees tightened relaxation error regardless of the given neural network. We use the feasible set geometry to show that partitions should be uniform along the coordinate axes of the input space. We then determine which coordinate best reduces the worst-case SDP relaxation error for the case of a two-part partition. The proposed methods are illustrated experimentally on IRIS classification networks and are shown to significantly reduce relaxation error, even offering certificates that are otherwise void without partitioning.

The second technique we introduce to reduce the relaxation gap is to add linear constraints to the SDP program that best approximates non-convex cuts via the use of disjunctive programming. The proposed method amounts to a sequential SDP technique. We analyze the performance of this method both theoretically and empirically, and show that it bridges the gap as the number of cuts increases.

Contents

1 Introduction

When considering a neural network as a system, its robustness and sensitivity to input are some key aspects with which the reliability can be determined. As a result, much effort has been placed on developing methods to certify the robustness of neural networks to perturbations in their input data [1–8]. Posing this verification problem as a mathematical programming requires us to cast the neural network mapping as a hard constraint.

A common deterministic certification procedure is to verify that all possible unknown inputs are mapped to outputs that the network operator classifies as safe [1, 7]. From this perspective, certification amounts to certifying that the image of an input uncertainty set is a subset of the safe set. However, even when the input uncertainty set is convex, its output set may be nonconvex, which renders the certification an NP-complete nonconvex optimization problem [4, 9]. To address this challenge, recent works have proposed to solve this nonconvex optimization through convexification, namely linear program (LP) relaxations (e.g., [1]) and semidefinite program (SDP) relaxations (e.g., [2, 6]).

LP-based techniques relax each ReLU function individually, thus introducing a relatively large relaxation gap. Although some recent works, such as [10], proposed k-ReLU relaxations to consider multiple ReLU relaxations jointly, the relaxation gap is still large. On the contrary, SDP-based relaxations naturally couple ReLU relaxations together without any additional effort via a semidefinite constraint. Therefore, as the number of hidden layers of the network under verification grows, the relative reduction in the relaxation gap also grows when compared to LP-based methods. However, even with the power of SDP relaxation, the relaxation gap is still significant under most settings. In this work, we address the issue with the SDP relaxation and shrink the relaxation gap using 2 different approaches, namely by partitioning the input space and applying nonconvex cuts.

In this report, we focus on feedforward ReLU networks, which are popular due to their simplicity, fast training speeds, and non-vanishing gradient property [3].

1.1 Partition-Based Certification

The first approach to reducing the relaxation gap is based on partitioning the input uncertainty set and solving simpler semidefinite programs over each input part. Partitioning heuristics have been applied in areas such as robust optimization [11] and deep learning [12], and are often found to tighten bounds on optimization objectives. Furthermore, partitioning naturally allows for parallelization of the optimization, resulting in computational advantages over centralized methods.

In this work, through the analysis of the geometric properties of the SDP relaxation, methods to partition the input space so that the relaxation gap can be reduced the most in a robust sense (i.e. in the worst case scenario) are proposed. The main source of inspiration of this technique is [13], and we extended the formalisms to adapt to SDP relaxations in this report.

Since this partitioning technique requires no added effort to implement, it can actually be used in harmony with other techniques, like the second technique being proposed in this report.

1.2 Certification with added Nonconvex Cuts

To reduce relaxation gap in convex relaxations, the main idea is to reduce the search space, since relaxation means finding a convex set that is a strict superset of the original feasible set and use this new convex set as the feasible set of the relaxed problem. Therefore, to tighten the relaxation (synonymous to reducing the relaxation gap), the primary concern is to make this overestimation as small as possible, and the only explicit way to handle this is to add in more constraints into the relaxed problem, while still making sure that the reduced search space remains a super set of that of the original problem.

One common technique of this constraint adding approach is called the reformulation-linearization technique (RLT) [14], and it adds in linearized quadratic constraints generated by multiplying existing linear constraints in the relaxation. However, as can be formally shown in the later sections, this approach is not going to work with the problem of neural network certification.

Towards this end, a method of adding nonconvex contraints into the relaxed problem to reduce relaxation gap is proposed in this report, which is shown to be theoretically and empirically effective.

This work initially appeared in [15].

2 Notations

We write the set of real-valued n-vectors as $\mathbb{R}^n$ and the set of real-valued $m \times n$ matrices as $\mathbb{R}^{m\times n}$. The set of $n \times n$ symmetric matrices with real elements is $\mathbb{S}^n$. For $X, Y \in \mathbb{R}^{m\times n}$, we write $X \leq Y$ to mean $X_{ij} \leq Y_{ij}$ for all $i \in \{1, 2, \dots, m\}$ and all $j \in \{1, 2, \dots, n\}$. We write the Hadamard (element-wise) product between X and Y as $X \odot Y$ and the Hadamard division of X by Y as $X \oslash Y$. Furthermore, for $f\colon \mathbb{R} \to \mathbb{R}$, we define $f(X)$ to be an $m \times n$ matrix whose (i, j) element is equal to $f(X_{ij})$ for all $i \in \{1, 2, \dots, m\}$ and all $j \in \{1, 2, \dots, n\}$. In particular, let the ReLU function be denoted as $\mathrm{ReLU}(\cdot) = \max\{0, \cdot\}$. We use $\mathbb{I}$ to denote the indicator function, i.e., $\mathbb{I}(A) = 1$ if event A holds and $\mathbb{I}(A) = 0$ if event A does not hold. Finally, we denote the n-vector of all ones by $\mathbf{1}_n$, and for an index set $\mathcal{I} \subseteq \{1, 2, \dots, n\}$, the symbol $\mathbf{1}_{\mathcal{I}}$ is used to denote the n-vector with $(\mathbf{1}_{\mathcal{I}})_i = 1$ for $i \in \mathcal{I}$ and $(\mathbf{1}_{\mathcal{I}})_i = 0$ for $i \in \mathcal{I}^c = \{1, 2, \dots, n\} \setminus \mathcal{I}$. Operator $\mathrm{diag}(\cdot)$ coverts its vector argument to a diagonal matrix. $\mathrm{idx}(A, a)$ denotes the position/index of element a in vector or matrix A (e.g. $\mathrm{idx}([1, 2, 3], 3) = 3$). $\dagger$ denotes the Penrose-Moore generalized inverse, and $\langle A, B\rangle$ denotes $\mathrm{tr}(A^\top B)$ for square matrices A, B.

3 Backgrounds

3.1 Multi-Layer Perceptron (MLP) ReLU Network

Consider a K-layer ReLU neural network defined by

$$x^{[0]} = x, \quad x^{[k]} = \mathrm{ReLU}(W^{[k-1]}x^{[k-1]}) \tag{1}$$

for all $k \in \{1, 2, \dots, K\}$, where $x \in \mathbb{R}^{n_x}$ is the input to the neural network, $z \triangleq x^{[K]} \in \mathbb{R}^{n_z}$ is the output, and $\hat{x}^{[k]} = W^{[k-1]}x^{[k-1]} + b^{[k-1]} \in \mathbb{R}^{n_k}$ is the preactivation of the k^{th} layer. The parameters $W^{[k]} \in \mathbb{R}^{n_{k+1}\times n_k}$ and $b^{[k]} \in \mathbb{R}^{n_{k+1}}$ are the weight matrix and bias vector applied to the k^{th} layer's activation $x^{[k]} \in \mathbb{R}^{n_k}$, respectively. Without loss of generality, assume that the bias terms are accounted for in the activations $x^{[k]}$, thereby setting $b^{[k]} = 0$ for all layers k. Let the function $f\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$ denote the mapping $x \mapsto z$ defined by (1).

Following the spirits of [1, 2], define the input uncertainty set $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ using l_∞ norms: $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon\}$. Similarly, define $\mathcal{S} \subseteq \mathbb{R}^{n_z}$ as the *safe set.* For classification networks, safe sets are (possibly unbounded) usually polyhedral sets defined as the intersection of a finite number of half-spaces:

$$\mathcal{S} = \{z \in \mathbb{R}^{n_z} : Cz \leq 0\},$$

where $C \in \mathbb{R}^{n_{\mathcal{S}}\times n_z}$ is given. Any output $z \in \mathcal{S}$ is said to be *safe.* The safe set is assumed to be polyhedral in the rest of this report.

3.2 Robustness Certification

The goal of certification is to ensure that $f(x) \in \mathcal{S}$ for all $x \in \mathcal{X}$, which is equivalent to checking the satisfaction of the following inequality:

$$\max_{i\in\{1,\dots,n_{\mathcal{S}}\}} \{f_i^*(\mathcal{X})\} \leq 0 \tag{2}$$

where $f_i^*(\mathcal{X}) = \sup\{c_i^\top z : z = f(x),\ x \in \mathcal{X}\}$ and $c_i^\top$ is the i^{th} row of C. Thus, the certification procedure amounts to solving an optimization problem corresponding to each c_i. For the remainder of this report, the objective is assumed to be $\sup_{x\in\mathcal{X}} c^\top f(x)$ for any generic c, since the generalization to the case $n_{\mathcal{S}} > 1$ is straightforward.

In general, the optimization $\sup_{x\in\mathcal{X}} c^\top f(x)$ is a nonconvex problem and $f(\mathcal{X})$ is a nonconvex set, and therefore computing a robustness certificate is intractable in polynomial time. To circumvent this issue, one can instead certify that a convex outer approximation of $f(\mathcal{X})$ is safe, as this inherently certifies the safety of the true nonconvex set $f(\mathcal{X})$, and hence certifies the robustness of the network. This process is illustrated in Fig. 1.

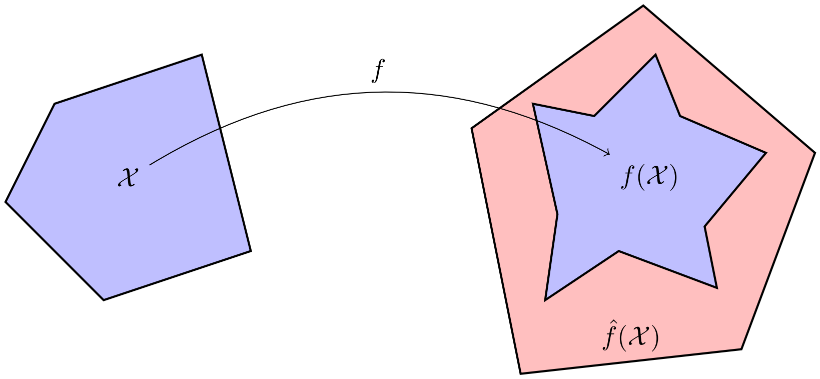


Figure 1: The convex outer approximation of the nonconvex set $f(\mathcal{X})$ is $\hat{f}(\mathcal{X})$. If the outer approximation is safe, i.e., $\hat{f}(\mathcal{X}) \subseteq \mathcal{S}$, then so is $f(\mathcal{X})$.

3.3 SDP Relaxation

In this section, we present the SDP relaxation used for robustness certification. The details can be found in [2]. The main idea is to convert the ReLU constraints to quadratic constraints and then reformulate the non-convex certification problem (2) as a quadratically-constrained quadratic program (QCQP). Then, the standard SDP relaxation of the resulting QCQP leads to checking whether the following inequality is satisfied:

$$\max_{i \in \{1,\ldots,n_{\mathcal{S}}\}} \{\hat{f}_i^*(\mathcal{X})\} \leq 0 \tag{3}$$

where $\hat{f}_i^*(\mathcal{X}) = \sup\{c_i^\top z : (x,z) \in \mathcal{N}_{\text{SDP}},\ x \in \mathcal{X}\}$. The notation $(x,z) \in \mathcal{N}_{\text{SDP}}$ means that there exist $\tilde{x} \in \mathbb{R}^{n_{\tilde{x}}}$ and $\tilde{X} \in \mathbb{S}^{n_{\tilde{x}}}$ with $z \triangleq \tilde{x}[x^{[K]}]$ such that

$$\tilde{x}[x^{[0]}] = x,\ \tilde{X} \succeq \tilde{x}\tilde{x}^\top,\ (\tilde{X},\tilde{x}) \in \mathcal{N}_{\text{SDP}}^{[k]}\ \forall k = \{1 \ldots K\} \tag{4}$$

where the membership condition $(\tilde{X},\tilde{x}) \in \mathcal{N}_{\text{SDP}}^{[k]}$ is defined by the following conditions:

$$\begin{aligned}
&\tilde{x}[x^{[k]}] \geq 0,\\
&\tilde{x}[x^{[k]}] \geq W^{[k-1]}\tilde{x}[x^{[k-1]}],\\
&\operatorname{diag}(\tilde{X}[x^{[k]}(x^{[k]})^T]) = \operatorname{diag}(W^{[k-1]}\tilde{X}[x^{[k-1]}(x^{[k]})^T]),\\
&\operatorname{diag}(\tilde{X}[x^{[k-1]}(x^{[k-1]})^T]) \leq (l^{[k-1]} + u^{[k-1]})\odot\\
&\tilde{x}[x^{[k-1]}] - l^{[k-1]} \odot u^{[k-1]},
\end{aligned} \tag{5}$$

where $n_{\tilde{x}} = \Sigma_{j=0}^{K} n_j$. $l^{[k]}$ and $u^{[k]}$ are the lower and upper bounds on $x^{[k]}$, respectively. In the following sections, especially in section 5, l and u usually denotes $l^{[0]}$ and $u^{[0]}$, thus bounds for the input set $\mathcal{X}$ unless specified otherwise. Given $\mathcal{X}$, only $l^{[0]}$ and $u^{[0]}$ are known. The indexing notation in this report is inherited from [2] to promote consistency. Namely, $a[x^{[k]}] = a[\Sigma_{j=0}^{k-1} n_j + 1 : \Sigma_{j=0}^{k} n_j]$ for any vector $a \in \mathbb{R}^{n_{\tilde{x}}}$ and $A[x^{[k]}(x^{[l]})^T] = A[\Sigma_{j=0}^{k-1} n_j + 1 : \Sigma_{j=0}^{k} n_j, \Sigma_{j=0}^{l-1} n_j + 1 : \Sigma_{j=0}^{l} n_j]$ for any matrix $A \in \mathbb{S}^{n_{\tilde{x}}}$.

For the sake of brevity, henceforth we use the shorthand notations $\tilde{x}[k] \triangleq \tilde{x}[x^{[k]}]$, $\tilde{X}[A_k] \triangleq \tilde{X}[x^{[k-1]}(x^{[k-1]})^T]$, $\tilde{X}[B_k] \triangleq \tilde{X}[x^{[k-1]}(x^{[k]})^T]$, $\tilde{X}[C_k] \triangleq \tilde{X}[x^{[k]}(x^{[k]})^T]$, and:

$$\tilde{X}[k] \triangleq \begin{bmatrix} \tilde{X}[A_k] & \tilde{X}[B_k] \\ \tilde{X}[B_k^\top] & \tilde{X}[C_k] \end{bmatrix} \tag{6}$$

Furthermore, define

$$P \triangleq \begin{bmatrix} 1 & \tilde{x}^\top \\ \tilde{x} & \tilde{X} \end{bmatrix} \tag{7}$$

Note that $\tilde{X} \succeq \tilde{x}\tilde{x}^\top$ if and only if $P \succeq 0$. P^* denotes the optimal P that minimizes $\max_{i\in\{1,\dots,n_S\}}\{\hat{f}_i^*(\mathcal{X})\}$

The above relaxation implies that if $\hat{f}_i^*(\mathcal{X}) \leq 0$, then it is guaranteed that $f_i^*(\mathcal{X}) \leq 0$. However, if $\hat{f}_i^*(\mathcal{X}) \geq 0$, it is impossible to conclude whether $f_i^*(\mathcal{X}) \geq 0$ or the relaxation is loose.

4 Tightness of SDP Relaxation

Compared to previous convex relaxation schemes (such as LP), SDP indeed yields a tighter lower bound [2]. However, according to the above report and the recent results in [16], SDP relaxations of Multi-Layer Perceptron (MLP) ReLU networks are not tight even for single-layer instances. This issue will be elaborated below.

Since P is positive-semidefinite, it can be decomposed as:

$$P = VV^\top, \quad \text{where } V = \begin{bmatrix} \vec{e} \\ \vec{x}_1^\top \\ \vec{x}_2^\top \\ \vdots \\ \vec{x}_{n_{\tilde{x}}}^\top \end{bmatrix} \in \mathbb{R}^{(n_{\tilde{x}}+1)\times r} \tag{8}$$

Each vector $\vec{x}_i$ has dimension r, which is the rank of P. Since $\vec{e}\cdot\vec{e} = 1$, $\vec{e}$ is a unit vector in $\mathbb{R}^r$. Furthermore, we have $\tilde{\vec{x}}_i = \vec{e}\cdot\vec{x}_i$ for $i \in \{1,\dots,\tilde{n}_x\}$. The constraints in $\mathcal{N}_{\text{SDP}}^{[k]}$ can be broken down into 2 parts:

1. *Input constraints.* The constraint $\text{diag}(\tilde{X}[A_k]) \leq (l^{[k-1]}+u^{[k-1]})\odot\tilde{x}[k-1]-l^{[k-1]}\odot u^{[k-1]}$ is equivalent to $\|\vec{x}_k\|^2 \leq (l^{[k-1]}+u^{[k-1]})\vec{x}_i\cdot e - l^{[k-1]}u^{[k-1]}$ for all $k \in \{1,2,\dots,n_x\}$. Geometrically, this constrains each vector in the set $\{\vec{x}_i | i \in \text{idx}(\tilde{x},\tilde{x}[k-1])\}$ to lie in a circle centered at $\frac{1}{2}(l_i+u_i)\vec{e}$ with radius $\frac{1}{2}(u_i - l_i)$. This can be seen as follows:

$$\begin{aligned} (\vec{x}_k - \tfrac{1}{2}(l^{[k-1]}+u^{[k-1]})e)^\top(\vec{x}_k - \tfrac{1}{2}(l^{[k-1]}+u^{[k-1]})e) &\leq (\tfrac{1}{2}(u^{[k-1]}-l^{[k-1]}))^2 \\ \iff \|\vec{x}_k\|_2^2 - (l^{[k-1]}+u^{[k-1]})\vec{x}_k\cdot e + \tfrac{1}{4}(l^{[k-1]}+u^{[k-1]})^2\|e\|_2^2 &\leq \tfrac{1}{4}((l^{[k-1]})^2 + (u^{[k-1]})^2 - u^{[k-1]}l^{[k-1]}) \\ \iff \|\vec{x}_k\|_2^2 &\leq (l^{[k-1]}+u^{[k-1]})\vec{x}_k\cdot e - l^{[k-1]}u^{[k-1]}. \end{aligned}$$

2. *ReLU constraints.* The constraint $\text{diag}(\tilde{X}[C_k]) = \text{diag}(W^{[k-1]}\tilde{X}[B_k])$ is equivalent to the condition that $\|\vec{x}_j\|^2 = \vec{\chi}_j\cdot\vec{x}_j$ for all $\{j | j \in \text{idx}(\tilde{x},\tilde{x}[k])\}$, where $\vec{\chi}_j = \sum_{i\in\text{idx}(\tilde{x},\tilde{x}[k-1])} w_{ij}\vec{x}_i$ and $\{w_{ij}\}_{i\in\text{idx}(\tilde{x},\tilde{x}[k-1])}$ is the j^{th} row of $W^{[k]}$. This implies $(\vec{x}_j - \vec{\chi}_j)\cdot\vec{x}_j = 0$, which translates to the geometric relationship of $\vec{x}_j$ lying *on* the circle with $\vec{\chi}_j$ as its diameter for all $\{j | j \in \text{idx}(\tilde{x},\tilde{x}[k])\}$ (recall the basic fact that angles formed by drawing lines from the ends of the diameter of a circle to its circumference form a right angle). The added linear constraints $\tilde{x}[k] \geq 0$ and $\tilde{x}[k] \geq W^{[k-1]}\tilde{x}[k-1]$ restricts the sphere to the subset where the projection of $\vec{x}_j$ on $\vec{e}$ is nonnegative and larger than the projection of $\vec{\chi}_j$ on $\vec{e}$.

As pointed out in Lemma 6.1 of [16], the SDP relaxation is tight if and only if $\vec{e}$ and all the vectors $\vec{x}_i$'s are collinear. Since there is no constraint on the angle between $\vec{x}_i$ and $\vec{e}$, the resulting spherical cap (as termed in Section 4 of the report) always exists, and the height of this cap will be an upper bound on the relaxation gap of the corresponding entry in $\tilde{x}$. The following lemma makes this claim more concrete:

Lemma 1 (Rank-1 gap)**.** *The rank-1 gap is nonnegative:*

$$g(P) = \sum_{i=1}^{n_{\tilde{x}}}(\|\vec{x}_i\|_2^2 - (\vec{x}_i\cdot\vec{e})^2) \geq 0 \tag{9}$$

Proof. Since $P \succeq 0$, Schur complements give that $\tilde{X} - \tilde{x}\tilde{x}^\top \succeq 0$, and taking the trace of the LHS we get that it has to be larger or equal to 0. The proof is concluded by noticing $\text{tr}(\tilde{X} - \tilde{x}\tilde{x}^\top) = \sum_{i=1}^{n_{\tilde{x}}}(\|\vec{x}_i\|_2^2 - (\vec{x}_i\cdot\vec{e})^2)$. □

Since $g(P) = 0$ is necessary for P to be rank-1, this is a partial result of Lemma 6.1 of [16], stating that if $\vec{e}$ and all the vectors $\vec{x}_i$'s are collinear, there will be a zero rank-1 gap, making the relaxation exact.

5 Partitioned SDP Relaxation

In this section, we explore how to partition the input space $\mathcal{X}$ so that the relaxation gap can be reduced. Before proceeding to the partitioning techniques, the motivation and notations of partitioning are made exact through the following proposition:

Proposition 1 (Improving the SDP relaxation bound)**.** *Consider a one-layer feedforward neural network. Let $\{\mathcal{X}^{(j)} : j \in \{1, 2, \dots, p\}\}$ be a partition of $\mathcal{X}$. For the j^{th} input part $\mathcal{X}^{(j)}$, denote the corresponding input bounds by $l \le l^{(j)} \le x \le u^{(j)} \le u$, where $x \in \mathcal{X}^{(j)}$ and l, u are assumed to be bounds for $\mathcal{X}$. Then it holds that*

$$\max_{j\in\{1,2,\dots,p\}} \hat{f}^*(\mathcal{X}^{(j)}) \le \hat{f}^*(\mathcal{X}). \tag{10}$$

Proof. Let $j \in \{1, 2, \dots, p\}$. From the definition of the partition, it holds that $\mathcal{X}^{(j)} \subseteq \mathcal{X}$. What remains to be shown is that $\mathcal{N}_{\text{SDP}}^{(j)} \subseteq \mathcal{N}_{\text{SDP}}$.

Let $P \in \mathcal{N}_{\text{SDP}}^{(j)}$. Define $u' = u^{(j)}$ and $l' = l^{(j)}$. Since $P \in \mathcal{N}_{\text{SDP}}^{(j)}$, it follows that

$$\begin{aligned} \tilde{x}[k] &\ge 0, \\ \tilde{x}[k] &\ge W^{[k-1]}\tilde{x}[k-1], \\ \operatorname{diag}(\tilde{X}[C_k]) &= \operatorname{diag}(W^{[k-1]}\tilde{X}[B_k]), \\ \operatorname{diag}(\tilde{X}[A_k]) &\le (l' + u') \odot \tilde{x}[k-1] - l' \odot u', \\ P_1 &= 1, \\ P &\succeq 0. \end{aligned}$$

To show that $P \in \mathcal{N}_{\text{SDP}}$, we should show that the above expressions imply that $\operatorname{diag}(\tilde{X}[A_k]) \le (l + u) \odot \tilde{x}[k-1] - l \odot u$. To do so, define $\Delta l_i \ge 0$ and $\Delta u_i \ge 0$ such that $l'_i = l_i + \Delta l_i$ and $u'_i = u_i - \Delta u_i$ for all $i \in \{1, 2, \dots, n_{k-1}\}$. Note the subscript i denotes the i^{th} entry of the vector. Then we find that

$$\begin{aligned} ((l' + u') \odot \tilde{x}[k-1] - l' \odot u')_i &= (l'_i + u'_i)(\tilde{x}[k-1])_i - l'_i u'_i \\ &= (l_i + u_i)(\tilde{x}[k-1])_i - l_i u_i + (\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= ((l + u) \odot \tilde{x}[k-1] - l \odot u)_i + (\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= ((l + u) \odot \tilde{x}[k-1] - l \odot u)_i + \Delta_i, \end{aligned}$$

where $\Delta_i \coloneqq (\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i)$. Therefore, it suffices to prove that $\Delta_i \le 0$ for all i. Since $-l_i u_i \ge -l'_i u'_i$ by definition, it holds that $\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i \ge 0$. Thus, when $(\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i \le 0$, it holds that $\Delta_i \le 0$, as desired. On the other hand, suppose that $(\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i \ge 0$. Then we find two cases:

1. $(\Delta l_i - \Delta u_i) \ge 0$ and $(\tilde{x}[k-1])_i \ge 0$. In this case, the maximum value of $(\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i$ is $(\Delta l_i - \Delta u_i)u'_i$. Therefore, the maximum value of Δ_i is

$$\begin{aligned} \Delta_i &= (\Delta l_i - \Delta u_i)u'_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= \Delta l_i(u'_i - u_i) - \Delta u_i u'_i + \Delta u_i l_i + \Delta u_i \Delta l_i \\ &= \Delta l_i(-\Delta u_i) + \Delta u_i \Delta l_i - \Delta u_i u'_i + \Delta u_i l_i \\ &= -\Delta u_i u'_i + \Delta u_i l_i. \end{aligned}$$

   Both of the two final terms are nonpositive, and therefore $\Delta_i \le 0$.

2. $(\Delta l_i - \Delta u_i) \le 0$ and $(\tilde{x}[k-1])_i \le 0$. In this case, the maximum value of $(\Delta l_i - \Delta u_i)(\tilde{x}[k-1])_i$ is $(\Delta l_i - \Delta u_i)l'_i$. Therefore, the maximum value of Δ_i is

$$\begin{aligned} \Delta_i &= (\Delta l_i - \Delta u_i)l'_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= -\Delta u_i \Delta l_i + \Delta u_i \Delta l_i + \Delta l_i l'_i - \Delta l_i u_i \\ &= \Delta l_i l'_i - \Delta l_i u_i. \end{aligned}$$

   Both of the two final terms are nonpositive, and therefore $\Delta_i \le 0$.

Hence, we find that $(l' + u') \odot \tilde{x}[k-1] - l' \odot u' \leq (l+u) \odot \tilde{x}[k-1] - l \odot u$ for all $P \in \mathcal{N}_{\text{SDP}}^{(j)}$, proving that $P \in \mathcal{N}_{\text{SDP}}$, and therefore $\mathcal{N}_{\text{SDP}}^{(j)} \subseteq \mathcal{N}_{\text{SDP}}$.

Since $\mathcal{X}^{(j)} \subseteq \mathcal{X}$ and $\mathcal{N}_{\text{SDP}}^{(j)} \subseteq \mathcal{N}_{\text{SDP}}$, it holds that the solution to the problem over the smaller feasible set lower bounds the original solution: $\hat{f}^*(\mathcal{X}^{(j)}) \leq \hat{f}^*(\mathcal{X})$. Finally, since j was chosen arbitrarily, this implies the desired inequality (10). □

From the above proposition it is clear that any form of partitioning could potentially reduce relaxation gap and that no harm could be incurred by doing so. Furthermore, this technique can be applied concurrently with other techniques, and can be easily parallelized, making it an ideal way to tighten the relaxation.

Thus in the remainder of this section, we will develop the optimal partitioning strategy for our verification problem.

5.1 Scaling of Objective via Partitioning

Before diving into the derivation of the optimal strategy, one natural question to ask is that if the input bounds are reduced by a factor of t, how much will the objective decrease? It could guide us in the choice of number of partitions when a specific reduction in objective is in mind.

When the weights of the neural network change, the exact scaling factor of the objective also changes, but when the weights are uniformly distributed, we show that the scaling in objective is linear in expectation.

Note that, for all $i \in \text{idx}(\tilde{x}, \tilde{x}[k-1]), j \in \text{idx}(\tilde{x}, \tilde{x}[k])$, for all $k \in \{1, \ldots, K\}$, since every $\vec{x}_i$ is bounded within a ball, $\vec{\chi}_j$ is also bounded within a ball given that $\|W\|_\infty < \infty$, namely one of radius $\check{r}_j$ and center $\check{c}_j$. Those parameters can be arbitrarily chosen so that all possible $\vec{\chi}_j$ are covered.

Now, let $h_j = (\vec{\chi}_j)^\top e$ and denote by $\theta_j \in [0, 2\pi]$ the angle between $\vec{\chi}_j$ and e. Then h_j and θ_j uniquely determine $\vec{\chi}_j$ for a given e. Consequently, for fixed h_j and θ_j, the largest possible relaxation error in coordinate j is given by $\sup_{\{\vec{x}_j : P \in \mathcal{N}_{\text{SDP}},\ e,h,\theta \text{ fixed}\}} (\vec{x}_j - \vec{\chi}_j)^\top e$. To simplify calculations in the following sections, we introduce a parameter d_j to parametrize θ_j such that $d_j = \|\vec{\chi}_j\|_2$. This is obvious since $d_j = \frac{h_j}{cos(\theta_j)}$. Using similar triangles, it is straightforward to show that $\sup_{\{\vec{x}_j : P \in \mathcal{N}_{\text{SDP}},\ e,h,\theta \text{ fixed}\}} \vec{x}_j^\top e = \frac{1}{2} h_j + \frac{1}{2} d_j$. Therefore, for fixed h_j and d_j it holds that the maximum relaxation error in coordinate j is

$$\epsilon(h_j, d_j) := \sup_{\{\vec{x}_j : P \in \mathcal{N}_{\text{SDP}},\ e,h,d \text{ fixed}\}} (\vec{x}_j - \vec{\chi}_j)^\top e = \frac{1}{2} h_j + \frac{1}{2} d_j - h_j = \frac{1}{2} d_j - \frac{1}{2} h_j.$$

We now consider the effect of partitioning, namely, tightening the input bounds l, u. In particular, consider the special scenario in which the input bounds are decreased to $l' = tl$ and $u' = tu$ for some $t \in (0, 1)$. Then the input constraint moves $\vec{\chi}_j$ from a ball centered at $\frac{1}{2}(u+l)e$ with radius $\frac{1}{2}(u-l)$ to a ball centered at $\frac{t}{2}(u+l)$ with radius $\frac{t}{2}(u-l)$. Effectively, the input constraint ball on $\vec{\chi}_j$ moves closer to the origin with decreasing radius, both effects changing linearly with t.

To quantify the change of $\epsilon(h_j, d_j)$ with respect to t, we need to have some kind of assumption on the distribution of h. Namely, we assume that $h_j \sim \mathcal{U}(\check{c}_j + \check{r}_j, \check{c}_j - \check{r}_j)$, which is the uniform distribution, since we don't have a-priori information on the problem. In expectation, $\mathbb{E}[h_j] = \check{c}_j$. As for d_j, it depends on h_j, and we again assume uniform distribution, namely $[d_j | h_j] \sim \mathcal{U}(h_j, \sqrt{\check{r}_j^2 - \check{c}_j^2 + 2\check{c}_j h_j})$. This is evident from figure 2 since $\sqrt{\check{r}_j^2 - (\check{c}_j - h_j)^2 + h_j^2} = \sqrt{\check{r}_j^2 - \check{c}_j^2 + 2\check{c}_j h_j}$. This bound is valid since regardless of the direction of $\vec{\chi}_j$ vector, for the same $\|\vec{\chi}_j\|_2$, $\epsilon(h_j, d_j)$ stays constant. Therefore, in expectation:

$$\begin{aligned}
\mathbb{E}[\epsilon(h_j, d_j)] &= \frac{1}{2}\int_{\breve{c}_j-\breve{r}_j}^{\breve{c}_j+\breve{r}_j} \frac{1}{2\breve{r}_j}\int_{h_j}^{\sqrt{\breve{r}_j^2-\breve{c}_j^2+2\breve{c}_j h_j}} \frac{1}{\sqrt{\breve{r}_j^2-\breve{c}_j^2+2\breve{c}_j h_j}-h_j}(d_j-h_j)\mathrm{d}d_j\mathrm{d}h_j \\
&= \frac{1}{2}\int_{\breve{c}_j-\breve{r}_j}^{\breve{c}_j+\breve{r}_j} \frac{1}{2\breve{r}_j}(\frac{\sqrt{\breve{r}_j^2-\breve{c}_j^2+2\breve{c}_j h_j}+h_j}{2}-h_j)\mathrm{d}h_j \\
&= \frac{1}{2}\int_{\breve{c}_j-\breve{r}_j}^{\breve{c}_j+\breve{r}_j} \frac{1}{2\breve{r}_j}(\frac{\sqrt{\breve{r}_j^2-\breve{c}_j^2+2\breve{c}_j h_j}}{2})\mathrm{d}h_j - \frac{\breve{c}_j}{4} \\
&= \frac{(\breve{r}_j^2+\breve{c}_j^2+2\breve{r}_j\breve{c}_j)^{\frac{3}{2}}-(\breve{r}_j^2+\breve{c}_j^2-2\breve{r}_j\breve{c}_j)^{\frac{3}{2}}}{24\breve{r}_j\breve{c}_j} - \frac{\breve{c}_j}{4}
\end{aligned} \tag{11}$$

After the scaling t is applied to l, u, denote the new $\breve{c}_j, \breve{r}_j, h_j, d_j$ as $\breve{c}'_j, \breve{r}'_j, h'_j, d'_j$. Then, $\breve{c}'_j = t\breve{c}_j, \breve{r}'_j = \breve{t}r_j$. Therefore

$$\begin{aligned}
\mathbb{E}[\epsilon(h'_j, d'_j)] &= \frac{((\breve{r}'_j)^2+(\breve{c}'_j)^2+2\breve{r}'_j\breve{c}'_j)^{\frac{3}{2}}-((\breve{r}'_j)^2+(\breve{c}'_j)^2-2\breve{r}'_j\breve{c}'_j)^{\frac{3}{2}}}{24\breve{r}'_j\breve{c}'_j} - \frac{\breve{c}'_j}{4} \\
&= t(\frac{(\breve{r}_j^2+\breve{c}_j^2+2\breve{r}_j\breve{c}_j)^{\frac{3}{2}}-(\breve{r}_j^2+\breve{c}_j^2-2\breve{r}_j\breve{c}_j)^{\frac{3}{2}}}{24\breve{r}_j\breve{c}_j} - \frac{\breve{c}_j}{4}) \\
&= t\mathbb{E}[\epsilon(h_j, d_j)]
\end{aligned} \tag{12}$$

Since $\tilde{x}_j = \frac{1}{2}h_j + \frac{1}{2}d_j$, utilizing the same arguments as above we also know that $\tilde{x}_j$ scales linearly with t.

5.2 Partitioning Strategy

5.2.1 Worst-Case Rank-1 Bound

We want to partition the input uncertainty set to minimize $g(P^*)$, in hopes to influence P^* to be of low rank. However, there is a major hurdle with this approach. In particular, the optimal solution P^* depends on the partition we choose, and finding a partition to minimize $g(P^*)$ in turn depends on P^* itself. To overcome this cyclic dependence, we propose first bounding $g(P^*)$ by a worst-case sense upper bound. As we will see, choosing an optimal partition to minimize the upper bound makes the partition design much more tractable, resulting in a closed-form solution.

To derive the upper bound on the rank-1 gap at optimality, let $\{\mathcal{X}^{(j)} : j \in \{1, 2, \ldots, p\}\}$ denote the partition of $\mathcal{X}$. For the j^{th} input part $\mathcal{X}^{(j)}$, denote the corresponding input bounds by $l^{(j)}, u^{(j)}$. The upper bound is given in the following lemma.

Lemma 2 (Rank-1 gap upper bound)**.** *The rank-1 gap at the solution P^* of the partitioned SDP satisfies*

$$0 \leq g(P^*) \leq \frac{1}{4}\sum_{i=1}^{n_x} \max_{j\in\{1,2,\ldots,p\}} (u_i^{(j)} - l_i^{(j)})^2. \tag{13}$$

Proof. The left inequality is a direct result of Lemma 1. For the right inequality, note that

$$\begin{aligned}
g(P^*) &\leq \max_{j\in\{1,2,\ldots,p\}} \max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} g(P) \\
&\leq \sum_{i=1}^{n_x} \max_{j\in\{1,2,\ldots,p\}} \max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} (\|\vec{x}_i\|_2^2 - (\vec{x}_i^\top e)^2).
\end{aligned} \tag{14}$$

Let us focus on the optimization over the j^{th} part of the partition, namely,

$$\max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} (\|\vec{x}_i\|_2^2 - (\vec{x}_i^\top e)^2).$$

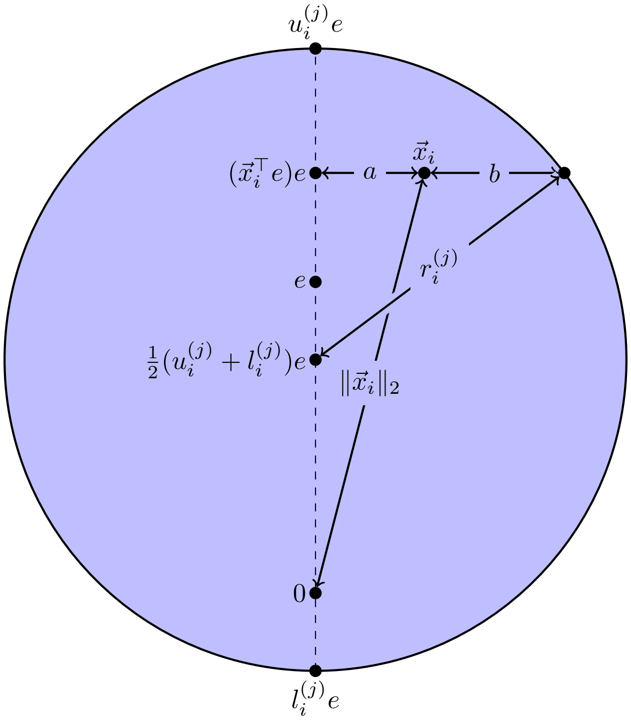


Figure 2: Geometry of the SDP relaxation in coordinate i over part j of the partition. The shaded region shows the feasible $\vec{x}_i$ satisfying the input constraint. [2]

To bound this quantity, we analyze the geometry of the SDP relaxation over part j, following the methodology of [2]; see Figure 2.

The shaded circle represents the set of feasible $\vec{x}_i$ over part j of the partition, namely, those satisfying the i^{th} coordinate of the constraint $\operatorname{diag}(\tilde{X}[A_k]) \leq (l^{(j)}+u^{(j)}) \odot \tilde{x}[k-1] - l^{(j)} \odot u^{(j)}$. To see this, note that the constraint is equivalent to $\|\vec{x}_i\|_2^2 \leq (l_i^{(j)}+u_i^{(j)})\vec{x}_i^\top e - l_i^{(j)}u_i^{(j)}$, or, more geometrically written, that $\|\vec{x}_i - \frac{1}{2}(u_i^{(j)}+l_i^{(j)})e\|_2^2 \leq \left(\frac{1}{2}(u_i^{(j)}-l_i^{(j)})\right)^2$. This shows that $\vec{x}_i$ is constrained to a 2-norm ball of radius $r_i^{(j)} = \frac{1}{2}(u_i^{(j)}-l_i^{(j)})$ centered at $\frac{1}{2}(u_i^{(j)}+l_i^{(j)})e$, as shown in Figure 2.

The geometry of Figure 2 immediately shows that $\|\vec{x}_i\|_2^2 = a^2 + (\vec{x}_i^\top e)^2$ and $r_i^{(j)2} = (a+b)^2 + (\vec{x}_i^\top e - \frac{1}{2}(u_i^{(j)}+l_i^{(j)}))^2$, and therefore

$$\begin{aligned}\|\vec{x}_i\|_2^2 - (\vec{x}_i^\top e)^2 &= a^2 \\ &= r_i^{(j)2} - (\vec{x}_i^\top e - \tfrac{1}{2}(u_i^{(j)}+l_i^{(j)}))^2 - 2ab - b^2.\end{aligned}$$

Since a and b are nonnegative,

$$\begin{aligned}\max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} \|\vec{x}_i\|_2^2 - (\vec{x}_i^\top e)^2 &= \max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} r_i^{(j)2} - (\vec{x}_i^\top e - \tfrac{1}{2}(u_i^{(j)}+l_i^{(j)}))^2 - 2ab - b^2 \\ &\leq \max_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ \tilde{x}[k-1]\in\mathcal{X}^{(j)}} r_i^{(j)2} - (\vec{x}_i^\top e - \tfrac{1}{2}(u_i^{(j)}+l_i^{(j)}))^2 \\ &\leq r_i^{(j)2} \\ &= \frac{1}{4}(u_i^{(j)}-l_i^{(j)})^2.\end{aligned}$$

Thus, (14) gives

$$g(P^*) \leq \frac{1}{4}\sum_{i=1}^{n_x} \max_{j\in\{1,2,\dots,p\}} (u_i^{(j)}-l_i^{(j)})^2,$$

as desired. □

5.2.2 Form of Partitioning

With Lemma 2 in place, we now have an upper bound on the rank-1 gap at optimality, in terms of the input bounds $\{l^{(j)},u^{(j)}\}_{j=1}^p$ associated with the partition. At this point, we now turn to minimizing the upper bound over all valid choices of p-part partitions of the input uncertainty set. Note that, in order for $\{l^{(j)},u^{(j)}\}_{j=1}^p$ to define valid input bounds for a p-part partition, it must be that the union of the input parts cover the input uncertainty set. In terms of the input bounds, this leads to the constraint that $[l_i,u_i]\subseteq\bigcup_{j=1}^p[l_i^{(j)},u_i^{(j)}]$ for all $i\in\{1,2,\dots,n_x\}$. We now give the optimal partitioning scheme for the SDP that minimizes the upper bound of Lemma 2.

Theorem 1 (Optimal SDP partition via rank-1 gap)**.** *Consider the optimization problem of finding the partition to minimize the upper bound* (13)*:*

$$\begin{array}{lll}\underset{\mathcal{P}=\{l^{(j)},u^{(j)}\}_{j=1}^p\subseteq\mathbb{R}^{n_x}}{\text{minimize}} & h(\mathcal{P})=\displaystyle\sum_{i=1}^{n_x}\max_{j\in\{1,2,\dots,p\}}(u_i^{(j)}-l_i^{(j)})^2 & \\ \text{subject to} & [l_i,u_i]\subseteq\displaystyle\bigcup_{j=1}^p[l_i^{(j)},u_i^{(j)}], & (15)\\ & l_i^{(j)}\le u_i^{(j)}, & \\ \text{for all} & i\in\{1,2,\dots,n_x\},\ j\in\{1,2,\dots,p\}. & \end{array}$$

Consider the uniform partition defined by $\bar{\mathcal{P}}=\{\bar{l}^{(j)},\bar{u}^{(j)}\}_{j=1}^p\subseteq\mathbb{R}^{n_x}$, where

$$\begin{aligned}\bar{l}^{(j)}&=\frac{j-1}{p}(u-l)+l,\\ \bar{u}^{(j)}&=\frac{j}{p}(u-l)+l,\end{aligned}$$

for all $j\in\{1,2,\dots,p\}$. It holds that $\bar{\mathcal{P}}$ is a solution to (15)*.*

Proof. To prove the result, we show that the proposed $\bar{\mathcal{P}}$ is feasible for the optimization, and that $h(\bar{\mathcal{P}})\le h(\mathcal{P})$ for all feasible $\mathcal{P}$. First, note that

$$\bar{u}^{(j)}=\frac{j}{p}(u-l)+l=\frac{(j+1)-1}{p}(u-l)+l=\bar{l}^{(j+1)},$$

and therefore $[\bar{l}_i^{(j)},\bar{u}_i^{(j)}]\bigcup[\bar{l}_i^{(j+1)},\bar{u}_i^{(j+1)}]=[\bar{l}_i^{(j)},\bar{u}_i^{(j+1)}]$ for all $i\in\{1,2,\dots,n_x\}$. Hence,

$$\bigcup_{j=1}^p[\bar{l}_i^{(j)},\bar{u}_i^{(j)}]=[\bar{l}_i^{(1)},\bar{u}_i^{(p)}]=[l_i,u_i].$$

Furthermore, $\bar{u}^{(j)}-\bar{l}^{(j)}=\frac{1}{p}(u-l)\ge 0$, and therefore we conclude that $\bar{\mathcal{P}}=\{\bar{l}^{(j)},\bar{u}^{(j)}\}_{j=1}^p$ is feasible.

Computing the objective at the proposed feasible point,

$$\begin{aligned}h(\bar{\mathcal{P}})&=\sum_{i=1}^{n_x}\max_{j\in\{1,2,\dots,p\}}(\bar{u}_i^{(j)}-\bar{l}_i^{(j)})^2\\ &=\sum_{i=1}^{n_x}\max_{j\in\{1,2,\dots,p\}}(\tfrac{1}{p}(u_i-l_i))^2\\ &=\frac{1}{p^2}\sum_{i=1}^{n_x}(u_i-l_i)^2.\end{aligned}$$

Now, let $\mathcal{P} = \{l^{(j)}, u^{(j)}\}_{j=1}^{p}$ be an arbitrary feasible point for the optimization (15). Then

$$\begin{aligned} h(\mathcal{P}) &= \sum_{i=1}^{n_x} \max_{j\in\{1,2,\dots,p\}} (u_i^{(j)} - l_i^{(j)})^2 \\ &= \sum_{i=1}^{n_x} \left(\max_{j\in\{1,2,\dots,p\}} (u_i^{(j)} - l_i^{(j)}) \right)^2 \\ &\geq \sum_{i=1}^{n_x} \left(\frac{1}{p} \sum_{j=1}^{p} (u_i^{(j)} - l_i^{(j)}) \right)^2 \\ &= \frac{1}{p^2} \sum_{i=1}^{n_x} \left(\sum_{j=1}^{p} (u_i^{(j)} - l_i^{(j)}) \right)^2 . \end{aligned}$$

Since $\mathcal{P}$ is feasible, it holds that $[l_i, u_i] \subseteq \bigcup_{j=1}^{p} [l_i^{(j)}, u_i^{(j)}]$ for all $i \in \{1, 2, \dots, n_x\}$. Therefore, by the monotonicity and subadditivity of the Lebesgue measure μ on $\mathbb{R}$,

$$u_i - l_i = \mu([l_i, u_i]) \leq \mu \left(\bigcup_{j=1}^{p} [l_i^{(j)}, u_i^{(j)}] \right) \leq \sum_{j=1}^{p} \mu([l_i^{(j)}, u_i^{(j)}]) = \sum_{j=1}^{p} (u_i^{(j)} - l_i^{(j)}).$$

Substituting this into our above expressions, we conclude that

$$h(\bar{\mathcal{P}}) = \frac{1}{p^2} \sum_{i=1}^{n_x} (u_i - l_i)^2 \leq \frac{1}{p^2} \sum_{i=1}^{n_x} \left(\sum_{j=1}^{p} (u_i^{(j)} - l_i^{(j)}) \right) \leq h(\mathcal{P})$$

for all feasible $\mathcal{P}$, i.e., $\bar{\mathcal{P}} = \{\bar{l}^{(j)}, \bar{u}^{(j)}\}_{j=1}^{p}$ is a solution to the optimization. □

Theorem 1 shows that by choosing the partition of the input set to be uniformly divided amongst the p parts, we are choosing an optimal partition that minimizes the worst-case bound on the gap of the rank-1 necessary condition (9). This gives a well-motivated, yet simple way to design a partition of the input uncertainty set in order to best push the SDP relaxation towards being rank-1, thereby reducing relaxation error.

5.3 Best Axis of Partitioning

In this section, we turn our attention from the *form* of an optimal SDP partition to the *coordinate* of an optimal partition. In particular, we seek to find the best two-part partition to minimize relaxation error of the SDP. The results of Section 5.2 suggest using a uniform partition, and in this section we seek to find which coordinate to apply the partitioning to. Similar to the LP relaxation, we derive an optimal partitioning scheme by first bounding the relaxation error in the worst-case sense.

5.3.1 Worst-Case Relaxation Bound

We start with a short lemma which will be used in the proof of Theorem 2.

Lemma 3 (Bound on elements of PSD matrices)**.** *Let $P \in \mathbb{S}^n$ be a positive semidefinite matrix. Then $|P_{ij}| \leq \frac{1}{2}(P_{ii} + P_{jj})$ for all $i, j \in \{1, 2, \dots, n\}$.*

Proof. Define $x \in \mathbb{R}^n$ by

$$x_k = \begin{cases} 1 & \text{if } k = i, \\ -1 & \text{if } k = j, \\ 0 & \text{if } k \notin \{i, j\}, \end{cases}$$

for all $k \in \{1, 2, \dots, n\}$. Evaluating the quadratic form of P at x gives

$$x^\top P x = \sum_{k,l \in \{1,2,\dots,n\}} x_k x_l P_{kl} = P_{ii} - 2P_{ij} + P_{jj} \geq 0,$$

implying that

$$P_{ij} \leq \frac{1}{2}(P_{ii} + P_{jj}). \tag{16}$$

On the other hand, define $y \in \mathbb{R}^n$ by

$$y_k = \begin{cases} 1 & \text{if } k \in \{i, j\}, \\ 0 & \text{if } k \notin \{i, j\}, \end{cases}$$

for all $k \in \{1, 2, \dots, n\}$. Evaluating the quadratic form of P at y gives

$$y^\top P y = \sum_{k,l \in \{1,2,\dots,n\}} y_k y_l P_{kl} = P_{ii} + 2P_{ij} + P_{jj} \geq 0,$$

implying that

$$P_{ij} \geq -\frac{1}{2}(P_{ii} + P_{jj}). \tag{17}$$

Inequalities (16) and (17) together imply the desired bound. □

Assumption 1 (Normalized rows)**.** In Theorem 2 below, we assume without loss of generality that the rows of the weight matrix have been normalized in the ℓ_1-norm sense, i.e., that $\|w_i\|_1 = 1$ for all $i \in \{1, 2, \dots, n_z\}$. In the case that this assumption doesn't hold, the network architecture can be rescaled as follows:

$$\begin{aligned} z &= \operatorname{ReLU}(Wx) \\ &= \operatorname{ReLU}\left(\begin{bmatrix} w_1^\top \\ w_2^\top \\ \vdots \\ w_{n_z}^\top \end{bmatrix} x \right) \\ &= \operatorname{ReLU}\left(\operatorname{diag}(\|w_1\|_1, \|w_2\|_1, \dots, \|w_{n_z}\|_1) \begin{bmatrix} \frac{w_1^\top}{\|w_1\|_1} \\ \frac{w_2^\top}{\|w_2\|_1} \\ \vdots \\ \frac{w_{n_z}^\top}{\|w_{n_z}\|_1} \end{bmatrix} x \right) \\ &= W_{\text{scale}} \operatorname{ReLU}(W_{\text{norm}} x), \end{aligned}$$

where $W_{\text{scale}} = \operatorname{diag}(\|w_1\|_1, \|w_2\|_1, \dots, \|w_{n_z}\|_1) \in \mathbb{R}^{n_z \times n_z}$ and

$$W_{\text{norm}} = \begin{bmatrix} \frac{w_1^\top}{\|w_1\|_1} \\ \frac{w_2^\top}{\|w_2\|_1} \\ \vdots \\ \frac{w_{n_z}^\top}{\|w_{n_z}\|_1} \end{bmatrix} \in \mathbb{R}^{n_z \times n_x}$$

are the scaling and normalized factors of the weight matrix W, respectively. The scaling factor can therefore be absorbed into the cost vector c, yielding a problem with normalized rows as desired.

Theorem 2 (Worst-case relaxation bound for SDP)**.** *Consider a one-layer feedforward neural network with the input uncertainty set $\mathcal{X}$. Assume the network weights are scaled according to Assumption 1. Let the network have input bounds $l, u \in \mathbb{R}^{n_x}$ and preactivation bounds $\hat{l}, \hat{u} \in \mathbb{R}^{n_z}$. Consider also the relaxation error $\Delta f^*_{SDP}(\mathcal{X}) \coloneqq \hat{f}^*(\mathcal{X}) - f^*(\mathcal{X})$. Let P^* and (x^*, z^*) be optimal solutions for the relaxation $\hat{f}^*(\mathcal{X})$ and* *the unrelaxed problem $f^*(\mathcal{X})$, respectively. Also, let $\|\cdot\|$ be a norm on $\mathbb{R}^{n_x}$. There exists $\epsilon \in \mathbb{R}$ such that $\|P_x^* - x^*\| \leq \epsilon$ and*

$$\Delta f_{SDP}^*(\mathcal{X}) \leq \sum_{i=1}^{n_z} \bigg(\mathrm{ReLU}(c_i) q(l,u) + \mathrm{ReLU}(-c_i) \min\{\hat{u}_i, \epsilon \|w_i\|_*\} \bigg), \tag{18}$$

where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$, and where

$$q(l,u) = \max_{k \in \{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}.$$

Proof. First, since $\mathcal{X}$ is bounded, there exists $\epsilon \geq 0$ such that $\|\tilde{x}[0]^* - x^*\| \leq \epsilon$. The definitions of $\tilde{x}[0]^*$ and (x^*, z^*) give that

$$\Delta f_{\mathrm{SDP}}^*(\mathcal{X}) = \sum_{i=1}^{n_z} c_i((\tilde{x}[1]^*)_i - z_i^*) \leq \sum_{i=1}^{n_z} \Delta f_i^*, \tag{19}$$

where

$$\Delta f_i^* = \sup \bigg\{ c_i((\tilde{x}[1])_i - z_i) : z_i = \mathrm{ReLU}(w_i^\top x),\ \tilde{x}[1] \geq 0,\ \tilde{x}[1] \geq W\tilde{x}[0], \\ \mathrm{diag}(\tilde{X}[C_1]) = \mathrm{diag}(W\tilde{X}[B_1]),\ P_1 = 1,\ P \succeq 0,\ \|\tilde{x}[0] - x\| \leq \epsilon,\ x, \tilde{x}[0] \in \mathcal{X} \bigg\}$$

for all $i \in \{1,2,\dots,n_z\}$. Defining the auxiliary variables $P_{\hat{z}} = W\tilde{x}[0]$ and $\hat{z} = Wx$, this is equivalent to

$$\Delta f_i^* = \sup \bigg\{ c_i((\tilde{x}[1])_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{x}[1] \geq 0,\ \tilde{x}[1] \geq P_{\hat{z}},\ \mathrm{diag}(\tilde{X}[C_1]) = \mathrm{diag}(W\tilde{X}[B_1]), \\ P_1 = 1,\ P \succeq 0,\ \|\tilde{x}[0] - x\| \leq \epsilon,\ P_{\hat{z}} = W\tilde{x}[0],\ \hat{z}_i = w_i^\top x,\ x, \tilde{x}[0] \in \mathcal{X} \bigg\}.$$

If $x, \tilde{x}[0] \in \mathcal{X}$ and $\hat{z}, P_{\hat{z}}$ satisfy $\hat{z} = Wx$, $P_{\hat{z}} = W\tilde{x}[0]$, and $\|\tilde{x}[0] - x\| \leq \epsilon$, then $|(P_{\hat{z}})_i - \hat{z}_i| = |w_i^\top(\tilde{x}[0] - x)| \leq \|w_i\|_* \|\tilde{x}[0] - x\| \leq \epsilon \|w_i\|_*$ for all $i \in \{1,2,\dots,n_z\}$ by the Cauchy-Schwarz inequality for dual norms. Therefore,

$$\Delta f_i^* \leq \sup \bigg\{ c_i((\tilde{x}[1])_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{x}[1] \geq 0,\ \tilde{x}[1] \geq P_{\hat{z}},\ \mathrm{diag}(\tilde{X}[C_1]) = \mathrm{diag}(W\tilde{X}[B_1]), \\ P_1 = 1,\ P \succeq 0,\ \hat{l} \leq \hat{z}, P_{\hat{z}} \leq \hat{u},\ |(P_{\hat{z}})_k - \hat{z}_k| \leq \epsilon \|w_k\|_* \text{ for all } k \in \{1,2,\dots,n_z\},\ \hat{z}, P_{\hat{z}} \in \mathbb{R}^{n_z} \bigg\}.$$

We now translate the optimization variables in the above problem from $\hat{z} \in \mathbb{R}^{n_z}$ and $P \in \mathbb{R}^{1+n_x+n_z}$ to $z_i, (P_z)_i \in \mathbb{R}$. To this end, we note that if P is feasible for the above supremum, then

$$\begin{aligned} \mathrm{diag}(\tilde{X}[C_1])_i &= \mathrm{diag}(W\tilde{X}[B_1])_i \\ &= w_i^\top (\tilde{X}[B_1])_i \\ &\leq \|(\tilde{X}[B_1])_i\|_\infty \|w_i\|_1, \end{aligned}$$

where $(\tilde{X}[B_1])_i$ is the ith column of $\tilde{X}[B_1]$, and the inequality again comes from Cauchy-Schwarz. By the weight matrix scaling assumption, this yields

$$\mathrm{diag}(\tilde{X}[C_1])_i \leq \|(\tilde{X}[B_1])_i\|_\infty.$$

Now, since P is positive semidefinite, Lemma 3 gives that

$$\begin{aligned}\|(\tilde{X}[B_1])_i\|_\infty &= \max_{k\in\{1,2,\dots,n_x\}} |(\tilde{X}[B_1])_i|_k \\ &= \max_{k\in\{1,2,\dots,n_x\}} |(\tilde{X}[B_1])_{ki}| \\ &\leq \max_{k\in\{1,2,\dots,n_z\}} \frac{1}{2}\bigg((\tilde{X}[A_1])_{kk} + (\tilde{X}[C_1])_{ii}\bigg) \\ &= \frac{1}{2}(\tilde{X}[C_1])_{ii} + \frac{1}{2}\max_{k\in\{1,2,\dots,n_x\}} (\tilde{X}[A_1])_{kk}.\end{aligned}$$

Noting that $(\tilde{X}[C_1])_{ii} = \mathrm{diag}(\tilde{X}[C_1])_i$, the bound of interest becomes

$$\mathrm{diag}(\tilde{X}[C_1])_i \leq \max_{k\in\{1,2,\dots,n_z\}} (\tilde{X}[A_1])_{kk}.$$

We now seek to bound $(\tilde{X}[A_1])_{kk}$. Recall that $(\tilde{X}[A_1])_{kk} = \mathrm{diag}(\tilde{X}[A_1])_k \leq (l_k + u_k)(\tilde{x}[0])_k - l_k u_k$. If $(l_k + u_k) \geq 0$, then $(\tilde{x}[0])_k \leq u_k$ implies that $(l_k + u_k)(\tilde{x}[0])_k \leq (l_k + u_k)u_k$, and therefore $(\tilde{X}[A_1])_{kk} \leq (l_k + u_k)u_k - l_k u_k = u_k^2$. On the other hand, if $(l_k + u_k) < 0$, then $(\tilde{x}[0])_k \geq l_k$ implies that $(l_k + u_k)(\tilde{x}[0])_k \leq (l_k + u_k)l_k$, and therefore $(\tilde{X}[A_1])_{kk} \leq (l_k + u_k)l_k - l_k u_k = l_k^2$. Hence, in all cases, it holds that

$$(\tilde{X}[A_1])_{kk} \leq \mathbb{I}(l_k + u_k \geq 0)u_k^2 + \mathbb{I}(l_k + u_k < 0)l_k^2.$$

We can further simplify this bound as follows. If $l_k + u_k \geq 0$, then $u_k \geq -l_k$ and $u_k \geq l_k$, implying $|l_k| \leq u_k$, so $l_k^2 \leq u_k^2$ and therefore $u_k^2 = \max\{l_k^2, u_k^2\}$. On the other hand, if $l_k + u_k < 0$, then an analogous argument shows that $l_k^2 = \max\{l_k^2, u_k^2\}$. Hence, we conclude that the above bound on $(\tilde{X}[A_1])_{kk}$ can be rewritten as

$$(\tilde{X}[A_1])_{kk} \leq \max\{l_k^2, u_k^2\}.$$

Therefore, returning to the bound on $(\tilde{X}[C_1])_i$, we find that

$$\mathrm{diag}(\tilde{X}[C_1])_i \leq \max_{k\in\{1,2,\dots,n_x\}} \max\{l_k^2, u_k^2\},$$

for all $i \in \{1, 2, \dots, n_z\}$. Now, note that since $P \succeq 0$, Schur complements give that

$$\begin{bmatrix} \tilde{X}[A_1] - \tilde{x}[0]\tilde{x}[0]^\top & \tilde{X}[B_1] - \tilde{x}[0]\tilde{x}[1]^\top \\ \tilde{X}[B_1]^\top - \tilde{x}[1]\tilde{x}[0]^\top & \tilde{X}[C_1] - \tilde{x}[1]\tilde{x}[1]^\top \end{bmatrix} \succeq 0,$$

which implies that

$$\mathrm{diag}(\tilde{X}[C_1]) \geq \mathrm{diag}(\tilde{x}[1]\tilde{x}[1]^\top) = \tilde{x}[1] \odot \tilde{x}[1].$$

Therefore, our upper bound on the diagonal elements of $\tilde{X}[C_1]$ yields that

$$(\tilde{x}[1])_i \leq \max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\} = q(l, u).$$

Hence, we have derived a condition on the component $(\tilde{x}[1])_i$ that all feasible P must satisfy. The supremum of interest may now be further upper bounded giving rise to

$$\begin{aligned}\Delta f_i^* \leq \sup\bigg\{ & c_i((\tilde{x}[1])_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ (\tilde{x}[1])_i \geq 0,\ (\tilde{x}[1])_i \geq (P_{\hat{z}})_i,\ (\tilde{x}[1])_i \leq q(l, u), \\ & \hat{l}_i \leq \hat{z}_i, (P_{\hat{z}})_i \leq \hat{u}_i,\ |(P_{\hat{z}})_i - \hat{z}_i| \leq \epsilon\|w_i\|_*,\ \hat{z}_i, (P_{\hat{z}})_i \in \mathbb{R}\bigg\},\end{aligned}$$

which is now in terms of the scalar optimization variables we desired.

We now turn to solving the above optimization. First, consider the case that $c_i \geq 0$. Then we seek to maximize the difference $(\tilde{x}[1])_i - z_i$ subject to the given constraints. Noting that $(\tilde{x}[1])_i \leq q(l, u)$ and

$z_i \geq 0$ on the above feasible set, we remark that the objective is upper bounded as $c_i((\tilde{x}[1])_i - z_i) \leq c_i q(l,u)$. Indeed, this upper bound is attained at the feasible point defined by $z_i = \hat{z}_i = (P_{\hat{z}})_i = 0$ and $(\tilde{x}[1])_i = q(l,u)$. Hence, we conclude that for all $i \in \{1,2,\dots,n_z\}$ such that $c_i \geq 0$, it holds that

$$\Delta f_i^* \leq c_i q(l,u). \tag{20}$$

Now consider the case that $c_i < 0$. Then we seek to minimize the difference $(\tilde{x}[1])_i - z_i$ subject to the given constraints. In this case, the optimal objective value depends on the relative sizes of $\hat{u}_i$ and $\epsilon \|w_i\|_*$. In particular, when $\hat{u}_i \leq \epsilon\|w_i\|_*$, the constraint $\hat{z}_i \leq \hat{u}_i$ becomes active at optimum, yielding a supremum value of $-c_i u_i$. Alternatively, when $\epsilon\|w_i\|_* \leq \hat{u}_i$, the constraint $|(P_{\hat{z}})_i - \hat{z}_i| \leq \epsilon\|w_i\|_*$ becomes active at optimum, yielding the supremum value of $-c_i\epsilon\|w_i\|_*$. Therefore, we conclude that for all $i \in \{1,2,\dots,n_z\}$ such that $c_i < 0$, it holds that

$$\Delta f_i^* \leq -c_i \min\{\hat{u}_i, \epsilon\|w_i\|_*\}. \tag{21}$$

Substituting (20) and (21) into (19) gives the desired bound. □

5.3.2 Proposed Partition

We now focus on developing an optimal two-part partitioning scheme based on the worst-case relaxation bound of Theorem 2. We take $\epsilon \approx 0$ to simplify the analysis. However, we see that neither the preactivation bounds nor the rows of the weight matrix affect the SDP relaxation bound when $\epsilon = 0$. In particular, the bound takes the form

$$\Delta f_{\text{SDP}}^*(\mathcal{X}) \leq q(l,u) \sum_{i=1}^{n_z} \text{ReLU}(c_i),$$

where

$$q(l,u) = \max_{k \in \{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}.$$

Therefore, since the design of the partition amounts to choosing inputs bounds l, u for the input parts, the input bounds serve as our optimization variables in minimizing the worst-case relaxation bound. In particular, its clear from the form of q that partitioning along the coordinate with the loosest input bound minimizes the worst-case relaxation bound. This observation is formalized in Theorem 3 that follows.

Theorem 3 (Optimal SDP partition). *Consider the two-part partitions defined by uniform division of $\mathcal{X}$ along the coordinate axes: $\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}$, with $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : l_i^{(1)} \leq x \leq u_i^{(1)}\}$ and $\mathcal{X}_i^{(2)} = \{x \in \mathcal{X} : l_i^{(2)} \leq x \leq u_i^{(2)}\}$, where $l_i^{(1)} = l$, $u_i^{(1)} = (u_1, u_2, \dots, u_{i-1}, \frac{1}{2}(l_i + u_i), u_{i+1}, \dots, u_{n_x})$, $l_i^{(2)} = (l_1, l_2, \dots, l_{i-1}, \frac{1}{2}(l_i + u_i), l_{i+1}, \dots, l_{n_x})$, and $u_i^{(2)} = u$, for all $i \in \{1,2,\dots,n_x\} =: \mathcal{I}$. Let*

$$i^* \in \arg \max_{k \in \{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}, \tag{22}$$

*and assume $|l_{i^*}| \neq |u_{i^*}|$. Then the partition $\{\mathcal{X}_{i^*}^{(1)}, \mathcal{X}_{i^*}^{(2)}\}$ is optimal in the sense that the upper bound factor $q(l_{i^*}^{(j)}, u_{i^*}^{(j)})$ in (**??**) equals the unpartitioned upper bound $q(l,u)$ on one part j of the partition, is strictly less than $q(l,u)$ on the other part, and $q(l_i^{(j)}, u_i^{(j)}) = q(u,l)$ for both $j \in \{1,2\}$ for all other $i \notin \arg\max_{k \in \{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}$.*

Proof. First consider partitioning along coordinate $i \notin \arg\max_{k \in \{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}$. Then

$$\begin{aligned} q(l_i^{(1)}, u_i^{(1)}) &= \max_{k \in \{1,2,\dots,n_x\}} \max\{|(l_i^{(1)})_k|, |(u_i^{(1)})_k|\} \\ &= \max\left\{|l_1|, \dots, |l_{n_x}|, |u_1|, \dots, \left|\frac{l_i + u_i}{2}\right|, \dots, |u_{n_x}|\right\} \\ &= \max\{|l_{i^*}|, |u_{i^*}|\} \\ &= q(l,u), \end{aligned}$$

since $|(l_i+u_i)/2| \leq (|l_i|+|u_i|)/2 < \max\{|l_{i^*}|,|u_{i^*}|\}$ and $i \neq i^*$ implies that

$$\max\{|l_{i^*}|,|u_{i^*}|\} \in \left\{|l_1|,\dots,|l_{n_x}|,|u_1|,\dots,\left|\frac{l_i+u_i}{2}\right|,\dots,|u_{n_x}|\right\}.$$

In an analogous fashion, it follows that

$$q(l_i^{(2)},u_i^{(2)}) = q(l,u).$$

Now consider partitioning along coordinate i^*. Note that either $\max_{k\in\{1,2,\dots,n_x\}}\max\{|l_k|,|u_k|\} = |l_{i^*}|$ or $\max_{k\in\{1,2,\dots,n_x\}}\max\{|l_k|,|u_k|\} = |u_{i^*}|$. Suppose the first case holds true. Then

$$\begin{aligned} q(l_{i^*}^{(1)},u_{i^*}^{(1)}) &= \max_{k\in\{1,2,\dots,n_x\}}\max\{|(l_{i^*}^{(1)})_k|,|(u_{i^*}^{(1)})_k|\} \\ &= \max\left\{|l_1|,\dots,|l_{n_x}|,|u_1|,\dots,\left|\frac{l_{i^*}+u_{i^*}}{2}\right|,\dots,|u_{n_x}|\right\} \\ &= |l_{i^*}| \\ &= q(l,u), \end{aligned}$$

since $|(l_{i^*}+u_{i^*})/2| \leq (|l_i^*|+|u_i^*|)/2 < |l_i^*|$ and

$$|l_{i^*}| \in \max\left\{|l_1|,\dots,|l_{n_x}|,|u_1|,\dots,\left|\frac{l_{i^*}+u_{i^*}}{2}\right|,\dots,|u_{n_x}|\right\}.$$

Over the second part of the partition,

$$\begin{aligned} q(l_{i^*}^{(2)},u_{i^*}^{(2)}) &= \max_{k\in\{1,2,\dots,n_x\}}\max\{|(l_{i^*}^{(2)})_k|,|(u_{i^*}^{(2)})_k|\} \\ &= \max\left\{|l_1|,\dots,\left|\frac{l_{i^*}+u_{i^*}}{2}\right|,\dots,|l_{n_x}|,|u_1|,\dots,|u_{n_x}|\right\} \\ &< |l_{i^*}| \\ &= q(l,u), \end{aligned}$$

since $|(l_{i^*}+u_{i^*})/2| < |l_{i^*}|$ and

$$|l_i^*| \notin \left\{|l_1|,\dots,\left|\frac{l_{i^*}+u_{i^*}}{2}\right|,\dots,|l_{n_x}|,|u_1|,\dots,|u_{n_x}|\right\}$$

since $|l_{i^*}| \neq |u_{i^*}|$. In the other case that $\max_{k\in\{1,2,\dots,n_x\}}\max\{|l_k|,|u_k|\} = |u_{i^*}|$, it follows via the same argument that

$$\begin{aligned} q(l_{i^*}^{(1)},u_{i^*}^{(1)}) &< q(l,u), \\ q(l_{i^*}^{(2)},u_{i^*}^{(2)}) &= q(l,u). \end{aligned}$$

Since partitioning along any other coordinate $i \notin \arg\max_{k\in\{1,2,\dots,n_x\}}\max\{|l_k|,|u_k|\}$ was shown to yield $q(l_i^{(1)},u_i^{(1)}) = q(l_i^{(2)},u_i^{(2)}) = q(u,l)$, we conclude that the coordinate i^* is optimal in the sense proposed. □

Intuitively, the partitioning scheme defined in Theorem 3 is optimal because any other uniform partition along a coordinate axis cannot tighten the worst-case relaxation error bound. On the other hand, Theorem 3 guarantees that using the partition coordinate in (22) results in a strict tightening of the worst-case relaxation error on at least one part of the partition.

6 SDP Relaxation with Nonconvex Cuts

6.1 Convex Cuts

As a well-known technique in nonlinear and mixed-integer optimization, adding valid convex constraints (convex cuts) could potentially reduce the relaxation gap of the problem. The most effective cut for SDP relaxation is based on the reformulation-linearization technique (RLT), which is obtained by multiplying linear constraints and relaxing the product into linear matrix constraints [14].

In the formulation of $\mathcal{N}_{\text{SDP}}^{[k]}$, there are only 2 linear constraints: $\tilde{x}[k] \geq 0$ and $\tilde{x}[k] \geq W^{[k-1]}\tilde{x}[k-1]$. The RLT cuts associated with these linear constraints exist in the original constraint set $\mathcal{N}_{\text{SDP}}^{[k]}$, except for the one obtained by the following multiplication:

$$(\tilde{x}[k] - W^{[k-1]}\tilde{x}[k-1])(\tilde{x}[k] - W^{[k-1]}\tilde{x}[k])^\top \geq 0 \tag{23}$$

which leads to the relaxed cut:

$$\begin{aligned}&\operatorname{diag}(\tilde{X}[C_k] - W^{[k-1]}X[B_k] - X[B_k^\top]W^{[k-1]^\top}\\&+ W^{[k-1]}X[A_k]W^{[k-1]^\top}) \geq 0\end{aligned} \tag{24}$$

Note that only the diagonal entries are of interest because the other terms do not appear in the original QCQP formulation of the problem.

6.1.1 Deficiency of RLT

Although the above RLT reformulation seems to add new information to the SDP relaxation, a closer examination reveals otherwise.

Proposition 2. *Constraint set (5) implies the RLT inequality (24) for all* $k \in \{1, \ldots, K\}$.

Proof. Since $\tilde{X} \succeq 0$, all principle submatrices $\tilde{X}[k]$ are positive semidefinite for $k \in \{1, \ldots, K\}$. Then, $\tilde{X}[k] \succeq 0$ can be restated in terms of the general Schur's complement:

$$\begin{aligned}&\tilde{X}[k] \succeq 0 \iff \{\tilde{X}[C_k] \succeq 0,\\&\tilde{X}[A_k] - \tilde{X}[B_k](\tilde{X}[C_k])^\dagger \tilde{X}[B_k^\top] \succeq 0,\\&(I - \tilde{X}[C_k](\tilde{X}[C_k])^\dagger)\tilde{X}[B_k^\top] = 0\}\end{aligned} \tag{25}$$

Now, one can write:

$$\begin{aligned}&\operatorname{diag}(W^{[k-1]}\tilde{X}[A_k]W^{[k-1]^\top} -\\&W^{[k-1]}\tilde{X}[B_k](\tilde{X}[C_k])^\dagger \tilde{X}[B_k^\top]W^{[k-1]^\top}) \geq 0\end{aligned} \tag{26}$$

After noticing that $\operatorname{diag}(\tilde{X}[C_k]) = \operatorname{diag}(W^{[k-1]}\tilde{X}[B_k])$ and $(I - \tilde{X}[C_k](\tilde{X}[C_k])^\dagger)\tilde{X}[B_k^\top] = 0$, we obtain:

$$\operatorname{diag}(W^{[k-1]}\tilde{X}[A_k]W^{[k-1]^\top} - \tilde{X}[B_k^\top]W^{[k-1]^\top}) \geq 0 \tag{27}$$

By adding $\operatorname{diag}(\tilde{X}[C_k] - W^{[k-1]}\tilde{X}[B_k]) = 0$ to both sides of equation, the above inequality yields (24). □

Therefore, adding convex cuts to (3) using RLT will only increase computation time without reducing the relaxation gap.

6.2 Source of Relaxation Gap

The SDP relaxation is obtained by replacing the equality constraint $\tilde{X} - \tilde{x}\tilde{x}^\top = 0$ with the convex inequality $\tilde{X} - \tilde{x}\tilde{x}^\top \succeq 0$. Hence, the non-convex constraint $\tilde{X} - \tilde{x}\tilde{x}^\top \preceq 0$ excluded in this formulation is the source of the relaxation gap.

One necessary condition for $\tilde{X} - \tilde{x}\tilde{x}^\top \preceq 0$ is:

$$\begin{aligned} &\phi_i^\top (\tilde{X} - \tilde{x}\tilde{x}^\top)\phi_i \leq 0 \ \forall i \text{ such that} \\ &\{\phi_1, \dots, \phi_{n_{\tilde{x}}}\} \text{forms a basis in } \mathbb{R}^{n_{\tilde{x}}} \end{aligned} \tag{28}$$

However, since $-\|\phi_i^\top \tilde{x}\|^2$ is concave in $\tilde{x}$, it is impossible to add this constraint under a convex optimization framework.

A non-convex cut is defined to be a valid constraint that is non-convex. In this case, any constraint in the form of (28) is a non-convex cut.

6.3 A Penalization Approach

Since directly incorporating any non-convex cut makes the resulting problem non-convex, one may resort to penalization techniques. Explicitly, the objective of (3) can be modified as:

$$c_i^\top z + \sum_{i=1}^{n_{\tilde{x}}} \phi_i^\top (\tilde{x}\tilde{x}^\top - \tilde{X})\phi_i$$

Nevertheless, the constraint $\tilde{X} - \tilde{x}\tilde{x}^\top \succeq 0$ implies that $c_i^\top z + \sum_{i=1}^{n_{\tilde{x}}} \phi_i^\top (\tilde{x}\tilde{x}^\top - \tilde{X})\phi_i \leq c_i^\top z$, therefore making the new problem not necessarily a relaxation of the original problem, i.e. the case $\hat{f}_i^*(\mathcal{X}) \leq f_i^*(\mathcal{X})$ is possible. Moreover, this penalization approach adds a convex term ($\|\phi_i^\top \tilde{x}\|^2$) to the maximizing objective, which destroys the convexity of the problem.

To avoid this issue, one may use the technique proposed in [17]. That work penalizes a given QCQP with a linear objective such that the problem remains a relaxation of the original problem after penalization. The authors proposed a sequential SDP procedure to approximate $f_i^*(\mathcal{X})$ for any i in (2) using the modified objective $p_i^*(\mathcal{X})$, defined as:

$$p_i^* = \sup_{(x,z)\in\mathcal{N}_{\text{SDP}},\ x\in\mathcal{X}} c_i^\top z + \sqrt{c_i^\top (\tilde{x}\tilde{x}^\top - \tilde{X})c_i} \tag{29}$$

it can be shown that $p_i^*(\mathcal{X})$ remains a relaxation of the original problem (2), i.e. $p_i^*(\mathcal{X}) \geq f_i^*(\mathcal{X})$.

6.3.1 Deficiency of Penalty Methods

The problem with the approach proposed in (29) is that the vector c_i inside the square root must match that of the original linear objective, making it practically limited. Arguing under the framework of adding constraints in the form of (28), the method in [17] only enforces one of them, namely $c_i^\top (\tilde{X} - \tilde{x}\tilde{x}^\top)c_i \leq 0$. Objective (29) ensures that $c_i^\top (\tilde{X} - \tilde{x}\tilde{x}^\top)c_i$ is as small as possible.

Although interior point methods are known to converge to maximum rank solutions, the relaxed matrix is low rank in general. Therefore, it is likely that the constraint $c_i^\top (\tilde{X} - \tilde{x}\tilde{x}^\top)c_i \leq 0$ is already satisfied for the unmodified solution. Simulations found in Section 6 corroborate this fact.

6.4 Secant Approximation

To address the above-mentioned issues, we leverage the method of secant approximation, inspired by [18]. Note that the constraint (28) can be written as:

$$-(\phi_i^\top \tilde{x})^2 \leq -\langle \tilde{X}, \phi_i\phi_i^\top \rangle$$

Graphically, the grey area in Figure 3 indicates the original feasible space of $(\tilde{x}, \tilde{X})$ constrained by $-(\phi_i^\top \tilde{x})^2 \leq -\langle \tilde{X}, \phi_i\phi_i^\top \rangle$. Since the set is non-convex, we use secant lines to approximate it. The approximated feasible space contains every point above the 3 secant lines, namely the grey area plus the red area. ξ_1, ξ_2 are the points by which the feasible space is divided. For each part of the space, we use a separate line to lower-bound it. In other words, for each divided space, a necessary condition for $(\tilde{x}, \tilde{X})$ to lie in the original feasible space is given. As observed in Figure 3, since secants are used for approximation, it is

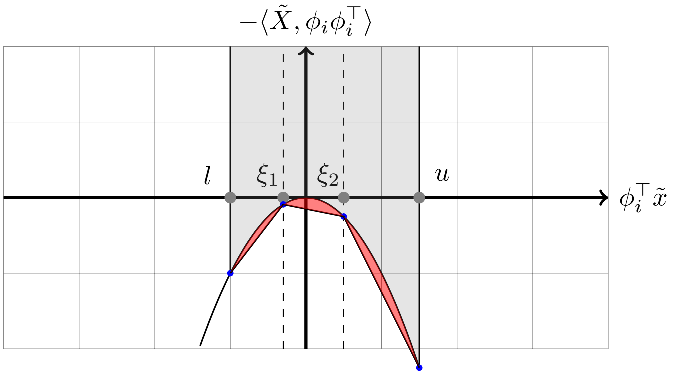


Figure 3: l and u are lower and upper bounds of $\{\phi_i^\top \tilde{x} |\ (\tilde{x}, \tilde{X}) \in \mathcal{N}_{\text{SDP}}\}$, respectively. Red areas indicate false feasible space induced by the secant approximation.

important that l and u be known in advance. For this certification problem, those bounds can be calculated with auxiliary SDPs. More importantly, for any $\{(\tilde{x}, \tilde{X}) \in \mathcal{N}_{\text{SDP}} | \tilde{X} = \tilde{x}\tilde{x}^\top\}$, $(\tilde{x}, \tilde{X})$ lies above exactly one secant line.

In mathematical language, this means that

$$\bigvee_{q\in\{0,\ldots,Q+1\}} (\tilde{x}, \tilde{X}) \in \Gamma_q(\phi_i) \tag{30}$$

where $\Gamma_q(\phi_i)$ is defined as

$$\bigvee_{q\in\{0,\ldots,Q+1\}} \begin{bmatrix} \xi_q \le \phi_i^\top \tilde{x} \le \xi_{q+1}, \text{ and} \\ \xi_q\xi_{q+1} - (\phi_i^\top \tilde{x})(\xi_q + \xi_{q+1}) \le -\langle \tilde{X}, \phi_i\phi_i^\top \rangle \end{bmatrix} \tag{31}$$

and $\bigvee$ is a logical OR symbol commonly used in boolean algebra and $\xi_0 \triangleq l, \xi_{q+1} \triangleq u$, with Q being the number of dividing points. This means that every pair $(\tilde{x}, \tilde{X})$ in $\{(\tilde{x}, \tilde{X}) \in \mathcal{N}_{\text{SDP}} | \tilde{X} = \tilde{x}\tilde{x}^\top\}$ belongs to only *one* $\Gamma_q(\cdot)$ for some q, except when $\phi_i^\top \tilde{x} = \xi_q$ for $q = \{0, \ldots, Q+1\}$.

Lemma 4. *The secant approximations (30) exactly recovers the constraint $-(\phi_i^\top \tilde{x})^2 \le -\langle \tilde{X}, \phi_i\phi_i^\top \rangle$ when $Q \to \infty$.*

Proof. As $Q \to \infty$, $\xi_q = \phi_i^\top \tilde{x} = \xi_{q+1}$ for $q = \{0, \ldots, Q+1\}$. Therefore, $\xi_q\xi_{q+1} - (\phi_i^\top \tilde{x})(\xi_q + \xi_{q+1}) = (\phi_i^\top \tilde{x})^2 - 2(\phi_i^\top \tilde{x})^2 = -(\phi_i^\top \tilde{x})^2$, resulting in $-(\phi_i^\top \tilde{x})^2 \le -\langle \tilde{X}, \phi_i\phi_i^\top \rangle$ □

Of course, an infinite number of divisions is impractical from both a complexity standpoint and a numerical standpoint. Thus, selecting ξ_q's intelligently to minimize the red area in Figure 3 is critical.

Proposition 3. *Given l and u for $\{\phi_i^\top \tilde{x} |\ (\tilde{x}, \tilde{X}) \in \mathcal{N}_{SDP}\}$, a secant approximation with Q division points $\{\xi_q\}_{q=1}^{Q}$ achieves the best approximation when $[l, u]$ is equally partitioned.*

Proof. We optimize for the red area in Figure 3:

$$\min_{\{\xi_q\}_{q=1}^{Q}} \sum_{q=1}^{Q+1} \int_{\xi_{q-1}}^{\xi_q} -x^2 - (\xi_{q-1}\xi_q - x(\xi_{q-1} + \xi_q))dx$$

Since the objective $f(\cdot)$ is convex in $\{\xi_q\}_{q=1}^{Q}$, $\nabla f(\{\xi_q\}_{q=1}^{Q}) = 0$ implies that $\xi_q = \frac{1}{2}(\xi_{q-1} + \xi_{q+1})$ for $q = \{0, \ldots, Q+1\}$. This further implies an equal partitioning of $[l, u]$. □

6.5 Constructing Valid Cuts

Naturally, only ϕ_i's for which the negative definite constraints are violated ($\phi_i^\top(\tilde{X} - \tilde{x}\tilde{x}^\top)\phi_i > 0$) are of interest. Towards this end, we run the unstrengthened SDP problem (3) once to obtain a candidate solution $(\tilde{x}^*, \tilde{X}^*)$, and then find the eigenvectors corresponding to the positive eigenvalues of $\tilde{x}^*(\tilde{x}^*)^\top - \tilde{X}^*$.

However, if the candidate solution $(\tilde{x}^*, \tilde{X}^*)$ lies in the false feasible space, then the secant approximation may not cut off the candidate solution, making the added constraints redundant. This is well demonstrated by Example 1 in [18].

Therefore, it is necessary to actively search for a *valid cut* with secant approximations. Formally, this can be accomplished via disjunctive programming.

6.6 Disjunctive Programming

Theorem 3.1 in [19] serves as the basis to constructing families of valid inequalities from a disjunctive programming problem, named Cut Generating Linear Programming (CGLP) [18]:

$$\begin{aligned} \min_{\alpha,\beta,\{\mu^q\},\{\nu^q\}} \quad & \alpha^\top \chi^* - \beta \\ \text{s.t.} \quad & A^\top \mu^q + D_q^\top \nu^q \le \alpha \quad q \in \{1,\dots,Q\} \\ & b^\top \mu^q + d_q^\top \nu^q \ge \beta \quad q \in \{1,\dots,Q\} \\ & \mu^q, \nu^q \ge 0 \quad q \in \{1,\dots,Q\} \\ & \sum_{q=1}^{Q} (\kappa^\top \mu^q + \eta_q^\top \nu^q) = 1 \end{aligned} \tag{CGLP}$$

where χ is a vectorized version of the variables $(\tilde{x}, \tilde{X})$ with $\chi^* = [\tilde{x}^{*\top}, \text{vec}(\tilde{X}^*)^\top]^\top$, $\{\mu^q\}, \{\nu^q\}$ being vectors of nonnegative entries, and $\kappa, \{\eta_q\}$ being normalizing constants. $\kappa, \{\eta_q\}$ are some constants to help with numerical stabilities, and are not of theoretical interest.

Most importantly, $A\chi \ge b$ ($A\chi$ is just matrix multiplication) is a reformulation of the constraint $(\tilde{x}, \tilde{X}) \in \mathcal{N}_{\text{SDP}}$, and $D_q^\top \chi \ge d_q$ is a reformulation of the constraint $(\tilde{x}, \tilde{X}) \in \Gamma_q(\phi_i)$ for any ϕ_i. This is possible because every constraint is affine after lifting (i.e. introducing $\tilde{X}$ in the place of $\tilde{x}\tilde{x}^\top$). That is, different CGLPs exist for different ϕ_is. Moreover, note that given $Q \in \mathbb{Z}$, $\{\xi_q\}_{q=1}^{Q}$ are calculated according to Proposition 3.

The following result is a modification of Theorem 1 in [18]:

Theorem 4. *If the optimal value of (CGLP) is negative, then a valid cut $\alpha\chi \ge \beta$ is found that cuts off the candidate solution $(\tilde{x}^*, \tilde{X}^*)$. Conversely, if the optimal value is nonnegative, then no such cut exists for $\{\Gamma_q(\phi_i)\}_{q=0}^{Q+1}$.*

Proof. According to Theorem 3.1 of [19], $\alpha\chi \ge \beta$ is a valid constraint for (3) with the secant approximations (30) if and only if the first three lines of the CGLP constraints hold true. All alphas and betas for which $\alpha\chi \ge \beta$ is a valid constraint is in the polyhedral search space of CGLP. If there exist α^* and β^* such that $\alpha^*\chi^* < \beta^*$, the optimal value of CGLP must be negative. On the other hand, if the optimal value of CGLP is negative, such α^* and β^* must exist. In this case, α^* and β^* must also meet the condition $\alpha^*\chi \ge \beta^*$ due to the convexity of polyhedra. Then, χ^* contradicts the requirements of χ, making the constraint $\alpha\chi \ge \beta$ a valid cut.

Conversely, if no such α^* and β^* exist, then χ^* already satisfies all valid constraints with respect to $\{\Gamma_q(\phi_i)\}_{q=0}^{Q+1}$ since the polyhedral search space is convex, and convexity guarantees an exhaustive search. Thus, no valid cut can be found. □

Theorem 4 and Lemma 4 lead to the following result:

Corollary 1. *If $D_k^\top \chi \ge d_k$ corresponds to a set of constraints $\bigvee_{q\in\{0,\dots,Q+1\}} (\tilde{x}, \tilde{X}) \in \Gamma_q(\phi_i)$ with ϕ_i being an eigenvector of $\tilde{x}^*\tilde{x}^{*\top} - \tilde{X}^*$ associated with a positive eigenvalue, then a valid cut $\alpha\chi \ge \beta$ always exists if $Q \to \infty$.*

6.7 A Sequential Algorithm

We propose Algorithm 1 to sequentially apply non-convex cuts to the verification problem.

Algorithm 1: Sequential SDP verification of NN Robustness by adding secant-approximated non-convex cuts

```
verification ($\mathcal{X}, \mathcal{S}, \{W^0 \dots W^{K-1}\}, Q, \max_{\text{iter}}, \gamma$);
Input  : Input uncertainty set $\mathcal{X}$, safe set $\mathcal{S}$, trained network weights $\{W^0 \dots W^{K-1}\}$, number of
         partitions for the secant approximation, maximum number of iterations $\max_{iter}$, and the
         eigenvalue threshold $\gamma$.
Output: $\tau_r = \max C_r^\top z$ for $r \in \{1, \dots, R\}$, where $C_r$ is the $r^{\text{th}}$ row of $C$, and R is the number of
         rows of $C$, as specified by the safe set $\mathcal{S}$
for $c = C_1, \dots, C_R$ do
    $(\tilde{x}^*, \tilde{X}^*) \leftarrow \arg\max\{c^\top z : (x, z) \in \mathcal{N}_{\text{SDP}},\ x \in \mathcal{X}\}$ ;
    $\tilde{\alpha} = [], \tilde{\beta} = []$ ;
    for $i = 1, \dots, max_{iter}$ do
        $(\lambda \in \mathbb{R}^m, V \in \mathbb{R}^{n_{\tilde{x}} \times m}) \leftarrow$ eigenvalues of $\tilde{x}^* \tilde{x}^{*^\top} - \tilde{X}^*$ bigger than $\gamma$, and with eigenvector
          corresponding to the $i^{\text{th}}$ eigenvalue being $i^{\text{th}}$ column of $V$ ;
        for $\phi = V_1, \dots, V_m$ do
            $(l, u) \leftarrow \{\min, \max\}\{\phi^\top z : (x, z) \in \mathcal{N}_{\text{SDP}},\ x \in \mathcal{X}\}$ ;
            $(\alpha, \beta) \leftarrow \text{CGLP}(Q, \phi, u, l)$;
            if $\alpha^\top \chi^* < \beta$ then
                $\tilde{\alpha} = [\tilde{\alpha}; \alpha^\top], \tilde{\beta} = [\tilde{\beta}; \beta]$
            end
        end
        $(\tilde{x}^*, \tilde{X}^*) \leftarrow \arg\max\{c^\top z : (x, z) \in \mathcal{N}_{\text{SDP}}, \tilde{\alpha}\chi \geq \tilde{\beta},\ x \in \mathcal{X}\}$ ;
    end
    $\tau_r = c^\top \tilde{x}^*[K]$;
end
```

The main result of this report is stated below.

Theorem 5. *For every iteration $i \in \{2, \dots, \max_{iter}\}$ in Algorithm 1, it holds that*

$$f^*(\mathcal{X}) \leq c^\top \tilde{x}_i^*[K] \leq c^\top \tilde{x}_{i-1}^*[K] \leq \hat{f}^*(\mathcal{X}) \tag{32}$$

with $\tilde{x}_i^*, \tilde{X}_i^*$, and by extention P_i^* being the optimizers of the i^{th} iteration. The middle inequality becomes strict if at least one of the optimal values of CGLPs in iteration i is negative and either of the following holds:

- *$\hat{c}$ lies in the null space of P_{i-1}, where $\hat{c} \triangleq [0_{1\times(n_{\tilde{x}}-n_K)}\ c^\top]^\top$*
- *$Q \to \infty$*

Proof. The inequalities $c^\top \tilde{x}_i^*[K] \leq c^\top \tilde{x}_{i-1}^*[K] \leq \hat{f}^*(\mathcal{X})$ are due to the fact that a strict reduction in the search space will not yield a higher optimal value. $f^*(\mathcal{X})$ serves as a lower bound since the constraints are valid, according to Theorem 4.

Now, since the objective is linear, $\hat{c}^\top \tilde{x} = \gamma$ must be a supporting hyperplane for the spectrahedral (convex body arising from linear matrix inequalities) feasible set at $\tilde{x}_{i-1}^*$, where γ denotes as the constant $c^\top \tilde{x}_{i-1}^*[K]$. By Proposition 2 in [20], the intersection of a closed convex set (e.g., this feasible set) and the supporting hyperplane is an exposed face. Furthermore, by [21], it is known that any exposed face of a spectraheron is a proper face, and the null space of P will stay constant over the relative interior of this face. Thus, the intersection will consist of points $(\tilde{x}, \tilde{X})$ such that $\hat{c}^\top \tilde{x} = \gamma$ and $\mathcal{N}(P) = \mathcal{N}(P_{i-1})$.

Let a basis of $\mathcal{N}(P_{i-1})$ be denoted as $\{\psi_j\}$, where each vector is partitioned as $\psi_j \triangleq [\omega_j \ \ \hat{\psi}_j^\top]^\top$. Therefore, $\hat{\psi}_j^\top \tilde{x} = -\omega_j$ for $j \in \{1, \ldots, n_{\tilde{x}} + 1\}$. If $\hat{c} \in \mathcal{N}(P_{i-1})$, the supporting hyperplane will intersect the spectrahedron at exactly one point (the face is of affine dimension 0) because $\tilde{x}$ is already fixed in $\{\hat{\psi}_j\}$ coordinates, and γ can only take on one value. If the intersection only consists of one point, then a valid cut will invalidate this point, and the objective value will strictly decrease.

On the other hand, if $Q \to \infty$, the original negative semidefinite constraint is recovered, as per Lemma 4. Then, after using the valid cut, P_i will have a strictly lower rank than P_{i-1}. Since the intersection consists of points such that $\mathcal{N}(P) = \mathcal{N}(P_{i-1})$, all those points will not be in the feasible space anymore under the presence of this valid cut. This leads to a strict decrease in objective value. □

Since $n_{\tilde{x}}$ is usually very large for multilayer networks, the first condition is often satisfied. However, it is important to note that those 2 conditions are only sufficient conditions, and that in practice there is normally a non-zero reduction in relaxation gap whenever a valid cut is calculated. Moreover, if a valid cut is given, the algorithm will always reach new optimal points, thus avoiding the situation of repeating the same search again and again.

This algorithm is asymptotically exact if an infinite number of iterations is taken, as explained below.

Lemma 5. *The relation $c^\top \tilde{x}_i^*[K] = f^*(\mathcal{X})$ holds as $i \to \infty$, provided that $\gamma = 0$ and $Q \to \infty$*

Proof. The proof follows directly from Theorems 4, 5, and Corollary 1. □

6.8 Geometric Analysis of Non-convex Cuts

In this section, we offer a geometric intuition into the success of non-convex cuts. We revisit (30) and write it as:

$$\begin{gathered} \xi_q \xi_{q+1} - (\phi^\top \tilde{x})(\xi_q + \xi_{q+1}) \leq -\langle \tilde{X}, \phi\phi^\top \rangle \\ \Leftrightarrow \langle \tilde{X}, \phi\phi^\top \rangle \leq (\phi^\top \tilde{x} - \xi_q)(\xi_{q+1} - \phi^\top \tilde{x}) + (\phi^\top \tilde{x})^2 \end{gathered} \tag{33}$$

Since there always exists a partition number Q large enough such that $(\phi^\top \tilde{x} - \xi_q)(\xi_{q+1} - \phi^\top \tilde{x}) \leq \epsilon$, it is possible to rewrite equation (33):

$$\begin{gathered} \sum_{i,j \in \{n_{\tilde{x}}\} \times \{n_{\tilde{x}}\}, i \neq j} (\phi^i \phi^j) \|\vec{x}_i\| \|\vec{x}_j\| cos(\theta_{ij}) \leq \\ \sum_{i,j \in \{n_{\tilde{x}}\} \times \{n_{\tilde{x}}\}, i \neq j} (\phi^i \phi^j) \|\vec{x}_i\| \|\vec{x}_j\| cos(\theta_i) cos(\theta_j) + \epsilon \end{gathered} \tag{34}$$

after substituting equation (8). Here, ϕ^i is the i^{th} entry of ϕ (to be distinguished from ϕ_i, which is a vector in the set of basis), and θ_{ij} is the angle between the vectors $\vec{x}_i, \vec{x}_j$, and θ_i denotes the angle between $\vec{e}$ and $\vec{x}_i$ for $i \in \{1, \ldots, n_{\tilde{x}}\}$.

Therefore, there must exist a pair (i, j) such that

$$cos(\theta_{ij}) \leq cos(\theta_i) cos(\theta_j) + \frac{\epsilon}{n_{\tilde{x}}} \tag{35}$$

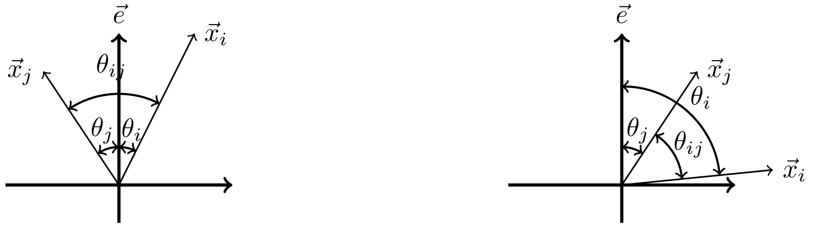
Since $n_{\tilde{x}}$ is usually large in multi-layer networks and ϵ is chosen to be very small, $\frac{\epsilon}{n_{\tilde{x}}}$ can be neglected for practical purposes.

Recall from Section 4 that $\theta_i = 0 \ \forall i \in \{1, \ldots, n_{\tilde{x}}\}$ is both sufficient and necessary for the tightness of the relaxation. Furthermore, in the following lemma we will show that any decrease in θ_i for any i can contribute to a tighter relaxation.

Lemma 6. *For any row vector $\vec{x}$ in V, as part of the factorization of $\tilde{X}$, any increase in the lower bound for $\frac{\vec{x} \cdot \vec{e}}{\|\vec{x}\|}$ will decrease the upper bound on the relaxation gap for at least one entry in $\tilde{x}$.*

Proof. Extending the results from Section 4, it can be verified that for a fixed θ, the angle between $\vec{e}$ and $\vec{\chi}$, a decrease in $\|\vec{\chi}\|$ will lead to a smaller spherical cap. Now, consider a fixed $\|\vec{\chi}\|$. It can be shown that the height of the spherical cap is $\frac{\|\vec{\chi}\|}{2}(1 - cos(\theta))$. Therefore, any increase in $cos(\theta)$ will lead to a smaller cap. Since $\vec{\chi}$ is just a linear combination of different $\vec{x}$, increasing the lower bound for $\frac{\vec{x} \cdot \vec{e}}{\|\vec{x}\|}$ will also increase the lower bound for $\frac{\vec{\chi} \cdot \vec{e}}{\|\vec{\chi}\|}$. □

Given any such pair (i, j), if $cos(\theta_{ij})$ is large, namely possibly close to one, the term $cos(\theta_i)cos(\theta_j)$ will have to be larger under the constraint $cos(\theta_{ij}) \leq cos(\theta_i)cos(\theta_j) + \frac{\epsilon}{n_{\tilde{x}}}$. This means that $cos(\theta_i) \approx 1$ and $cos(\theta_j) \approx 1$, making both angles very small. Without this constraint, it is possible for $\vec{x}_j, \vec{x}_i$ to be collinear, but not collinear with $\vec{e}$, thus making them have large angles θ_i and θ_j. Figure 4 graphically showcases this difference.



(a) $cos(\theta_{ij}) \leq cos(\theta_i)cos(\theta_j)$

(b) $cos(\theta_{ij}) \geq cos(\theta_i)cos(\theta_j)$

Figure 4: Illustration of possible configurations of $\vec{x}_i$, $\vec{x}_j$ given a fixed θ_{ij}

In particular, since ReLU only outputs positive values, we have $\theta_i \in [0, \frac{\pi}{2}]$ for all $i \in \{1, \ldots, n_{\tilde{x}}\}$. However, the vectors $\vec{x}_i$ that correspond to the input layer are exceptions. Without loss of generality, it is possible to also assume these to be nonnegative by changing the weight matrix of the first layer. Therefore, $cos(\theta_i) \geq 0 \; \forall i \in \{1, \ldots, n_{\tilde{x}}\}$. Thus, when $cos(\theta_i)cos(\theta_j) \approx 1$, we have $\theta_i \approx 0$ and $\theta_j \approx 0$. Without this constraint, it is possible that $\theta_j \approx \pi$ and $\theta_i \approx \pi$. In other words, with the ReLU constraints in place, the non-convex cuts will push the vectors towards $\vec{e}$ instead of $-\vec{e}$.

If $cos(\theta_{ij})$ is small, then inequality (35) will not be binding, and the original semidefinite constraints will work satisfactorily. Ideally, inequality (35) will hold for all pairs (i, j), but this is not guaranteed. However, under the assumption that the weight matrices in the neural network are relatively well conditioned (no large condition number) and that ϵ is negligible, a large number of the pairs (i, j) are expected to satisfy or approximately satisfy inequality (35). The proof can be found in the supplementary material.

Proof. As explained, only those pairs (i, j) with a relatively large $cos(\theta_{ij})$ are of interest. Denote the threshold as $\rho \in [0.5, 1)$ such that $cos(\theta_{ij}) \geq \rho$. We call a pair (i, j) to be valid if it satisfies inequality (35) and invalid otherwise. In the worst case, $cos(\theta_i)cos(\theta_j) - cos(\theta_{ij}) = 1 - \rho$ for all valid pairs and $cos(\theta_i)cos(\theta_j) - cos(\theta_{ij}) = -\varrho < 0$ for all invalid pairs. This leads to the case where the number of valid pairs is the smallest (denoted as ϑ_1) and the number of invalid pairs is the largest (denoted as ϑ_2). The other threshold ϱ exists because a number $cos(\theta_{ij})$ too close to $cos(\theta_i)cos(\theta_j)$ approximately satisfies the inequality. Under the assumption that weight matrices are well conditioned, it implies that $\|\vec{x}_i\|\|\vec{x}_j\|$s are approximately the same amongst all (i, j) pairs. Thus, if ϵ is negligible:

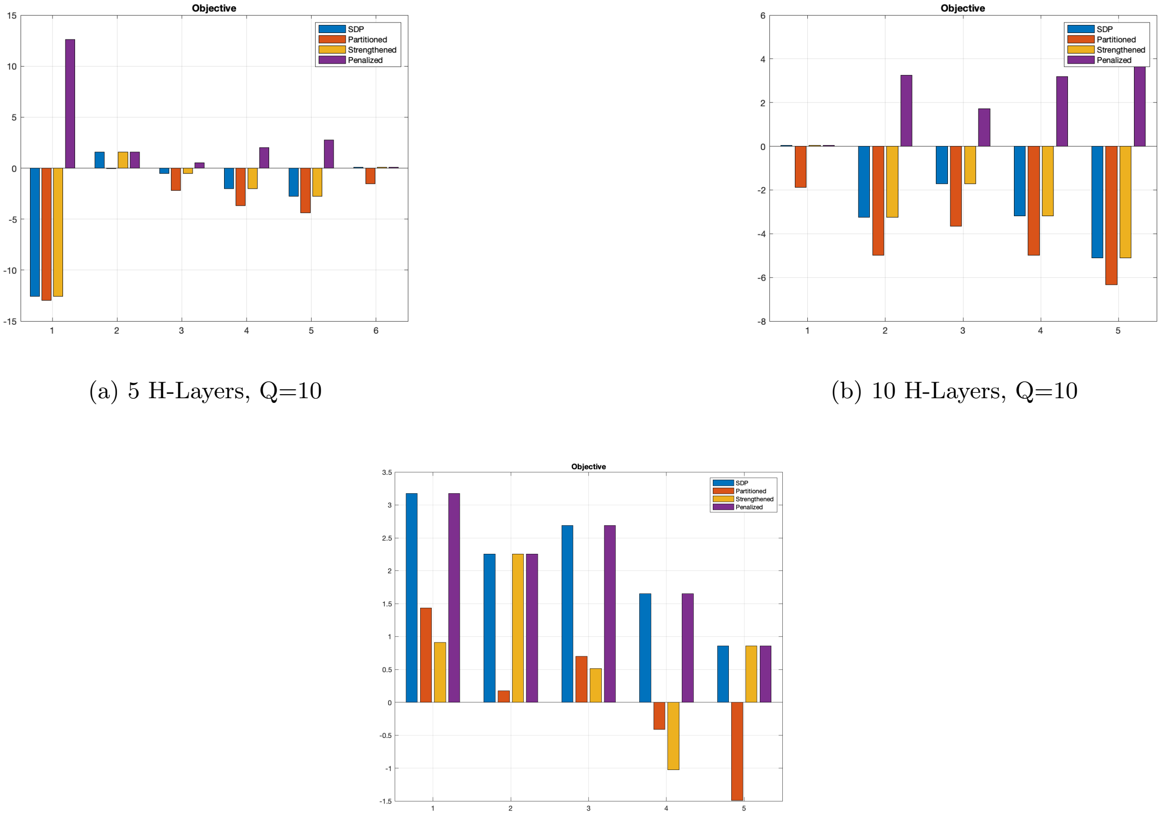
$$\begin{gathered}\sum_{i,j}(\phi^i\phi^j)\|\vec{x}_i\|\|\vec{x}_j\|(cos(\theta_i)cos(\theta_j) - cos(\theta_{ij})) \geq 0 \\ \implies \vartheta_1(1-\rho) - \vartheta_2(\varrho) \approx 0 \implies \frac{\vartheta_1}{\vartheta_2} \approx \frac{\varrho}{1-\rho}\end{gathered} \tag{36}$$

If $\rho = 0.8$ and $\varrho = 0.2$, then at least half of all pairs such that $cos(\theta_{ij}) \geq \rho$ will be valid. As ρ increases, so does $\frac{\vartheta_1}{\vartheta_2}$. This means that almost all of the most important pairs (i, j) are valid. □

Furthermore, the nature of a multi-layer setup means that if $\vec{x}_i$ and $\vec{x}_j$ are from different layers of the network, they depend on each other. Figure 4 shows that the non-convex cut tends to make $\vec{x}_i$ and $\vec{x}_j$ lie on different sides of $\vec{e}$ in the event of a large $cos(\theta_{ij})$. This means that as vectors build on each other (see Figure 1 in [2]), they will revolve around $\vec{e}$, instead of branching out into a certain direction. This further implies a small angle with $\vec{e}$ for all $\vec{x}$.

7 Experiments

We certify the robustness of the IRIS dataset[1] with pre-trained MLP classification network with 99% accuracy on test data using the original SDP approach, the partitioning approach as outlined in section 5, Algorithm 1, and the penalization approach in Section 6.3. For all experiments, $\max_{\text{iter}}$ is set to 10. Moreover, the l_∞ radius of $\mathcal{X}$ is set to 0.15 for networks with 5 and 10 hidden layers, and set to 0.075 for the other 2 networks. This is because larger networks are naturally more sensitive to perturbation.



(a) 5 H-Layers, Q=10

(b) 10 H-Layers, Q=10

(c) 15 H-Layers, Q=10

Figure 5: $\hat{f}_i^*(\mathcal{X})$ for different certification methods and different networks

The objective $\hat{f}_i^*(\mathcal{X})$ of (3) for the aforementioned approaches is shown in Figure 5 for 5 data points each. It can be observed that when the network is small, the original SDP procedure is already sufficient. However, as the number of hidden layers grows, Algorithm 1 shows its strength by certifying more data points. The penalization approach remains ineffective although sometimes it decreases the trace gap.

Moreover, it is apparent that when the network is small, the partitioning technique works better than other techniques.

7.1 How to accelerate the computation

In order to accelerate the optimization process, one would often consider leveraging the low-rank nature of the problem and use Monteiro-Burer method [22] to solve the SDP. This would reduce the complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(nr)$, where r is the rank of the true solution. Since the true rank is 1 in this case, this would dramatically speed up the algorithm.

[1] https://archive.ics.uci.edu/ml/datasets/iris

However, the increase in speed comes at the expense of global guarantee. When using the Monteiro-Burer method, it essentially converts the convex SDP problem into a non-convex one, therefore making the extraction of the true global optimum of the original SDP very hard. In some applications this would not be the case, but again this is impossible to certify from a theoretical viewpoint generally. Given that the very purpose of this optimization problem is about conservative certification, direct utilization of this method is indeed inappropriate.

In [23], the author proposed a new low-rank optimization algorithm prepared for robotic perception problems (It's straight forward to show that (3) is in the form of Problem 1 in [23] by decomposing diag($\cdot$) operator into separate sensing matrices). It iteratively solves low-rank instances of the original SDP problem via the Burer-Monteiro method [22], and the algorithm can detect wether the local minimum attained in the low-rank problem is indeed global minimum. If not, it will return to a higher-rank space and re-search.

By doing so, we can both retain the global guarantee and speed up the algorithm. Since the implementation of this algorithm is no yet released by the author, this is left as a future work.

8 Conclusions

This report studies the problem of neural network verification for which the existing SDP algorithm can fail to provide meaningful certificates due to a large relaxation gap.

2 strategies are devised to counter this effect.

The first strategy is to partition the input space to decrease the relaxation gap. The partitions are derived by minimizing the worst-case error induced by the corresponding convex relaxations, which is theoretically justified by showing that minimizing the true relaxation error is NP-hard. The proposed techniques are experimentally substantiated by demonstrating significant reduction in relaxation error on real and synthetic networks, with only a 2× increase computation time. Our experiments show that the partitioning schemes exhibit tradeoffs between different regimes, namely, as the input size and the number of layers is varied.

The second strategy is to approximate the effect of adding in non-convex constraints by using carefully calculated linear surrogates. By actively constructing constraints using CGLP, we obtain provably valid cuts. More importantly, a negative optimal value in CGLP also guarantees a nonzero reduction in the relaxation gap. It is shown in the experiments that a relatively small number of partitions can help the algorithm successfully reduce relaxation gap, while keeping the computation tractable. It is verified that the relaxation gap for the existing methods is large when tested on large-scale networks, while the proposed algorithm offers a satisfactory performance.

Both algorithms can give exact guarantees in the asymptotic regime. However, the focus here is how to avoid going to the asymptotic regime and solve the problem in as few iterations as possible. Rigorous theoretical and empirical findings corroborate the fact that it is indeed possible to solve many instances of this problem in a few number of iterations by the utilization of theories presented in this report.

References

[1] E. Wong and J. Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," *arXiv preprint arXiv:1711.00851*, 2017.

[2] A. Raghunathan, J. Steinhardt, and P. S. Liang, "Semidefinite relaxations for certifying robustness to adversarial examples," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 877–10 887.

[3] W. Xiang and T. T. Johnson, "Reachability analysis and safety verification for neural network control systems," *arXiv preprint arXiv:1805.09944*, 2018.

[4] T.-W. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, D. Boning, I. S. Dhillon, and L. Daniel, "Towards fast computation of certified robustness for relu networks," *arXiv preprint arXiv:1804.09699*, 2018.

[5] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *Advances in neural information processing systems*, 2018, pp. 4939–4948.

[6] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *arXiv preprint arXiv:1903.01287*, 2019.

[7] V. R. Royo, R. Calandra, D. M. Stipanovic, and C. Tomlin, "Fast neural network verification via shadow prices," *arXiv preprint arXiv:1902.07247*, 2019.

[8] M. Jin, H. Chang, W. Zhu, and S. Sojoudi, "Power up! robust graph convolutional network against evasion attacks based on graph powering," *arXiv preprint arXiv:1905.10029*, 2019.

[9] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," in *International Conference on Computer Aided Verification.* Springer, 2017, pp. 97–117.

[10] G. Singh, R. Ganvir, M. Püschel, and M. Vechev, "Beyond the single neuron convex barrier for neural network certification," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 098–15 109.

[11] D. Bertsimas and I. Dunning, "Multistage robust mixed-integer optimization with adaptive partitions," *Operations Research*, vol. 64, no. 4, pp. 980–998, 2016.

[12] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Advances in neural information processing systems*, 2014, pp. 2924–2932.

[13] B. G. Anderson, Z. Ma, J. Li, and S. Sojoudi, "Tightened convex relaxations for neural network robustness certification," *arXiv preprint arXiv:2004.00570*, 2020.

[14] H. D. Sherali and W. P. Adams, *A reformulation-linearization technique for solving discrete and continuous nonconvex problems.* Springer Science & Business Media, 2013, vol. 31.

[15] Z. Ma and S. Sojoudi, "Strengthened sdp verification of neural network robustness via non-convex cuts," *arXiv preprint arXiv:2010.08603*, 2020.

[16] R. Y. Zhang, "On the tightness of semidefinite relaxations for certifying robustness to adversarial examples," *arXiv preprint arXiv:2006.06759*, 2020.

[17] H. Luo, X. Bai, and J. Peng, "Enhancing semidefinite relaxation for quadratically constrained quadratic programming via penalty methods," *Journal of Optimization Theory and Applications*, vol. 180, no. 3, pp. 964–992, 2019.

[18] A. Saxena, P. Bonami, and J. Lee, "Convex relaxations of non-convex mixed integer quadratically constrained programs: extended formulations," *Mathematical programming*, vol. 124, no. 1-2, pp. 383–411, 2010.

[19] E. Balas, “Disjunctive programming: Properties of the convex hull of feasible points,” *Discrete Applied Mathematics*, vol. 89, no. 1-3, pp. 3–44, 1998.

[20] V. Roshchina, “Face of convex sets,” 2017. [Online]. Available: https://www.roshchina.com/wp-content/uploads/2017/03/faces.pdf

[21] M. Ramana and A. J. Goldman, “Some geometric results in semidefinite programming,” *Journal of Global Optimization*, vol. 7, no. 1, pp. 33–50, 1995.

[22] S. Burer and R. D. Monteiro, “A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization,” *Mathematical Programming*, vol. 95, no. 2, pp. 329–357, 2003.

[23] D. Rosen, “Scalable low-rank semidefinite programming for certifiably correct machine perception,” 2020.