Tight Certification of Adversarially Trained Neural Networks via Nonconvex Low-Rank Semidefinite Relaxations

Hong-Ming Chiu [1] **Richard Y. Zhang** [1]

Abstract

Adversarial training is well-known to produce high-quality neural network models that are empirically robust against adversarial perturbations. Nevertheless, once a model has been adversarially trained, one often desires a certification that the model is truly robust against all future attacks. Unfortunately, when faced with adversarially trained models, all existing approaches have significant trouble making certifications that are strong enough to be practically useful. Linear programming (LP) techniques in particular face a "convex relaxation barrier" that prevent them from making high-quality certifications, even after refinement with mixed-integer linear programming (MILP) and branch-and-bound (BnB) techniques. In this paper, we propose a nonconvex certification technique, based on a low-rank restriction of a semidefinite programming (SDP) relaxation. The nonconvex relaxation makes strong certifications comparable to much more expensive SDP methods, while optimizing over dramatically fewer variables comparable to much weaker LP methods. Despite nonconvexity, we show how off-the-shelf local optimization algorithms can be used to achieve and to certify global optimality in polynomial time. Our experiments find that the nonconvex relaxation almost completely closes the gap towards exact certification of adversarially trained models.

1. Introduction

To make neural network models robust to adversarial perturbation attacks, one popular strategy, known as *adversarial training* (Kurakin et al., 2016; Goodfellow et al., 2015; Madry et al., 2018; Shafahi et al., 2019; Wong et al., 2019), is to attack a pre-trained model, and then to re-train the model with the training set augmented or replaced by the attack. Despite its simplicity, adversarial training works remarkably well in practice. For example, the robust models adversarially trained by Madry et al. (2018) back in 2017 remain essentially unbroken in 2022, after more than four years of white-box penetration testing by researchers world-wide. Later, Shafahi et al. (2019); Wong et al. (2019) have extended the idea to train robust ImageNet classifiers, that achieve a similar level of accuracy, and within a comparable amount of training time, to nonrobust classifiers.

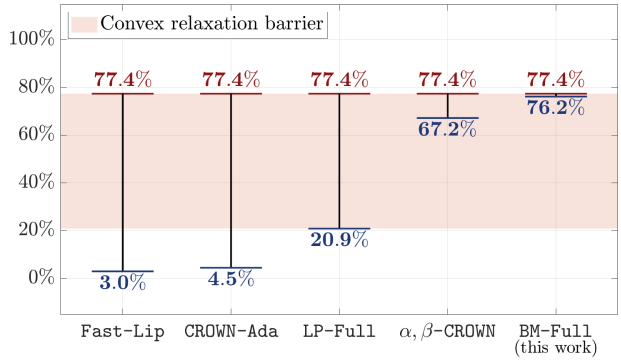

Figure 1: **Certified adversarial training vs. ℓ_2 attacks**. While PGD attacks indicate an empirical upper-bound of 77.4% (red), state-of-the-art LP-based verifiers face a "convex relaxation barrier" (pink shading) that prevent them from certifying lower-bounds better than 20.9% (blue). Even an award-winning state-of-the-art branch-and-bound verifier like α, β-CROWN cannot significantly improve past 67.2% in reasonable time. Our nonconvex relaxation overcomes the convex relaxation barrier, certifying a lower-bound of 76.2% that almost fully closes the empirical-certified gap. (See Section 5 for details.)

Nevertheless, adversarial training is an empirical strategy that does not promise a truly robust model. Given a model that has been made empirically robust through adversarial training, one often desires a formal mathematical proof or *certification* that the model is truly robust against all future attacks. Unfortunately, when faced with an adversarially trained model, all existing approaches have significant trouble making certifications that are strong enough to be practically useful. A model that achieves 77.4% test accuracy on adversarial inputs might only have a certified robust

[1]Department of Electrical and Computer Engineering, University of Illinois at Urbana–Champaign. Correspondence to: Hong-Ming Chiu <hmchiu2@illinois.edu>, Richard Y. Zhang <ryz@illinois.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

accuracy of 20.9%, using state-of-the-art methods (Weng et al., 2018a; Zhang et al., 2018b; Salman et al., 2019) based on a linear programming (LP) relaxation (Wong & Kolter, 2018) of the ReLU activation (see Figure 1).

In fact, recent work by Salman et al. (2019) suggest that it is fundamentally impossible for *any* method based on the LP relaxation to make substantially better certifications for adversarially trained models. Even mixed-integer linear programming (MILP) techniques like branch-and-bound and cutting planes (Tjeng et al., 2019; Xu et al., 2021; Wang et al., 2021), which in theory are capable of exact certification given unlimited time, cannot in practice significantly close the gap left by the LP relaxation within reasonable time. Applying α, β-CROWN (Zhang et al., 2018a; Wang et al., 2021; Xu et al., 2021; Zhang et al., 2022), the winning entry in the International Verification of Neural Networks Competition (VNN-COMP) competitions of 2021 and 2022, only improves the certified robust accuracy to 67.2% before timing out, still leaving an unsatisfactory optimality gap of 10.2%. There appears to be an insurmountable "convex relaxation barrier", between the high degree of robustness that is empirically observed for adversarially trained models, and the low degree of robustness that can be rigorously certified via the LP relaxation, and mixed-integer programming methods based on the LP relaxation.

One promising direction for overcoming the barrier faced by the LP relaxation is to develop methods based on the semidefinite programming (SDP) relaxation of Raghunathan et al. (2018b). Indeed, early experiments (Raghunathan et al., 2018a;b; Zhang, 2020; Dathathri et al., 2020; Batten et al., 2021) all suggest that the SDP relaxation can make significantly stronger certifications than the LP relaxation. However, the cost of solving the SDP relaxation—while technically still polynomial time—is so high as to be completely inaccessible. At its core, the SDP relaxation requires optimizing over an $n \times n$ matrix variable, where n is equal to the total number of ReLU activations, plus the dimension of the input layer. The fundamental difficulty is the need to store and to optimize over n^2 variables: even a single-layer MNIST classifier with 200 ReLU activations requires storing and optimizing over the $\approx 10^6$ elements of a 986×986 matrix, which is already nearing the limit of state-of-the-art SDP solvers like MOSEK (2019), whose worst-case runtime scales as $O(n^6)$. Despite widespread speculation, it is currently unknown whether SDP-based methods will truly be able to provide tight certification for adversarially trained models.

Contributions This paper proposes a nonconvex certification technique, based on a *low-rank restriction* of the usual convex SDP relaxation of the ReLU activation. Rigorously, we show that the nonconvex relaxation is always at least as tight as the SDP relaxation, but that it optimizes over the nr elements of an $n \times r$ rectangular matrix. If a small value of r can be used (we later validate this experimentally), then the nonconvex relaxation can make strong certifications comparable to the SDP relaxation, while optimizing over a dramatically smaller number of variables comparable to the LP relaxation.

The nonconvexity of the relaxation poses a serious issue. The correctness of our certification hinges critically on our ability to solve a nonconvex verification problem to *global* optimality, and then to *certify* this global optimality, but large-scale optimization algorithms are only capable of achieving and certifying *local* optimality. In this paper, we establish that if a local minimum for the verification problem is also global, then under a mild constraint qualification, the Lagrange multipliers that certify its local optimality are also guaranteed to certify its global optimality via Lagrangian duality (see Proposition 3.1 and Theorem 4.3). Conversely, if the local minimum is non-global, then under the same mild constraint qualification, the Lagrange multipliers generate a direction of global improvement (see Theorem 4.4), thereby ensuring that local optimization will eventually achieve certified global optimality.

Our experiments provide empirical confirmation of our theoretical claims. We re-examine the models originally used by Salman et al. (2019) to demonstrate the existence of a "convex relaxation barrier" for the LP relaxation. Using a relaxation rank r of no more than 10, we re-certify these models using our nonconvex relaxation, in time comparable to that of the best-possible LP relaxation. Our results find that even a basic nonconvex relaxation offers a significant reduction in conservatism. Augmenting the nonconvex relaxation by bound propagation (as is commonly done for the LP relaxation) allows us to almost fully close the gap towards exact certification (see Figure 1).

Related work Robustness certification methods can be broadly divided into exact and conservative methods. Exact methods based on mixed-integer linear programming (MILP) (Tjeng et al., 2019; Xu et al., 2021) and Satisfiability Modulo Theories (SMT) (Katz et al., 2017) can make necessary and sufficient certifications of robustness, but have worst-case runtimes that scale exponentially with the number of activations. Conservative methods can decline to certify a robust model, but have polynomial worst-case runtime, and therefore tend to be much more scalable in practice. Today, most state-of-the-art certification methods are conservative methods based on a triangle-shaped LP relaxation of the ReLU activation function introduced by Wong & Kolter (2018). In particular, (Weng et al., 2018b;a; Zhang et al., 2018a) proposed techniques for strengthening these LP-based relaxations, by propagating tighter layer-wise upper- and lower-bounds on the ReLU activation function. Later work by Wang et al. (2021) and Xu

et al. (2021) progressively refine these bounds using MILP and BnB techniques. Salman et al. (2019) proposed an optimal LP relaxation that unifies all the existing bound-propagating LP-based relaxation methods. This last paper pointed out that even the optimal LP relaxation has a gap cannot be improved; they refer to this inherent looseness as the "convex relaxation barrier". Another line of conservative methods are based on SDP relaxationof the ReLU gate (Raghunathan et al., 2018b; Dathathri et al., 2020). Later work by Batten et al. (2021) further tighten the SDP relaxation using linear cut constraints.

Our proposed approach can be interpreted as an application of the Burer–Monteiro approach (Burer et al., 2002; Burer & Monteiro, 2005) and the Riemannian staircase (Boumal et al., 2016; 2020) for solving the SDP relaxation of Raghunathan et al. (2018b). Here, we emphasize that the rigorous applicability of these prior techniques hinges critically on the linear independence constraint qualification (LICQ), a highly restrictive condition that is difficult to verify in practice. If LICQ does not hold, then the Riemannian staircase can become get stuck at a non-LICQ point, so rigorous global guarantees are lost. In the existing literature, LICQ is often taken as a strong blanket assumption (Rosen et al., 2014; Carlone et al., 2015; Cohen et al., 2019; Rosen et al., 2019), but this reduces the Riemannian staircase from a provable algorithm to an empirical heuristic. In this paper, we formally establish LICQ for the verification problem in Lemma 4.2. Assuming that local optimization does not get stuck at the "corner" of the ReLU (this is the same assumption that allows ReLU models to be trained via gradient descent), it immediately follows that our nonconvex relaxation can be globally optimized in polynomial time.

2. Background

Notations We use $(x_1, \ldots, x_\ell)$ to denote the vertical concatenation of $x_1, \ldots, x_\ell$, with x_k stacking on top of x_{k+1}. We use $(\{x_k\}_{k=1}^{\ell})$ as a shorthand notation for $(x_1, \ldots, x_\ell)$. We use the square bracket $x[i]$ and $X[i,j]$ to denote indexing. The size-n identity matrix is written as I_n; we suppress the subscript n whenever it can be inferred from context. We write $\mathbf{e}_i$ to denote the i-th canonical basis, i.e. the i-th column of the appropriate identity matrix. We write $\mathrm{diag}(X)$ to extract the diagonal from the matrix X, and $\mathrm{diag}(x)$ to convert length-n vector into an $n \times n$ diagonal matrix. The i-th largest eigenvalue of a matrix X is denoted $\lambda_i(X)$. We use $\odot$ to denote the elementwise product.

Consider the task of classifying a data point $\hat{x} \in \mathbb{R}^p$ as belonging to the $\hat{c}$-th of q classes. The standard approach is to train a classifier model $f : \mathbb{R}^p \to \mathbb{R}^q$ such that the prediction vector $f(\hat{x})$ takes on its maximum value at the $\hat{c}$-th element, as in $f(\hat{x})[\hat{c}] > f(\hat{x})[c]$ for all incorrect labels $c \neq \hat{c}$. In this paper, we focus our attention on ℓ-layer feed-forward ReLU-based neural networks, defined recursively

$$f(x) \equiv W_\ell x_\ell + b_\ell, \quad x_{k+1} = \max\{0, W_k x_k + b_k\}$$

for $k \in \{1, 2, \ldots, \ell\}$, where $x_1 \equiv x$ is the input. Throughout the paper, we will use n_k to denote the number of neurons at the k-th layer, and $n = \sum_{k=1}^{\ell} n_k$ to denote the total number of neurons. Note that our convention includes the neurons at the input layer, i.e. $x_1 \equiv x$, but excludes those at the output layer, i.e. $f(x) \equiv W_\ell x_\ell + b_\ell$.

To compute an adversarial example $x \approx \hat{x}$, the standard approach is to apply projected gradient descent (PGD) to the following *semi-targeted attack* problem, which was first introduced by (Carlini & Wagner, 2017):

$$\begin{aligned} \phi[c] = \min_{x=(x_1,\ldots,x_\ell)\in\mathbb{R}^n} \quad & w_\ell^T x_\ell + w_0 x_0 \\ \text{s.t.} \quad & x_{k+1} = \max\{0, W_k x_k + b_k\}, \\ & \|x_1 - \hat{x}\| \leq \rho, \end{aligned} \tag{A}$$

for all $k \in \{1, 2, \ldots, \ell - 1\}$, where $w_\ell = (\mathbf{e}_{\hat{c}} - \mathbf{e}_c)^T W_\ell$ and $w_0 = (\mathbf{e}_{\hat{c}} - \mathbf{e}_c)^T b_\ell$. [1]Robustness to adversarial perturbations can be certified by verifying that (A) achieves a positive global minimum $\phi[c] > 0$ for every incorrect class $c \neq \hat{c}$. The numerical value of the minimum global minimum, written $\phi^\star = \min_{c \neq \hat{c}} \phi[c]$, is a *robustness margin* that measures how robust the model is to adversarial perturbations. The more positive is the robustness margin $\phi^\star$, the more the model is able to resist misclassification.

3. Rank-constrained SDP relaxation

Our goal in this paper is to develop better lower-bounds on the semi-targeted attack problem (A). Following existing work on the SDP relaxation, we begin by substituting the rank-1 SDP reformulation of the ReLU activation in (Raghunathan et al., 2018b; Zhang, 2020) to (A). But instead of deleting the rank-1 constraint altogether, we propose to slightly relax it to a rank-r constraint with a bounded trace, where $1 \leq r \leq n + 1$, to result in a family of *nonconvex* relaxations

$$\phi_r[c] = \min_{X \in \mathbb{S}^{n+1}} \quad w_\ell^T x_\ell \tag{SDP-r}$$

subject to

$$\begin{aligned} & \mathrm{tr}(X_{1,1}) - 2x_1^T \hat{x}_1 + \|\hat{x}_1\|^2 x_0 \leq \rho^2 x_0, && (y_0) \\ & x_{k+1} \geq 0, \quad x_{k+1} \geq W_k x_k + b_k x_0, && (y_{k,1}, y_{k,2}) \\ & \mathrm{diag}(X_{k+1,k+1} - W_k X_{k,k+1} - b_k x_{k+1}^T) = 0, && (z_k) \\ & x_0 = 1, \quad \mathrm{tr}(X) \leq R^2 && (z_0, \mu) \end{aligned}$$

[1]To simplify presentation, we focus our attention on the ℓ_2 norm, and assume $w_0 = 0$ without loss of generality. Our results can be extended for the ℓ_∞ norm and are included in the appendix.

for all $k \in \{1, 2, \ldots, \ell - 1\}$, whose optimization variable X is an $(n+1) \times (n+1)$ rank-r constrained symmetric positive semidefinite matrix

$$X = \left[\begin{array}{c|ccc} x_0 & x_1^T & \cdots & x_\ell^T \\ \hline x_1 & X_{1,1} & \cdots & X_{1,\ell} \\ \vdots & \vdots & \ddots & \vdots \\ x_\ell & X_{1,\ell}^T & \cdots & X_{\ell,\ell} \end{array}\right] \succeq 0, \quad \operatorname{rank}(X) \leq r.$$

Here we assign the dual variable associated with each constraint in the parenthesis. We will assume throughout the paper that the trace bound R has been chosen large enough so that $\operatorname{tr}(X^\star) < R^2$, or equivalently $\mu^\star = 0$, holds at optimality. It follows from (Zhang, 2020) that $r = 1$ instance of (SDP-r) coincides with (A) exactly. Due to our use of a rank upper-bound, every subsequent instance then provides a lower-bound on its previous relaxation:

$$\phi[c] = \phi_1[c] \geq \phi_2[c] \geq \cdots \geq \phi_{n+1}[c].$$

Finally, setting $r = n+1$ has the same effect as deleting the rank constraint. Therefore, the $r = n+1$ instance coincides with the convex semidefinite relaxation as originally proposed by (Raghunathan et al., 2018b).

For relaxation ranks of $r < n+1$, the corresponding nonconvex instances of (SDP-r) are NP-hard in general to solve to global optimality. Even if we are provided with a globally optimal solution $X^\star$, there is generally no way to (rigorously) tell that $X^\star$ is indeed globally optimal. The most we can say is that $X^\star$ provides an upper-bound on the global minimum of (SDP-r). Unfortunately, this upper-bound is not helpful in our goal of lower-bounding (A).

Instead, we will derive a lower-bound on (SDP-r) via Lagrangian duality, which will also serve as a valid lower-bound on (A). Our motivating insight is that all instances of (SDP-r), including those nonconvex instances with $r < n+1$, have the *same* convex Lagrangian dual. Define dual variables $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1}) \geq 0$, $z = (y_0, \{z_k\}_{k=1}^{\ell-1})$ and $\mu \leq 0$ to correspond to the linear constraints in (SDP-r) as shown in parentheses. Then, the dual problem of (SDP-r) is written:

$$\max_{y \geq 0,\ z,\ \mu \leq 0} \ z_0 + R^2 \mu \quad \text{s.t.} \quad S(y,z) \succeq \mu I, \qquad \text{(SDD)}$$

in which the components of the slack matrix

$$S(y,z) \equiv \frac{1}{2} \left[\begin{array}{c|ccccc} s_0 & s_1^T & s_2^T & \cdots & s_\ell^T \\ \hline s_1 & S_{1,1} & S_{1,2} & & \\ s_2 & S_{1,2}^T & S_{2,2} & \ddots & \\ \vdots & & \ddots & \ddots & S_{\ell-1,\ell} \\ s_\ell & & & S_{\ell-1,\ell}^T & S_{\ell,\ell} \end{array}\right]$$

are written

$$\begin{aligned} s_0 &= 2\left[y_0(\|\hat{x}\|^2 - \rho^2) + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - z_0\right], \\ s_1 &= W_1^T y_{1,2} - 2\hat{x} y_0, \\ s_{k+1} &= W_{k+1}^T y_{k+1,2} - (Z_k b_k + y_{k,1} + y_{k,2}), \\ s_\ell &= w_\ell - [Z_{\ell-1} b_{\ell-1} + y_{\ell-1,1} + y_{\ell-1,2}], \\ S_{1,1} &= 2y_0 I,\ S_{k,k+1} = -W_k^T Z_k,\ S_{k+1,k+1} = 2Z_k, \end{aligned}$$

where $Z_k = \operatorname{diag}(z_k)$. Here, s_{k+1} is defined for all $k \in \{1, \ldots \ell-2\}$, and $S_{k,k+1}$ and $S_{k+1,k+1}$ are defined for all $k \in \{1, \ldots \ell-1\}$. We therefore obtain the following lower-bound on the semi-targeted attack problem in (A), which is valid for *any* choice of multipliers (y, z).

Proposition 3.1 (Dual lower-bound)**.** *Let $X^\star$ denote the global solution of (SDP-r) with rank $r \geq 1$ and $\operatorname{tr}(X^\star) < R^2$. Then, any dual multipliers $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$ and $z = (z_0, \{z_k\}_{k=1}^{\ell-1})$ that satisfy $y \geq 0$ provide the following lower-bound*

$$\phi[c] \geq \phi_r[c] \geq z_0 + R^2 \cdot \min\{0, \lambda_{\min}[S(y,z)]\}.$$

Let $X^\star$ denote the globally optimal solution for the convex instance of (SDP-r) with $r = n+1$. It turns out that *strong duality* is satisfied in this convex case, meaning that there exists optimal multipliers $y^\star, z^\star$ that exactly satisfy

$$\phi[c] \geq \phi_{n+1}[c] = z_0^\star + R^2 \cdot \min\{0, \lambda_{\min}[S(y^\star, z^\star)]\}$$

and therefore *certify* the global optimality $X^\star$ via Proposition 3.1. Now, suppose that the convex solution $X^\star$ is in fact low-rank, as in $r^\star = \operatorname{rank}(X^\star) \ll n$. The statement below says that the *nonconvex* instance of (SDP-r) with $r = r^\star$ also admits optimal multipliers $y^\star, z^\star$ that certify global optimality.

Theorem 3.2 (Existence of global optimality certificate)**.** *Let $r^\star = \operatorname{rank}(X^\star)$ and $\operatorname{tr}(X^\star) < R^2$, where $X^\star$ denotes the maximum-rank solution to the convex instance of (SDP-r) with $r = n+1$. Then, there exists optimal multipliers $y^\star = (y_0^\star, \{y_{k,1}^\star, y_{k,2}^\star\}_{k=1}^{\ell-1})$ and $z^\star = (z_0^\star, \{z_k^\star\}_{k=1}^{\ell-1})$ that satisfy $y^\star \geq 0$ and the following*

$$\phi[c] \geq \phi_{r^\star}[c] = z_0^\star + R^2 \cdot \min\{0, \lambda_{\min}[S(y^\star, z^\star)]\}.$$

In the following section, we use an approach of Burer & Monteiro (2003) and (Boumal et al., 2016; 2020) to constructively compute the optimal multipliers $y^\star, z^\star$ that have been asserted to exist by Theorem 3.2. In turn, plugging $y^\star, z^\star$ into Proposition 3.1 produces a tight lower-bound on the semi-targeted attack problem (A), thereby achieving the original goal of this section.

4. Solution via Nonlinear Programming

In order to expose the underlying degrees of freedom in the rank-r matrix X, we reformulate problem (SDP-r) into a low-rank factorization form first proposed by (Burer & Monteiro, 2003):

$$\phi_r[c] = \min_{u_0,u,V} \quad u_0 \cdot (w_\ell^T u_\ell) \qquad \text{(BM-}r\text{)}$$

subject to

$$\|u_1 - u_0\hat{x}\|^2 + \|V_1\|^2 \le \rho^2, \quad u_0^2 = 1, \qquad (y_0, z_0)$$

$$u_0 \cdot u_{k+1} \ge 0, \qquad (y_{k,1})$$

$$u_0 \cdot (u_{k+1} - W_k u_k - b_k u_0) \ge 0, \qquad (y_{k,2})$$

$$\begin{aligned}\operatorname{diag}[(u_{k+1} - W_k u_k - b_k u_0)u_{k+1}^T \\ + (V_{k+1} - W_k V_k)V_{k+1}^T] = 0,\end{aligned} \qquad (z_k)$$

$$u_0^2 + \sum_{k=1}^{\ell-1}(\|u_k\|^2 + \|V_k\|^2) \le R^2, \qquad (\mu)$$

for all $k \in \{1, \dots, \ell-1\}$, and over optimization variables are $u_0 \in \mathbb{R}$ and $u = (u_1, \dots, u_\ell) \in \mathbb{R}^n$ and $V = (V_1, \dots, V_\ell) \in \mathbb{R}^{n\times(r-1)}$. Problem (BM-$r$) is obtained by substituting the following into (SDP-r)

$$X = \left[\begin{array}{c|c} u_0 & 0 \\ \hline u_1 & V_1 \\ \vdots & \vdots \\ u_\ell & V_\ell \end{array}\right] \left[\begin{array}{c|c} u_0 & 0 \\ \hline u_1 & V_1 \\ \vdots & \vdots \\ u_\ell & V_\ell \end{array}\right]^T = UU^T. \qquad (1)$$

The equivalence between these two problems follows because every $(n+1)\times(n+1)$ matrix X of rank r can be factored as $X = LL^T$ into a low-rank Cholesky factor L that is both lower-triangular and of dimensions $(n+1)\times r$. The advantage of the formulation (BM-r) is that it reduces the number of explicit variables from the $\frac{1}{2}n(n+1) \approx \frac{1}{2}n^2$ in the original matrix X to $nr+1 \approx nr$ in the factor matrix U, while also allowing the positive semidefinite constraint $X \succeq 0$ to be enforced for free. For moderate values of $r \ll n$, the resulting instance of (BM-r) contains just $O(n)$ variables and constraints.

We propose solving (BM-r) as an instance of the standard-form nonlinear program,

$$\min_{\|x\|\le R} \quad f(x) \quad \text{s.t.} \quad g(x) \le 0, \quad h(x) = 0, \qquad \text{(NLP)}$$

using a high-performance general-purpose solver like `fmincon` or `knitro`. These are primal-dual solvers, and are designed to output a primal point $x = (u_0, u, \operatorname{vec}(V))$ that is *first-order optimal*, and dual multipliers $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$ and $z = (z_0, \{z_k\}_{k=1}^{\ell-1})$ that *certify* the first-order optimality of x. Below, the notion of first-order optimality is taken from (Nocedal & Wright, 2006, Theorem 12.3), and the notion of certifiability follows from the proof of (Nocedal & Wright, 2006, Theorem 12.1).

Definition 4.1 (Certifiably first-order optimal)**.** The point x is said to be *first-order optimal* if it satisfies the constraints $g(x) \le 0$ and $h(x) = 0$, and there exists no escape path $x(t)$ that begins at $x(0) = x$ and makes a first-order improvement to the objective while satisfying all constraints, as in

$$f(x(t)) \le f(x) - \delta t, \quad g(x(t)) \le 0, \quad h(x(t)) = 0,$$

for all $t \in [0, \epsilon)$ with sufficiently small $\delta > 0$ and $\epsilon > 0$. Additionally, x is said to be *certifiably* first-order optimal if there exist dual multipliers y and z that satisfy the Karush–Kuhn–Tucker (KKT) equations:

$$\nabla f(x) + \nabla g(x)y + \nabla h(x)z = 0, \; y \odot g(x) = 0, \; y \ge 0.$$

Our main idea is to simply take the dual multipliers y, z computed by the nonlinear programming solver, round them $y \leftarrow \max\{0, y\}$ to ensure that $y \ge 0$, and then to plug them back into Proposition 3.1. Our main result is that, if x is globally optimal and satisfies a mild constraint qualification, then the corresponding dual multipliers y, z exist and are unique. Therefore, if x is indeed globally optimal, then the dual multipliers y, z that certify the local optimality of x must also coincide with the optimal multipliers $y^\star, z^\star$ that were asserted to existed earlier in Theorem 3.2.

Lemma 4.2 (Nonzero preactivation)**.** *Suppose we have $x = (u_0, u_1, \dots, u_\ell, \operatorname{vec}(V_1), \dots, \operatorname{vec}(V_\ell))$ that satisfies:*

$$\mathbf{e}_i^T(W_k u_k + b_k u_0) \ne 0, \quad \mathbf{e}_i^T W_k V_k \ne 0, \qquad \text{(NPCQ)}$$

for all $k \in \{1, \dots, \ell-1\}$ and $i \in \{1, \dots, n_{k+1}\}$. Then, x is first-order optimal if and only if there exist dual multipliers y and z to certify x as being first-order optimal. Moreover, the choice of dual multipliers y, z is unique.

Theorem 4.3 (Zero duality gap)**.** *Let $r \ge r^\star$ where $r^\star$ is defined in Theorem 3.2. If x is globally optimal and satisfies (NPCQ), then the dual multipliers y and z that certify x to be first-order optimal must also certify x to be globally optimal, as in*

$$\phi_r[c] = u_0 \cdot (w_\ell^T u_\ell) = z_0 + R^2 \cdot \max\{0, \lambda_{\min}[S(y,z)]\}.$$

Conversely, if a first-order optimal point x satisfies the constraint qualification but is not globally optimal, then the dual multipliers y, z generate a direction of global improvement towards the global minimum. The key idea is to lift to a higher relaxation rank $r_+ = r+1$, in order to make x a saddle point. The statement below gives a direction to escape the saddle-point and make a decrement.

Theorem 4.4 (Escape lifted saddle point)**.** *Let x be certifiably first-order optimal for (BM-r) with dual multipliers (y, z). If x satisfies $\gamma = -\lambda_{\min}[S(y,z)] > 0$, (NPCQ) and $\|x\| < R$, then the eigenvector $\xi = (\xi_0, \xi_1, \xi_2, \dots, \xi_\ell)$ that*

satisfies $\xi^T S(y,z)\xi = -\gamma\|\xi\|^2$ *implicitly defines an escape path* $x_+(t) = (u_0, \{u_{k,+}(t)\}_{k=1}^{\ell}, \{V_{k,+}(t)\}_{k=1}^{\ell})$ *with*

$$u_{k,+}(t) = u_k + O(t^2),$$
$$V_{k,+}(t) = [V_k, 0] + t \cdot [0, u_k\xi_0/u_0 + \xi_k] + O(t^2)$$

that makes a second-order improvement to the objective while satisfying all constraints, as in

$$f(x_+(t)) = f(x) - t^2\gamma, \quad g(x_+(t)) \le 0, \quad h(x_+(t)) = 0$$

for all $t \in [0, \epsilon)$ *with sufficiently small but nonzero* $\epsilon > 0$.

In practice, it suffices to move along the *straight* path $\tilde{u}_{k,+}(t) = u_k$ and $\tilde{V}_{k,+}(t) = [V_k, 0] + t \cdot [0, u_k\xi_0/u_0 + \xi_k]$ then solve for feasibility $g(x) \le 0$ and $h(x) = 0$. Concretely, after computing a first-order optimal x, we increment the relaxation rank $r_+ = r + 1$, and initialize the nonlinear programming solver using the lifted point $\tilde{x}_+(\epsilon)$ as the initial primal point, and the old multipliers y, z as the initial dual multipliers. If this arrives at the global optimum, then the corresponding y, z must certify x as being so. Otherwise, we repeat the rank lifting procedure.

Progressively lifting the relaxation rank r, in our experience it takes no more than $r \le 10$ to reduce the duality gap to values of 10^{-8}. To rigorously guarantee a zero duality gap, however, can require a relaxation rank on the order of $r = O(\sqrt{n})$ (Boumal et al., 2020), irrespective of the value of $r^\star$. Indeed, counterexamples with $\epsilon_{\text{feas}} > 0$ exist for relaxation ranks r that are even slightly smaller than this threshold (Waldspurger & Waters, 2020; O'Carroll et al., 2022; Zhang, 2022). As a purely theoretical result, the ability to achieve a zero duality gap implies that the nonconvex relaxation (with $r \ge r^\star$) can be solved in polynomial time (and is therefore not NP-hard). Of course, setting $r = O(\sqrt{n})$ would also force us to optimize over $O(n^{3/2})$ variables, thereby offsetting much of our computational advantage against the usual convex SDP relaxation in practice.

Algorithm 1 summarizes our proposed approach in pseudocode form, and introduces a number of small practical refinements.

Algorithm 1 Summary of proposed algorithm

Input: Initial relaxation rank $r \ge 2$. Weights $W_1, \ldots, W_\ell$ and biases $b_1, \ldots, b_\ell$. Original input $\hat{x}$, true label $\hat{c}$, target label c, and perturbation size ρ. Variable radius bound R.
Output: Lower-bound $\phi_{\text{lb}}[c] \le \phi[c]$ on the optimal value of the semi-targeted attack problem (A).
Algorithm:

1. (Solve rank-r relaxation) Use a nonlinear programming solver to solve the following

$$\min_{\|x\| \le R} \quad f(x) \equiv (\mathbf{e}_c - \mathbf{e}_{\hat{c}})^T (W_\ell u_\ell u_0 + b_\ell)$$

subject to

$$g_0(x) \equiv \|u_1 - u_0\hat{x}\|^2 + \|V_1\|^2 - \rho^2 \le 0, \qquad (y_0)$$
$$h_0(x) \equiv 1 - u_0^2 = 0, \qquad (z_0)$$
$$g_k(x) \equiv \begin{bmatrix} -u_0 u_{k+1} \\ u_0(W_k u_k + b_k u_0 - u_{k+1}) \end{bmatrix} \le 0, \qquad \begin{matrix}(y_{k,1}) \\ (y_{k,2})\end{matrix}$$
$$h_k(x) \equiv \operatorname{diag}[(u_{k+1} - W_k u_k - b_k u_0)u_{k+1}^T + (V_{k+1} - W_k V_k)V_{k+1}^T] = 0, \qquad (z_k)$$

for all k and over $x = (u_0, \{u_k\}_{k=1}^{\ell}, \{\text{vec}(V_k)\}_{k=1}^{\ell})$. Retrieve the corresponding dual multipliers $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$ and $z = (z_0, \{z_k\}_{k=1}^{\ell-1})$.

2. (Check certifiable first-order optimality) If $\|\nabla f(x) + \nabla g(x)y + \nabla h(x)z\|$ is sufficiently small, and if $g(x) \le 0$ and $h(x) = 0$ and $\|x\| < R$ hold to sufficient tolerance, then continue. Otherwise, return error due to solver's inability to achieve first-order optimality.

3. (Check dual feasibility) If $\epsilon_{\text{feas}} = -\lambda_{\min}[S(y,z)]$ is sufficiently small, where the slack matrix $S(y,z)$ is defined in (SDD), then return $\phi_{\text{lb}}[c] = z_0 - \epsilon_{\text{feas}} \cdot R^2$. Otherwise, continue.

4. (Escape lifted saddle point) Compute the eigenvector $\xi = (\xi_0, \xi_1, \ldots, \xi_\ell)$ satisfying $\|\xi\| = 1$ and $\xi^T S(y,z)\xi = -\epsilon_{\text{feas}}$. Set up new primal initial point $x_+ = (u_0, \{u_k\}_{k=1}^{\ell}, \{\text{vec}(V_{+,k})\}_{k=1}^{\ell})$ where

$$V_{+,k} = [V_k, 0] + \epsilon \cdot [0, u_k\xi_0/u_0 + \xi_k].$$

Increment $r \leftarrow r + 1$ and repeat Step 1 with (x_+, y, z) as the initial point.

5. Experiments

We identically reproduce three models from Salman et al. (2019), two of which were trained to be robust against an ℓ_∞ adversary. We compare the performance of our proposed verifier `BM`, which is based on solving (BM-r), and `BM-Full`, which is an extension of `BM` with the addition of layer-wise preactivation bounds (see Appendix A for details), against state-of-the-art LP-based verifiers for certifying robustness against an ℓ_2 adversary. Our experiments for certifying the robustness of the same models against an ℓ_∞ adversary are deferred to the appendix.

Methods. `BM` and `BM-Full` denote the proposed method without and with preactivation bounds respectively. The source code for `BM` and `BM-Full` are available at `https://github.com/Hong-Ming/BM-r`. `PGD` denotes the projected gradient descent algorithm for finding the upper bound on (A). We compare `BM` and `BM-Full`

to three state-of-the-art LP-based verifiers: `CROWN-Ada` of Zhang et al. (2018b), `Fast-Lip` of Weng et al. (2018a), and `LP-Full` of Salman et al. (2019). `CROWN-Ada` and `Fast-Lip` are both large-scale LP verifiers that have linear complexity with respect to the number of activations; the implementations that we used were taken directly from the authors' project page[2]. `LP-Full` is the optimal LP verifier that uses the tightest possible preactivation bounds by solving LP problems for each hidden neuron; its complexity is cubic with respect to the number of activations. We reimplemented this algorithm to work with ℓ_2 adversaries, and then validated our implementation against that of the authors[3] on ℓ_∞ adversaries. The preactivation bounds in `BM-Full` are set to coincide with those used in `LP-Full`. We also compare our methods against the state-of-the-art branch-and-bound verifier α, β-`CROWN` (Zhang et al., 2018a; Wang et al., 2021; Xu et al., 2021; Zhang et al., 2022). The implementation of α, β-`CROWN` are taken from authors' project page [4]. We set the timeout of α, β-`CROWN` to be 300s.

Setup. We use an Apple laptop, running a silicon M1 pro chip with 10-core CPU, 16-core GPU, and 32GB of RAM. We implemented `BM` and `BM-Full` in MATLAB. The nonconvex problem (BM-r) is solved using the trust-region interior-point solver `knitro` (Byrd et al., 2006) with a warm-start strategy.

Models. We perform simulation on three models: NOR-MNIST, ADV-MNIST and LPD-MNIST. All three models are trained by Salman et al. (2019) and are taken directly from the authors' project page[3]. In particular, the architecture and the numerical values of the weights for NOR-MNIST, ADV-MNIST and LPD-MNIST are identical to NOR-MLP-B, ADV-MLP-B and LPD-MLP-B respectively in Salman et al. (2019). All three models are fully-connected feedforward neural network models with 2 hidden layers of 100 neurons each, and were trained on the MNIST dataset with different training procedures. NOR-MNIST was trained normally using the cross-entropy loss and used as a control. ADV-MNIST was adversarially trained against the PGD attack using the method of Madry et al. (2018), with ℓ_∞ radius of 0.1. LPD-MNIST was robustly trained via the adversarial polytope perturbation of Wong & Kolter (2018), with the size of the adversarial polytope also set to 0.1.

[2] https://github.com/IBM/CROWN-Robustness-Certification
[3] https://github.com/Hadisalman/robust-verify-benchmark
[4] https://github.com/Verified-Intelligence/alpha-beta-CROWN

5.1. Robustness verification on neural network inputs.

We certify the robustness of our three models against an ℓ_2 adversary using our proposed method, and compare their performance against the existing state-of-the-art. In each trial, we fix an attack radius ρ, and mark a correctly-classified image $\hat{x}$ as *robust* if the lower bound on (A) is positive with respect to all incorrect classes, i.e. $0 < \phi_{\text{lb}}[c] \le \phi[c]$ for all $c \neq \hat{c}$, and mark an image $\hat{x}$ as *not robust* if an attack is found by `PGD`, i.e. $\phi[c] \le \phi_{\text{ub}}[c] < 0$ for some $c \neq \hat{c}$. We mark the image as *status unknown* if a determination cannot be made either way. The lower bound for `BM` and `BM-Full` is obtained by Proposition 3.1.

Results and discussions. Table 1 shows the number of images that are certified *robust* within the first 1000 correctly classified images using `BM-Full`, `BM`, α, β-`CROWN`,

Table 1: **Robustness verification for neural networks.** We compare the number of images certified as *robust* by `BM-Full`, `BM`, α, β-`CROWN`, `LP-Full`, `CROWN-Ada` and `Fast-Lip` within the first 1000 images for normally and robustly trained networks. The upper bound (denoted as UB in the table) on the true number of robust images is obtained by `PGD`.

Network	ℓ_2 Radius	`PGD`	`BM-Full`		`BM`		α, β-`CROWN`		`LP-Full`		`CROWN-Ada`		`Fast-Lip`	
		UB	Robust	Time	Robust	Time	Robust	Time	Robust	Time	Robust	Time	Robust	Time
ADV-MNIST	1.0	774	**762**	47s	757	28s	672	24s	209	8s	45	12ms	30	13ms
ADV-MNIST	1.3	614	**569**	38s	559	28s	399	94s	25	10s	7	9ms	2	12ms
ADV-MNIST	1.5	471	**411**	56s	392	20s	248	138s	11	11s	1	9ms	1	13ms
LPD-MNIST	1.0	755	**730**	218s	708	29s	641	10s	411	16s	120	10ms	66	13ms
LPD-MNIST	1.3	612	**514**	129s	474	26s	430	33s	61	19s	16	10ms	8	13ms
LPD-MNIST	1.5	505	**391**	98s	350	23s	316	64s	23	20s	5	10ms	2	14ms
NOR-MNIST	0.3	916	**911**	128s	866	21s	797	23s	728	8s	420	9ms	348	12ms
NOR-MNIST	0.5	732	**696**	127s	534	27s	424	159s	232	16s	46	7ms	27	13ms
NOR-MNIST	0.7	485	**381**	156s	187	30s	124	253s	37	19s	0	13ms	0	17ms

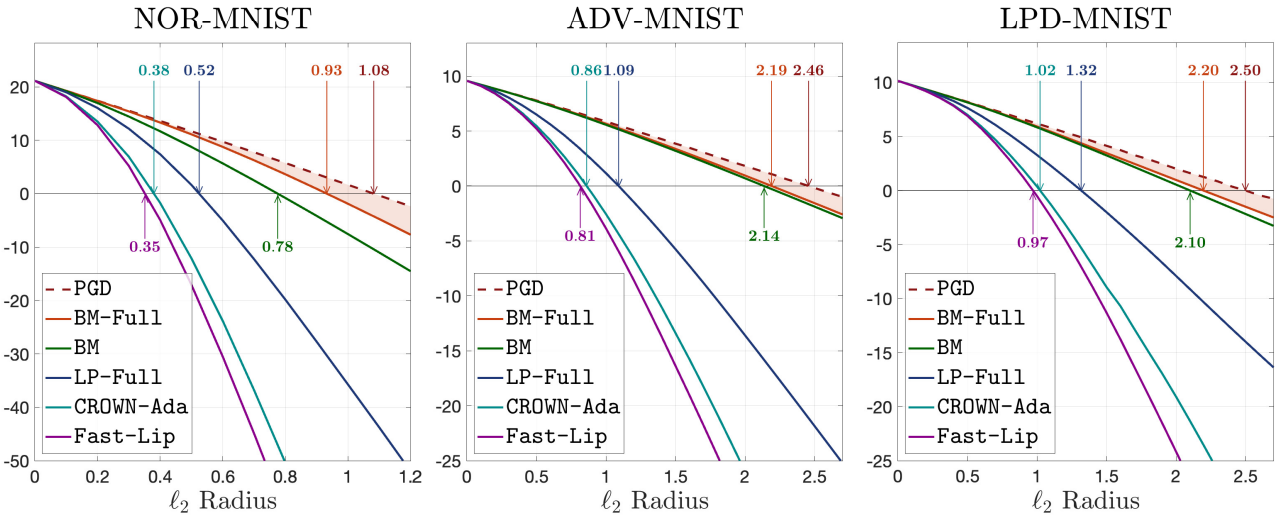

Figure 2: **Lower bounds on the robustness margin**. We take `BM-Full`, `BM`, `LP-Full`, `CROWN-Ada` and `Fast-Lip` to compute their average lower bound on (A), and then compare them to the average `PGD` upper bound on (A) over a wide range of ℓ_2 perturbation radius. **(Left.)** NOR-MNIST. **(Middle.)** ADV-MNIST. **(Right.)** LPD-MNIST.

`LP-Full`, `CROWN-Ada` and `Fast-Lip`. The average computation time per image for each verifier are also shown. In addition, to gauge the efficacy of our verifiers, we benchmark our results against the upper bound on the true number of *robust* images (denoted as UB in the table), which is the number of images that does not get marked as *not robust* by `PGD`. From the table, we see that our nonconvex verifiers are able to consistently outperform all the other verifiers under all cases, with time complexity only 5 to 10 times higher than `LP-Full`. Despite higher time complexity, our verifiers are the only verifiers that can still verify reasonable amount of images under larger perturbation radius. Notably, for small perturbation, our verifiers can nearly certify all images that cannot be attacked by `PGD`, leaving very few images as *status unknown*.

5.2. Tightness of our lower bound

Since robust verification is an NP-hard problem, all relaxation methods must become loose for sufficiently large radius. Fortunately, robust verification is only needed when the `PGD` upper bound of (A) is positive; therefore, we only need the relaxation to be tight when the `PGD` upper bound is still positive. In this experiment, we analyze the gap between our lower bound in Proposition 3.1 and the `PGD` upper bound over a wide range of perturbation radius. To accurately measure the gap between our lower bound and the `PGD` upper bound, we average each bound over all 9 incorrect classes of the first 10 correctly classified images in the test set, in total 90 samples are considered.

Results and discussions. Figures 2 shows the average `PGD` upper bound, and the average lower bound on (A) computed from `BM`, `BM-Full`, `LP-Full`, `CROWN-Ada` and `Fast-Lip`. Our lower bounds are significantly tighter than all the other verifiers across a wide range of ℓ_2 perturbation radius. Most importantly, our lower bounds are able to remain tight in regions where the `PGD` upper bound is still positive. We reiterate that it is not possible for our nonconvex verifiers to be exact with a large perturbation radius, because exact verification is NP-hard and our algorithm is polynomial-time. Nonetheless, so long as we can remain tight as the upper-bound crosses the zero line, our certification methods will be very close to exact.

6. Conclusion

In this work, we presented a neural network certification technique based on a nonconvex low-rank restricted SDP relaxation. Our experiments find that the method is able to overcome the convex relaxation barrier (Salman et al., 2019) with runtime only a small constant factor (5-10$\times$) worse than the existing state-of-the-art. Our results showed that even a basic nonconvex relaxation, `BM`, offers a significant reduction in relaxation gap, while augmenting with bound propagation, `BM-Full`, allows us to almost fully close the gap towards exact certification.

Acknowledgments

The authors thank Zico Kolter for insightful discussions and pointers to the literature. Financial support for this work was provided by the NSF CAREER Award ECCS-2047462 and C3.ai Inc. and the Microsoft Corporation via the C3.ai Digital Transformation Institute.

References

Batten, B., Kouvaros, P., Lomuscio, A., and Zheng, Y. Efficient neural network verification via layer-based semidefinite relaxations and linear cuts. In *IJCAI*, pp. 2184–2190, 2021.

Boumal, N., Voroninski, V., and Bandeira, A. The non-convex Burer-Monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems*, pp. 2757–2765, 2016.

Boumal, N., Voroninski, V., and Bandeira, A. S. Deterministic guarantees for Burer-Monteiro factorizations of smooth semidefinite programs. *Communications on Pure and Applied Mathematics*, 73(3):581–608, 2020.

Burer, S. and Monteiro, R. D. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

Burer, S. and Monteiro, R. D. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.

Burer, S., Monteiro, R. D., and Zhang, Y. Rank-two relaxation heuristics for max-cut and other binary quadratic programs. *SIAM Journal on Optimization*, 12(2):503–521, 2002.

Byrd, R. H., Nocedal, J., and Waltz, R. A. Knitro: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pp. 35–59. Springer, 2006.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017.

Carlone, L., Rosen, D. M., Calafiore, G., Leonard, J. J., and Dellaert, F. Lagrangian duality in 3d slam: Verification techniques and optimal solutions. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 125–132. IEEE, 2015.

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320, 2019.

Dathathri, S., Dvijotham, K., Kurakin, A., Raghunathan, A., Uesato, J., Bunel, R., Shankar, S., Steinhardt, J., Goodfellow, I., Liang, P., et al. Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. In *Advances in Neural Information Processing Systems*, 2020.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

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

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

MOSEK, A. The MOSEK optimization toolbox for MATLAB manual, 2019. URL https://docs.mosek.com/9.0/toolbox.pdf.

Nocedal, J. and Wright, S. *Numerical optimization*. Springer Science & Business Media, 2006.

O'Carroll, L., Srinivas, V., and Vijayaraghavan, A. The burer-monteiro sdp method can fail even above the barvinok-pataki bound. In *Advances in Neural Information Processing Systems*, 2022.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018a.

Raghunathan, A., Steinhardt, J., and Liang, P. S. Semidefinite relaxations for certifying robustness to adversarial examples. *Advances in Neural Information Processing Systems*, 31, 2018b.

Rosen, D. M., Kaess, M., and Leonard, J. J. Rise: An incremental trust-region method for robust online sparse least-squares estimation. *IEEE Transactions on Robotics*, 30(5):1091–1108, 2014.

Rosen, D. M., Carlone, L., Bandeira, A. S., and Leonard, J. J. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019.

Salman, H., Yang, G., Zhang, H., Hsieh, C.-J., and Zhang, P. A convex relaxation barrier to tight robustness verification of neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

Tjeng, V., Xiao, K., and Tedrake, R. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019.

Waldspurger, I. and Waters, A. Rank optimality for the burer–monteiro factorization. *SIAM journal on Optimization*, 30(3):2577–2602, 2020.

Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C.-J., and Kolter, J. Z. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems*, 34:29909–29921, 2021.

Weng, L., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Daniel, L., Boning, D., and Dhillon, I. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pp. 5276–5285. PMLR, 2018a.

Weng, T.-W., Zhang, H., Chen, P.-Y., Yi, J., Su, D., Gao, Y., Hsieh, C.-J., and Daniel, L. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*, 2018b.

Wong, E. and Kolter, Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pp. 5286–5295, 2018.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2019.

Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., and Hsieh, C.-J. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representation (ICLR)*, 2021.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31, 2018a.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31, 2018b.

Zhang, H., Wang, S., Xu, K., Li, L., Li, B., Jana, S., Hsieh, C.-J., and Kolter, J. Z. General cutting planes for bound-propagation-based neural network verification. *Advances in Neural Information Processing Systems*, 2022.

Zhang, R. Y. On the tightness of semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, 2020.

Zhang, R. Y. Improved global guarantees for the nonconvex burer–monteiro factorization via rank overparameterization. *arXiv preprint arXiv:2207.01789*, 2022.

A. Implementation details for BM and BM-Full

In this section, we present the implementation detail for our two proposed methods: BM, the nonconvex relaxation (BM-r) proposed in the main paper; and BM-Full, an extension of BM obtained by adding preactivation bounds on each hidden neuron in (BM-r). We focus our attention on how to efficiently implement both methods to verify ℓ_2 and ℓ_∞ adversaries for neural networks trained on MNIST dataset.

This section consists of three parts. First, we describe the valid input set constraint that we need to add into (BM-r) in order to certify MNIST images, and a few constraints in (BM-r) that can be simplified for improving efficiency. Second, in Appendix A.1 to A.4, we summarized the practical and efficient formulation for BM and BM-Full with respect to both ℓ_2 and ℓ_∞ perturbation. Third, in Appendix A.5, we present more details on how to efficiently solve BM and BM-Full using the procedure described in Algorithm 1.

Valid input set constraints For model train on MNIST dataset, we add an extra constraint $0 \leq u_0 \cdot u_1 \leq 1$ into (BM-r) because MNIST images are normalized to between 0 and 1 during training and testing. Notice that adding an extra inequality constraint $0 \leq u_0 \cdot u_1 \leq 1$ does not alter any theoretical results in this paper as it only add an extra term to s_1 in the slack matrix $S(y, z)$.

Simplify constraints in (BM-r) In our practical implementation, we fix u_0 to 1 in (BM-r). The reason is twofold. First, by fixing $u_0 = 1$, most constraints in (BM-r) become linear, and hence reduces the time complexity of our algorithm significantly. Second, the dual variable z_0, which is associated with the constraint $u_0 = 1$, can be solved via the KKT condition $S(x, y)U = 0$.

A.1. Efficient formulation of BM for ℓ_2 norm

We now turn to the practical aspect of implementing our proposed method BM. In particular, to verify ℓ_2 adversaries of neural networks trained on MNIST dataset, we solve (BM-r) in the following form

$$
\begin{aligned}
\phi_r[c] = \min_{u,V} \quad & w_\ell^T u_\ell && \text{(BM-}\ell_2\text{)}\\
\text{s.t.} \quad & \|u_1 - \hat{x}\|^2 + \|V_1\|^2 \leq \rho^2, && (y_0)\\
& u_1 \geq 0, \qquad u_1 \leq 1 && (y_{0,1}, y_{0,2})\\
& u_{k+1} \geq 0, \qquad u_{k+1} - W_k u_k - b_k \geq 0, && (y_{k,1}, y_{k,2})\\
& \operatorname{diag}\left[(u_{k+1} - W_k u_k - b_k)u_{k+1}^T + (V_{k+1} - W_k V_k)V_{k+1}^T\right] = 0, && (z_k)\\
& 1 + \sum_{k=1}^{\ell-1}(\|u_k\|^2 + \|V_k\|^2) \leq R^2, && (\mu)
\end{aligned}
$$

for $k \in \{1, \ldots, \ell - 1\}$. Notice that we have substituted $u_0 = 1$, added the valid input set constraints $u_1 \geq 0$ and $u_1 \leq 1$, and assigned their associated dual variable $y_{0,1}$ and $y_{0,2}$. In Definition A.1, we summarized how to evaluate the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) using the primal and dual solution of (BM-ℓ_2) in order to calculate the bound in Proposition 3.1.

Definition A.1. Let $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=0}^{\ell-1})$, $z = (\{z_k\}_{k=1}^{\ell-1})$, $u = (u_1, \ldots, u_\ell)$ and $V = (V_1, \ldots, V_\ell)$ be any *certifiably first-order optimal* point of (BM-ℓ_2). Each component in the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) can be evaluated as

$$
\begin{gathered}
s_0 = -\sum_{k=1}^{\ell} s_k^T u_k, \quad z_0 = y_0(\|\hat{x}\|^2 - \rho^2) - 1^T y_{0,2} + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - \frac{1}{2}s_0,\\
s_1 = W_1^T y_{1,2} - 2\hat{x}y_0 - y_{0,1} + y_{0,2}, \quad s_\ell = w_\ell - \left[Z_{\ell-1}b_{\ell-1} + y_{\ell-1,1} + y_{\ell-1,2}\right],\\
s_{k+1} = W_{k+1}^T y_{k+1,2} - (Z_k b_k + y_{k,1} + y_{k,2}) \qquad \text{for } k \in \{1, \ldots, \ell - 2\},\\
S_{1,1} = 2y_0 I, \quad S_{k,k+1} = -W_k^T Z_k, \quad S_{k+1,k+1} = 2Z_k \qquad \text{for } k \in \{1, \ldots, \ell - 1\},
\end{gathered}
$$

where $Z_k = \operatorname{diag}(z_k)$ for all k.

A.2. Efficient formulation of `BM-Full` for ℓ_2 norm

We now describe the practical formulation for `BM-Full`. Let lb_k and ub_k denote the lower bound and the upper bound on preactivation neurons in the k-th layer. lb_k and ub_k gives us the postactivation bound constraints $\max\{lb_k, 0\} \leq x_k \leq \max\{ub_k, 0\}$ for each postactivation neuron x_k in (SDP-r) . To incorporate these bound constraints into (BM-r), we first rewrite each of them into an elementwise ℓ_2 constraint for which $\mathbf{e}_i^T x_k$ is restricted in a ℓ_2 norm ball centered at $\frac{1}{2}\mathbf{e}_i^T(\max\{ub_k, 0\} + \max\{lb_k, 0\})$ with radius $\frac{1}{2}\mathbf{e}_i^T(\max\{ub_k, 0\} - \max\{lb_k, 0\})$ as in

$$\max\{lb_k, 0\} \leq x_k \leq \max\{ub_k, 0\} \quad \Longleftrightarrow \quad \|\mathbf{e}_i^T x_k - \mathbf{e}_i^T \hat{x}_k\|^2 \leq \rho_k^2 \quad \text{for all } \ i \in \{1, \dots, n_k\}$$

where $\hat{x}_k = \frac{1}{2}(\max\{ub_k, 0\} + \max\{lb_k, 0\})$ and $\rho_k = \frac{1}{2}(\max\{ub_k, 0\} - \max\{lb_k, 0\})$. The above elementwise ℓ_2 constraint has the following Burer-Monteiro formulation

$$\operatorname{diag}\left[(u_{k+1} - \hat{x}_{k+1})(u_{k+1} - \hat{x}_{k+1})^T + V_{k+1}V_{k+1}^T\right] \leq \rho_{k+1}^2$$

for $k \in \{1, \dots, \ell - 1\}$. In turn, to verify neural networks trained on MNIST dataset with respect to ℓ_2 perturbation, we add the above bound constraints into (BM-r) and solve the following

$$\begin{aligned}
\phi_r[c] = \min_{u,V} \quad & w_\ell^T u_\ell && \text{(BM-Full-}\ell_2\text{)} \\
\text{s.t.} \quad & \|u_1 - \hat{x}\|^2 + \|V_1\|^2 \leq \rho^2, && (y_0) \\
& u_1 \geq 0, \qquad u_1 \leq 1 && (y_{0,1}, y_{0,2}) \\
& \operatorname{diag}\left[(u_{k+1} - \hat{x}_{k+1})(u_{k+1} - \hat{x}_{k+1})^T + V_{k+1}V_{k+1}^T\right] \leq \rho_{k+1}^2 && (y_k) \\
& u_{k+1} - W_k u_k - b_k \geq 0, && (y_{k,2}) \\
& \operatorname{diag}\left[(u_{k+1} - W_k u_k - b_k)u_{k+1}^T + (V_{k+1} - W_k V_k)V_{k+1}^T\right] = 0, && (z_k) \\
& 1 + \sum_{k=1}^{\ell-1}(\|u_k\|^2 + \|V_k\|^2) \leq R^2, && (\mu)
\end{aligned}$$

for $k \in \{1, \dots, \ell - 1\}$. Notice that we delete the constraints $u_k \geq 0$ because they overlap with the bound constraints $\max\{lb_k, 0\} \leq u_k$. In Definition A.2, we summarized how to evaluate the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) using the primal and dual solution of (BM-Full-ℓ_2) in order to calculate the bound in Proposition 3.1.

Definition A.2. Let $y = (y_0, y_{0,1}, y_{0,2}, \{y_k, y_{k,2}\}_{k=1}^{\ell-1})$, $z = (\{z_k\}_{k=1}^{\ell-1})$, $u = (u_1, \dots, u_\ell)$ and $V = (V_1, \dots, V_\ell)$ be any *certifiably first-order optimal* point of (BM-Full-ℓ_2). Each component in the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) can be evaluated as

$$\begin{gathered}
s_0 = -\sum_{k=1}^{\ell} s_k^T u_k, \quad z_0 = y_0(\|\hat{x}\|^2 - \rho^2) + \sum_{k=1}^{\ell-1} y_k^T(\hat{x}_{(k+1)}^2 - \rho_{(k+1)}^2) - 1^T y_{0,2} + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - \frac{1}{2}s_0, \\
s_1 = W_1^T y_{1,2} - 2\hat{x}y_0 - y_{0,1} + y_{0,2}, \quad s_\ell = w_\ell - \left[Z_{\ell-1}b_{\ell-1} + 2\hat{X}_\ell y_{\ell-1} + y_{\ell-1,2}\right], \\
s_{k+1} = W_{k+1}^T y_{k+1,2} - \left(Z_k b_k + 2\hat{X}_{k+1}y_k + y_{k,2}\right) \qquad \text{for } k \in \{1, \dots, \ell - 2\}, \\
S_{1,1} = 2y_0 I, \quad S_{k,k+1} = -W_k^T Z_k, \quad S_{k+1,k+1} = 2(Z_k + \hat{X}_k) \qquad \text{for } k \in \{1, \dots, \ell - 1\},
\end{gathered}$$

where $Z_k = \operatorname{diag}(z_k)$ and $\hat{X}_k = \operatorname{diag}(\hat{x}_k)$ for all k.

A.3. Efficient formulation of `BM` for ℓ_∞ norm

We now describe how to implement `BM` for verifying ℓ_∞ adversaries. In the case of MNIST image, the ℓ_∞ norm ball constraint on the input x_1, i.e. $\|x_1 - \hat{x}\|_\infty \leq \rho$, can be combined with the valid input set constraints $0 \leq x_1 \leq 1$. Specifically, combining the two constraints yields: $\max\{0, \hat{x} - \rho\} \leq x_1 \leq \min\{1, \hat{x} + \rho\}$. Similar to the postactivation bound constraints in `BM-Full`, this constraint can be written as an elementwise ℓ_2 norm constraint as in

$$\max\{0, \hat{x} - \rho\} \leq x_1 \leq \min\{1, \hat{x} + \rho\} \quad \Longleftrightarrow \quad \|\mathbf{e}_i^T x_1 - \mathbf{e}_i^T \hat{x}_1\|^2 \leq \rho_1^2 \quad \text{for all } \ i \in \{1, \dots, n_1\}$$

where $\hat{x}_1 = \frac{1}{2}(\min\{1, \hat{x}+\rho\} + \max\{0, \hat{x}-\rho\})$ and $\rho_1 = \frac{1}{2}(\min\{1, \hat{x}+\rho\} - \max\{0, \hat{x}-\rho\})$. The above constraint yields the following Burer-Monteiro formulation

$$\operatorname{diag}\left[(u_1 - \hat{x}_1)(u_1 - \hat{x}_1)^T + V_1 V_1^T\right] \leq \rho_1^2$$

In turn, to verify neural networks train on MNIST with respect to ℓ_∞ perturbation, we solve (BM-r) in the following form

$$\begin{aligned}
\phi_r[c] = \min_{u,V} \quad & w_\ell^T u_\ell && \text{(BM-}\ell_\infty\text{)}\\
\text{s.t.} \quad & \operatorname{diag}\left[(u_1 - \hat{x}_1)(u_1 - \hat{x}_1)^T + V_1 V_1^T\right] \leq \rho_1^2, && (y_0)\\
& u_{k+1} \geq 0, \qquad u_{k+1} - W_k u_k - b_k \geq 0, && (y_{k,1}, y_{k,2})\\
& \operatorname{diag}\left[(u_{k+1} - W_k u_k - b_k)u_{k+1}^T + (V_{k+1} - W_k V_k)V_{k+1}^T\right] = 0, && (z_k)\\
& 1 + \sum_{k=1}^{\ell-1}(\|u_k\|^2 + \|V_k\|^2) \leq R^2, && (\mu)
\end{aligned}$$

for $k \in \{1, \ldots, \ell-1\}$. Notice that the ℓ_∞ norm constraint, i.e. $\|u_1 - \hat{x}\|_\infty \leq \rho$, has been combined with the valid input set constraint, i.e. $0 \leq u_1 \leq 1$. In Definition A.3, we summarized how to evaluate the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) using the primal and dual solution of (BM-ℓ_∞) in order to calculate the bound in Proposition 3.1.

Definition A.3. Let $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$, $z = (\{z_k\}_{k=1}^{\ell-1})$, $u = (u_1, \ldots, u_\ell)$ and $V = (V_1, \ldots, V_\ell)$ be any *certifiably first-order optimal* point of (BM-ℓ_∞). Each component in the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) can be evaluated as

$$\begin{gathered}
s_0 = -\sum_{k=1}^{\ell} s_k^T u_k, \quad z_0 = y_0^T(\hat{x}_1^2 - \rho_1^2) + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - \frac{1}{2}s_0,\\
s_1 = W_1^T y_{1,2} - 2Y_0\hat{x}, \quad s_\ell = w_\ell - [Z_{\ell-1}b_{\ell-1} + y_{\ell-1,1} + y_{\ell-1,2}],\\
s_{k+1} = W_{k+1}^T y_{k+1,2} - (Z_k b_k + y_{k,1} + y_{k,2}) \qquad \text{for } k \in \{1, \ldots, \ell-2\},\\
S_{1,1} = 2Y_0, \quad S_{k,k+1} = -W_k^T Z_k, \quad S_{k+1,k+1} = 2Z_k \qquad \text{for } k \in \{1, \ldots, \ell-1\},
\end{gathered}$$

where $Z_k = \operatorname{diag}(z_k)$ for all k. $Y_0 = \operatorname{diag}(y_0)$.

A.4. Efficient formulation of `BM-Full` for ℓ_∞ norm

Combine the results in A.2 and A.3, to verify ℓ_∞ adversaries via `BM-Full`, we solve (BM-r) in the following form

$$\begin{aligned}
\phi_r[c] = \min_{u,V} \quad & w_\ell^T u_\ell && \text{(BM-Full-}\ell_\infty\text{)}\\
\text{s.t.} \quad & \operatorname{diag}\left[(u_1 - \hat{x}_1)(u_1 - \hat{x}_1)^T + V_1 V_1^T\right] \leq \rho_1^2, && (y_0)\\
& \operatorname{diag}\left[(u_{k+1} - \hat{x}_{k+1})(u_{k+1} - \hat{x}_{k+1})^T + V_{k+1} V_{k+1}^T\right] \leq \rho_{k+1}^2 && (y_k)\\
& u_{k+1} - W_k u_k - b_k \geq 0, && (y_{k,2})\\
& \operatorname{diag}\left[(u_{k+1} - W_k u_k - b_k)u_{k+1}^T + (V_{k+1} - W_k V_k)V_{k+1}^T\right] = 0, && (z_k)\\
& 1 + \sum_{k=1}^{\ell-1}(\|u_k\|^2 + \|V_k\|^2) \leq R^2, && (\mu)
\end{aligned}$$

for $k \in \{1, \ldots, \ell-1\}$, where $\hat{x}_1$ and ρ_1 are defined in Appendix A.3. $\hat{x}_k$ and ρ_k for $k \in \{2, \ldots, \ell-1\}$ are defined in Appendix A.2. Notice that we also delete the constraints $u_k \geq 0$ because they overlap with the bound constraints $\max\{lb_k, 0\} \leq u_k$. In Definition A.4, we summarized how to evaluate the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD) using the primal and dual solution of (BM-Full-ℓ_∞) in order to calculate the bound in Proposition 3.1.

Definition A.4. Let $y = (y_0, \{y_k, y_{k,2}\}_{k=1}^{\ell-1})$, $z = (\{z_k\}_{k=1}^{\ell-1})$, $u = (u_1, \ldots, u_\ell)$ and $V = (V_1, \ldots, V_\ell)$ be any *certifiably first-order optimal* point of (BM-Full-ℓ_∞). Each component in the slack matrices $S(y, z)$ and the dual variable z_0 in (SDD)

can be evaluated as

$$s_0 = -\sum_{k=1}^{\ell} s_k^T u_k, \quad z_0 = \sum_{k=0}^{\ell-1} y_k^T(\hat{x}_{(k+1)}^2 - \rho_{(k+1)}^2) + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - \frac{1}{2}s_0,$$
$$s_1 = W_1^T y_{1,2} - 2Y_0\hat{x}, \quad s_\ell = w_\ell - \left[Z_{(\ell-1)}b_{(\ell-1)} + 2\hat{X}_\ell y_{\ell-1} + y_{(\ell-1),2}\right],$$
$$s_{k+1} = W_{k+1}^T y_{(k+1),2} - \left(Z_k b_k + 2\hat{X}_{k+1}y_k + y_{k,2}\right) \quad \text{for } k \in \{1,\dots,\ell-2\},$$
$$S_{1,1} = 2Y_0,\ S_{k,(k+1)} = -W_k^T Z_k,\ S_{(k+1),(k+1)} = 2(Z_k + \hat{X}_k) \quad \text{for } k \in \{1,\dots,\ell-1\},$$

where $Z_k = \mathrm{diag}(z_k)$ and $\hat{X}_k = \mathrm{diag}(\hat{x}_k)$ for all k. $Y_0 = \mathrm{diag}(y_0)$.

A.5. Efficient algorithm for solving `BM` and `BM-Full`

The efficient formulations described in Appendix A.1 to A.4 can be efficiently solved using a similar procedure described in Algorithm 1. In this section, we focus our attention on the practical and efficient algorithm for solving (BM-ℓ_2). We start by describing the initialization scheme for the primal variables u_k and V_k in (BM-ℓ_2), and then we summarize the efficient procedure for solving (BM-ℓ_2) in Algorithm 2. Notice that Algorithm 2 can be easily extended for (BM-Full-ℓ_2), (BM-ℓ_∞) and (BM-Full-ℓ_∞).

Initialize the primal variables. We initialize each u_k and V_k in (BM-ℓ_2) as close to their optimal as possible, which can be done as follows. First, apply `PGD` to estimate $x_1,\dots,x_\ell$ in the following semi-targeted attack problem (A-ℓ_2), which is the original semi-targeted attack problem (A) with the valid input set constraint

$$\phi[c] = \min_{x_1,\dots,x_\ell} \ w_\ell^T x_\ell \quad \text{s.t.} \quad x_{k+1} = \max\{0, W_k x_k + b_k\}, \quad 0 \le x_1 \le 1, \quad \|x_1 - \hat{x}\| \le \rho. \tag{A-ℓ_2}$$

Second, initialize each u_k to x_k; notice that since (BM-ℓ_2) is a nonconvex relaxation of (A-ℓ_2), x_k would usually be a good initialization for u_k. Finally, initialize each V_k to a random matrix that has small elements. The reason for applying small initialization to each V_k is to reduce the degree of constraint violation at the initial point.

Algorithm 2 Efficient algorithm for (BM-ℓ_2)

Input: Initial relaxation rank $r \ge 2$. Weights $W_1,\dots,W_\ell$ and biases $b_1,\dots,b_\ell$. Original input $\hat{x}$, true label $\hat{c}$, target label c, and perturbation size ρ. Variable radius bound R.
Output: Lower-bound $\phi_{\mathrm{lb}}[c] \le \phi[c]$ on the optimal value of the semi-targeted attack problem (A-ℓ_2).
Initialization: Initialize primal variable $x_+ = (\{x_k\}_{k=1}^{\ell}, \{\mathrm{vec}(M_k)\}_{k=1}^{\ell})$ where $x_1,\dots,x_\ell$ are estimated by solving (A-ℓ_2) via `PGD`, and each M_k is a random matrix of small elements that has the same shape as V_k. Initialize dual variables $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=0}^{\ell-1}) = 0$ and $z = (\{z_k\}_{k=1}^{\ell-1}) = 0$.
Algorithm:

1. (Solve rank-r relaxation) Warm-start the nonlinear solver with the initial point (x_+, y, z), and then use the solver to solve (BM-ℓ_2) over $x = (\{u_k\}_{k=1}^{\ell}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell})$. After the solver converges, retrieve the corresponding dual multipliers y and z. We choose `knitro` (Byrd et al., 2006) as the nonlinear solver in our experiment.

2. (Check certifiable first-order optimality) Let $f(x)$, $g(x)$, and $h(x)$ denote the objective, the inequality constraints associated with y, and the equality constraints associated with z in (BM-ℓ_2), respectively. If $\|\nabla f(x) + \nabla g(x)y + \nabla h(x)z\|$ is sufficiently small, and if $g(x) \le 0$, $h(x) = 0$ and $1 + \|x\| < R$ hold to sufficient tolerance, then continue. Otherwise, return error due to solver's inability to achieve certifiable first-order optimality.

3. (Check dual feasibility) Compute $S(y,z)$ and z_0 using the formula in Definition A.1. If $\epsilon_{\mathrm{feas}} = -\lambda_{\min}[S(y,z)]$ is sufficiently small, then return $\phi_{\mathrm{lb}}[c] = z_0 - \epsilon_{\mathrm{feas}} \cdot R^2$. Otherwise, continue.

4. (Escape lifted saddle point) Compute the eigenvector $\xi = (0, \xi_1, \dots, \xi_\ell)$ satisfying $\|\xi\| = 1$ and $\xi^T S(y,z)\xi = -\epsilon_{\mathrm{feas}}$. Set up new primal initial point $x_+ = (\{u_k\}_{k=1}^{\ell}, \{\mathrm{vec}(V_{+,k})\}_{k=1}^{\ell})$ where $V_{+,k} = [V_k, 0] + \epsilon \cdot [0, \xi_k]$. Increment $r \leftarrow r+1$ and repeat Step 1 with (x_+, y, z) as the initial point.

B. Additional experiment for ℓ_2 norm

Model architectures In this experiment, we consider three deepter feedforward ReLU networks trained on MNIST dataset: MLP-4×100, MLP-6×100 and MLP-9×100. All three networks are adversarially trained using (Madry et al., 2018) with ℓ_∞ radius equals to 0.1[5]. MLP-4×100 has 4 hidden layers of 100 neurons each. MLP-6×100 has 6 hidden layers of 100 neurons each. MLP-9×100 has 9 hidden layers of 100 neurons each.

B.1. Tightness plots for deeper neural networks

It is known that the relaxation gap generally increases along with the number of layers in the neural network. To demonstrate the performance of our proposed methods in deeper networks, in this experiment, we measure the relaxation gap of `BM-Full` and `BM` with respect to models of three different depths.

Results and discussions. Figure 3 plots the average bounds against ℓ_2 perturbation radius for three MNIST networks with 4, 6, 9 hidden layers of 100 neurons each. Notably, `BM` become loose for MLP-6×100 and become looser than `LP-Full` for MLP-9×100. This result is expected and is consistent with Zhang (2020); in particular, the SDP relaxation for ReLU gate, without any bound constrains on preactivations, does become loose for multiple layers. On the other hand, `BM-Full` remain significantly tighter than `LP-Full` for all cases. Furthermore, since the preactivation bounds used in `BM-Full` and `LP-Full` are the same in this experiment, Figure 2 and Figure 3 suggest that with the same quality of preactivation bounds, `BM-Full` would yield a tighter relaxation than `LP-Full`. Based on this finding, we note that the preactivation bounds for `BM-Full` can also be computed via nonconvex relaxation methods, which should yield a tighter bound on the preactivations and hence further reduces the relaxation gap for `BM-Full`; one example would be recursively apply `BM-Full` to compute the upper and lower bound on each neuron, however, such method can be extremely computational expensive for small to medium size networks. We leave `BM-Full` with better preactivation bounds to our future work.

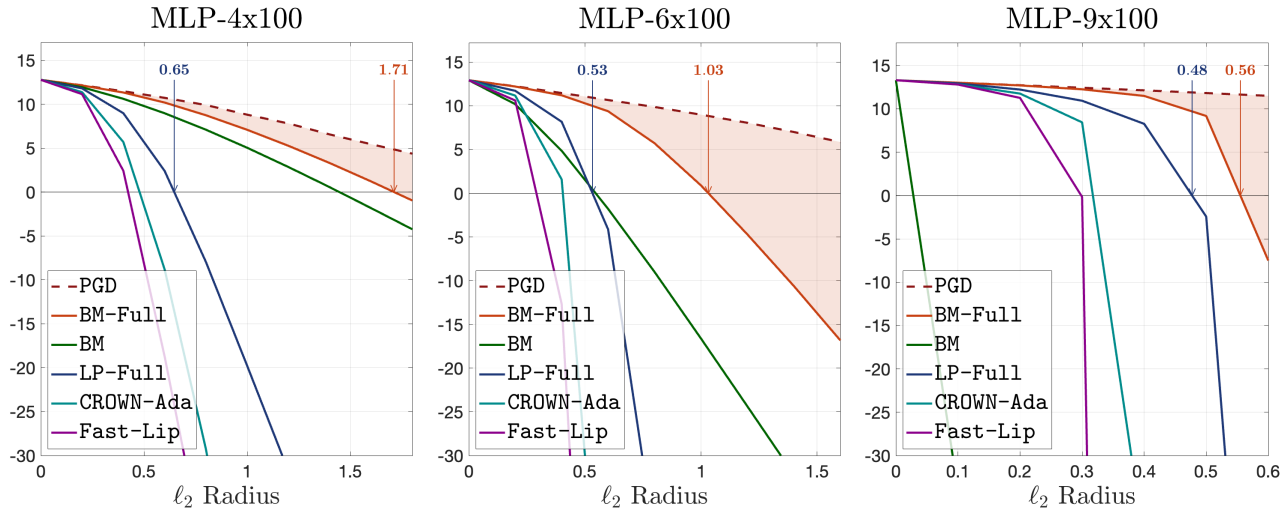

Figure 3: **Lower bound on (A) with different network depth (ℓ_2 norm).** We compute the average lower bound on (A) for three models with 4, 6, 9 hidden layers of 100 neurons each, respectively. The upper bound on robustness margin is estimated via `PGD`. Observe that the gap between the `PGD` upper bound and lower bound from `BM-Full` are significantly smaller than that from `LP-Full`. We note that without bound propagations, `BM` does get loose when the number of hidden layers is more than 6. **(Left.)** MLP-4×100. **(Middle.)** MLP-6×100. **(Right.)** MLP-9×100.

B.2. Visualizing adversarial attacks and robustness verification

To illustrate why robustness verification is important in image classification, in this experiment, we perform a case study based on an image in the test set using the model ADV-MNIST. In particular, we focus on showing how would the ℓ_2

[5]We train all three models using the code available at https://github.com/locuslab/convex_adversarial/blob/master/examples/mnist.py

adversaries look like in practice for four perturbation radius $\rho \in \{0.5, 1.0, 1.7, 2.0\}$, as well as their corresponding lower bound on (A) computed from our nonconvex and LP-based verifiers. We choose ℓ_2 norm over ℓ_∞ norm for this experiment because ℓ_2 norm allows perturbations to be concentrated on a small group of pixels, which produces adversaries that are more perceptually consistent to the original dataset when the perturbation radius is large.

Results and discussions. Figure 4 shows: the adversarial attacks targeting the second most probable class of a MNIST image of class "1"; the probability of the top-2 classes for the attacked image (calculated using the softmax function); the `PGD` upper bound; and the lower bounds from different verifiers. For the MNIST image shown in the figure, the second most probable class is "7", and its adversarial attacks are computed via `PGD`.

The first and second column of Figure 4 correspond to attacks within two small ℓ_2 perturbation radius $\rho \in \{0.5, 1.0\}$. We see that every verifiers is able to verify robustness for $\rho = 0.5$, however, only `BM-Full` and `BM` are able to verify robustness for $\rho = 1.0$, all the other verifiers fail because their corresponding lower bounds become loose. In both cases, the adversaries look really similar to the original image.

The third and fourth column of Figure 4 correspond to attack within two large ℓ_2 perturbation radius $\rho \in \{1.7, 2.0\}$. Notice that `BM-Full` and `BM` can still verify robustness for $\rho = 1.7$ even though the image is at the boundary of becoming not robust. In addition, the image is not robust for $\rho = 2.0$ as the `PGD` upper bound is negative. Notably, both images start gaining features of the digit "7".

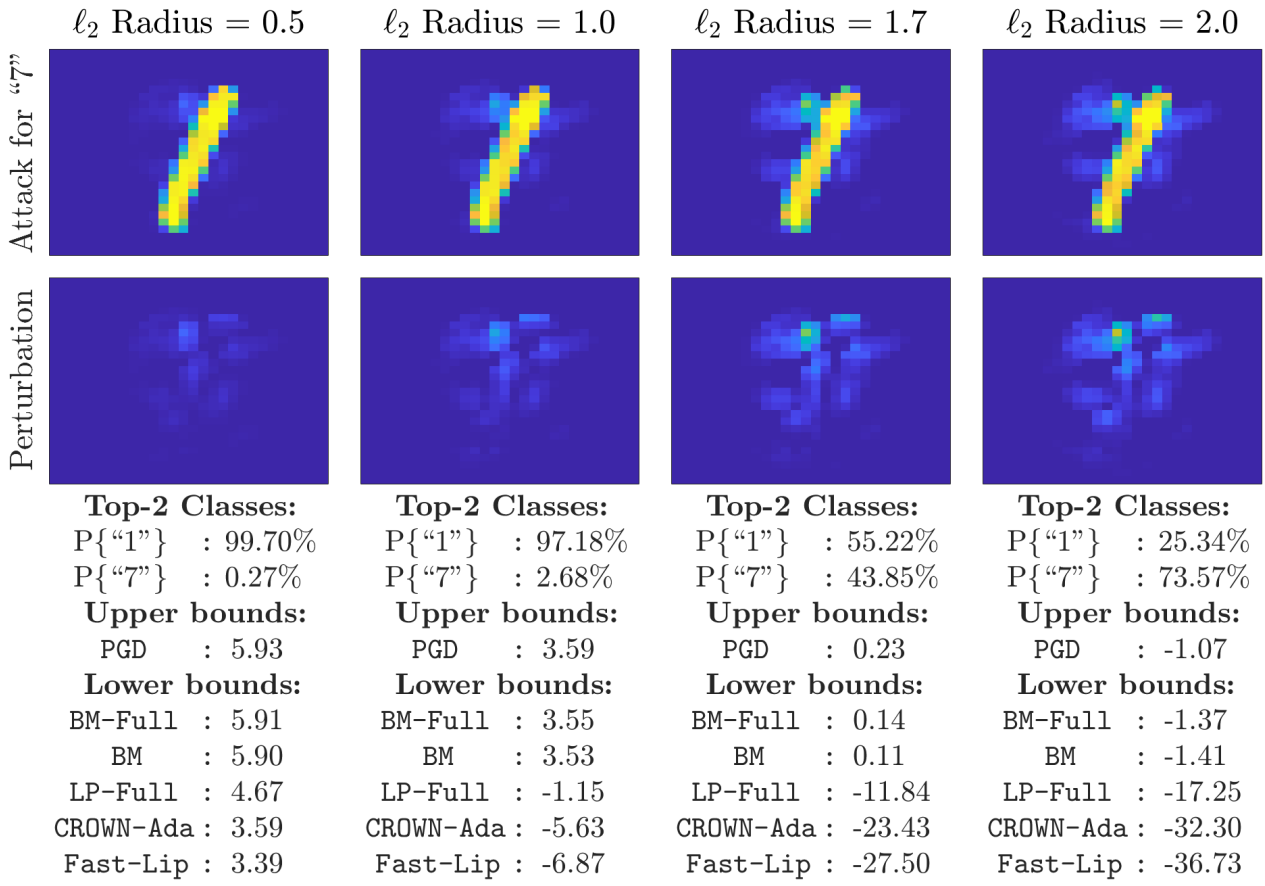

Figure 4: **Visualizing ℓ_2 adversarial examples for ADV-MNIST.** We take an image in the test set to compare the capability of different verifiers for certifying robustness within four different ℓ_2 radius $\rho \in \{0.5, 1.0, 1.7, 2.0\}$. Each column in the figure shows (left to right): **(Column 1.)** The image is robust for $\rho = 0.5$, and it can be certified by all verifiers. **(Column 2.)** The image is robust for $\rho = 1.0$, but it can only be certified by our verifiers, `BM-Full` and `BM`. **(Column 3.)** The image is closed to become not robust for $\rho = 1.7$, but it can still be certified by our verifiers. **(Column 4.)** The image is not robust for $\rho = 2.0$.

C. Additional experiment for ℓ_∞ norm

In this section, we apply BM and BM-Full to perform the same experiments in Section 5 and Appendix B.1 with respect to ℓ_∞ perturbation. Similar to the experimental results for ℓ_2 perturbation, we set the preactivation bounds in BM-Full to be the same as those in LP-Full. The neural network models used in this section are defined in Section 5 and Appendix B.

C.1. Robust verification for ℓ_∞ adversaries

We apply BM and BM-Full to verify robustness of the first 1000 correctly classified images. The simulation settings in this experiment are the same as those in Table 1.

Results and discussions. Table 2 shows the number of images certified as *robust* within the first 1000 correctly classified images. We see that BM-Full is able to consistently outperform LP-based verifiers in all cases. BM outperforms LP-Full in ADV-MNIST and NOR-MNIST but achieves similar performance in LPD-MNIST. This is because LPD-MNIST is robustly trained by maximizing dual lower bound of LP relaxation over a convex outer polytope (Wong & Kolter, 2018); therefore, LP-based verifiers tend to work well for models that are robustly trained using this method. However, we note that even though models that are trained using Wong & Kolter (2018) may be efficiently verified via LP-based verifiers, the robust training method of Wong & Kolter (2018) is generally more conservative than the method of Madry et al. (2018) as it is optimized over a convex outer polytope, which results in lower test accuracy in the final model. In this experiment, LPD-MNIST, the model that is trained using Wong & Kolter (2018), has test accuracy 95.91%, and ADV-MNIST, the model that is trained using Madry et al. (2018) has accuracy 96.67%. The performance of our verifiers, BM-Full and BM, is invariant to both robust training methods and achieve similar level of tightness with respect to all three model in Table 2. We provide a thorough analysis on the tightness of BM-Full and BM in the next experiment.

Table 2: **Robustness verification for neural networks.** We compare the number of images certified as *robust* by BM-Full, BM, α, β-CROWN, LP-Full, CROWN-Ada and Fast-Lip within the first 1000 images for normally and robustly trained networks. The upper bound (denoted as UB) on the true number of robust images is obtained by PGD.

Network	ℓ_∞ Radius	PGD	BM-Full		BM		α, β-CROWN		LP-Full		CROWN-Ada		Fast-Lip	
		UB	Robust	Time	Robust	Time	Robust	Time	Robust	Time	Robust	Time	Robust	Time
ADV-MNIST	0.10	831	**791**	87s	760	124s	791	2s	314	16s	8	10ms	4	13ms
ADV-MNIST	0.13	731	632	102s	574	144s	**673**	11s	46	17s	1	9ms	0	13ms
ADV-MNIST	0.15	626	484	127s	366	125s	**535**	22s	11	15s	0	9ms	0	14ms
LPD-MNIST	0.10	868	**855**	125s	828	163s	818	0.2s	829	13s	589	12ms	540	10ms
LPD-MNIST	0.13	791	**768**	104s	713	126s	743	0.2s	689	13s	154	10ms	120	11ms
LPD-MNIST	0.15	727	**672**	120s	597	132s	672	0.3s	545	12s	43	9ms	32	12ms
NOR-MNIST	0.02	910	**898**	99s	859	116s	881	1.4s	686	3s	130	9ms	88	12ms
NOR-MNIST	0.03	775	**729**	143s	617	107s	713	16s	278	4s	8	8ms	3	12ms
NOR-MNIST	0.05	401	**267**	229s	127	154s	238	43s	10	5s	0	12ms	0	17ms

C.2. Tightness plots

We apply BM-Full and BM to compute the average lower bound on (A) with respect to ℓ_∞ perturbation radius. The simulation settings in this experiment are the same as those in Figure 2.

Results and discussions. As demonstrated in Figure 5, both BM-Full and BM achieve tighter lower bound than the LP-based verifiers for all three models, across a wide range of ℓ_∞ perturbation radius. However, the gap between BM (green line) and LP-Full (blue line) is a lot smaller in LPD-MNIST compared to NOR-MNIST and ADV-MNIST, especially within the interval of 0.1 to 0.15. This explains why BM and LP-Full achieve similar performance for LPD-MNIST in Table 2.

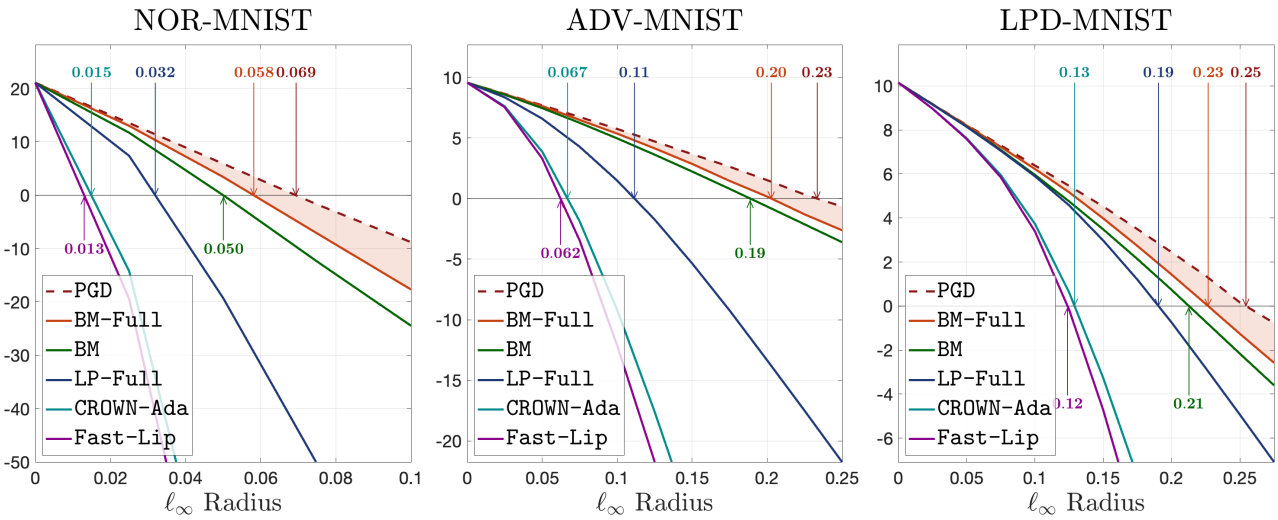

Figure 5: **Lower bounds on the robustness margin**. We take `BM-Full`, `BM`, `LP-Full`, `CROWN-Ada` and `Fast-Lip` to compute their average lower bound on (A), and then compare them to the average `PGD` upper bound on (A) over a wide range of ℓ_∞ perturbation radius. **(Left.)** NOR-MNIST. **(Middle.)** ADV-MNIST. **(Right.)** LPD-MNIST.

C.3. Tightness plots for deeper neural networks

We apply `BM-Full` and `BM` to compute the average lower bound on (A) for three robustly trained MNIST models of different depths. The simulation settings in this experiment are the same as those in Figure 3.

Results and discussions. We plot the average lower bound on (A) for three MNIST models of different depths. Similar to the results in Figure 3, we see that `BM` becomes loose when the network has more than 6 layers; this is expected as SDP relaxation, without any bound propagation, does become loose for multiple layers (Zhang, 2020). Though `BM` becomes loose in deeper networks, `BM-Full` remains significantly tighter than `LP-Full` in all cases. We again emphasize that the preactivation bounds used in `BM-Full` are the same as those in `LP-Full` in this experiment, and those bounds could be tighten by using the nonconvex relaxation techniques proposed in this paper. We defer it to our future work.

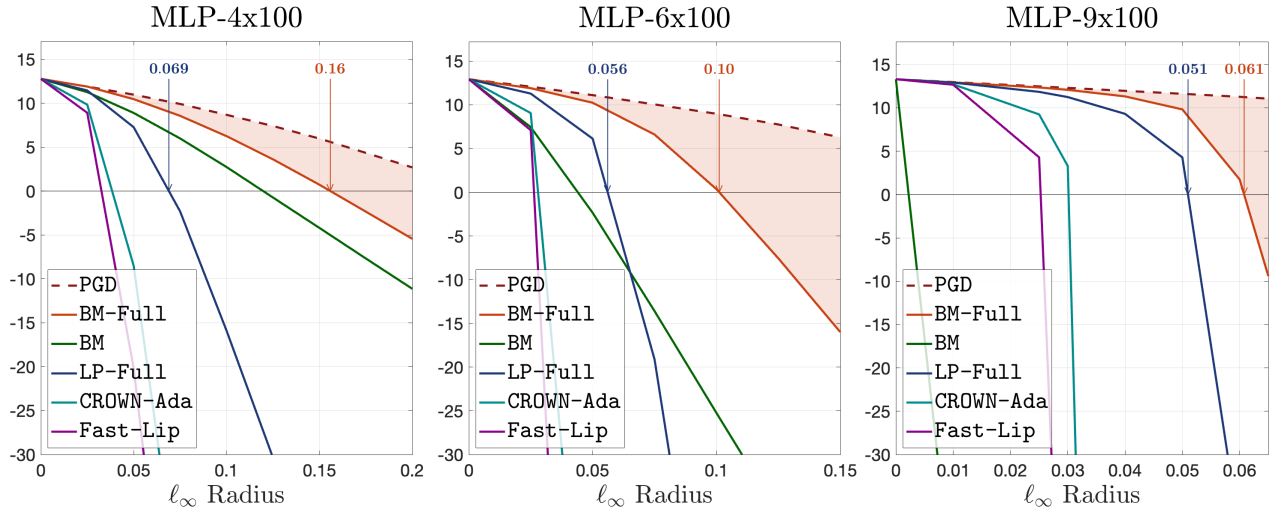

Figure 6: **Lower bound on (A) with different network depth (ℓ_∞ norm).** We compute the average lower bound on (A) for three models with 4, 6, 9 hidden layers of 100 neurons each, respectively. Observe that `BM-Full` is significantly tighter than `LP-Full` in all cases. We note that without bound propagations, `BM` does become loose when the number of hidden layers is more than 6. **(Left.)** MLP-4×100. **(Middle.)** MLP-6×100. **(Right.)** MLP-9×100.

D. Derivation of the dual problem (SDD)

Recall that primal problem (SDP-r) is written (with the corresponding Lagrange multipliers given in parantheses) as the following

$$\begin{aligned}
\phi_r[c] = \min_{X\in\mathbb{S}^{n+1}} \quad & w_\ell^T x_\ell + w_0 x_0 && \text{(SDP-}r\text{)}\\
\text{s.t.} \quad & \operatorname{tr}(X_{1,1}) - 2x_1^T\hat{x} + x_0\|\hat{x}_1\|^2 \le x_0\rho^2, \quad x_0 = 1 && (y_0, z_0)\\
& x_{k+1} \ge 0, \qquad x_{k+1} \ge W_k x_k + b_k x_0, && (y_{k,1}, y_{k,2})\\
& \operatorname{diag}(X_{k+1,k+1} - W_k X_{k,k+1} - b_k x_{k+1}^T) = 0 && (z_k)\\
& \operatorname{tr}(X) \le R^2, && (\mu)\\
& X = \left[\begin{array}{c|ccc} x_0 & x_1^T & \cdots & x_\ell^T \\ \hline x_1 & X_{1,1} & \cdots & X_{1,\ell} \\ \vdots & \vdots & \ddots & \vdots \\ x_\ell & X_{1,\ell}^T & \cdots & X_{\ell,\ell} \end{array}\right] \succeq 0, \quad \operatorname{rank}(X) \le r. && (S)
\end{aligned}$$

for all $k \in \{1, \ldots, \ell-1\}$. This problem can be viewed as a generic instance of the following primal problem

$$\min_{X\succeq 0,\ \operatorname{rank}(X)\le r} \langle F, X\rangle \quad \text{s.t.} \quad \mathcal{G}_0(X) \le 0, \quad \mathcal{G}_k(X) \le 0, \quad \mathcal{H}_0(X) + 1 = 0, \quad \mathcal{H}_k(X) = 0, \quad \operatorname{tr}(X) \le R^2$$

for all $k \in \{1, \ldots, \ell-1\}$. Here, we implicitly define F to satisfy $\langle F, X\rangle = w_\ell^T x_\ell + w_0 x_0$ and the linear constraint operators are respectively

$$\mathcal{G}_0(X) = \operatorname{tr}(X_{1,1}) - 2x_1^T\hat{x} + (\|\hat{x}\|^2 - \rho^2)x_0, \quad \mathcal{G}_k(X) = \begin{bmatrix} -x_{k+1} \\ W_k x_k + b_k x_0 - x_{k+1} \end{bmatrix},$$
$$\mathcal{H}_0(X) = -x_0, \quad \mathcal{H}_k(X) = \operatorname{diag}(X_{k+1,k+1} - W_k X_{k,k+1} - b_k x_{k+1}^T).$$

Notice that S is the dual variable associated with constraint $X \succeq 0$ that satisfies $S \succeq 0$ and $\operatorname{rank}(X) + \operatorname{rank}(S) \le n+1$ at optimality. Setting $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1}) > 0$, $z = (\{z_k\}_{k=1}^{\ell-1})$ and $\mu \le 0$, the Lagrangian of (SDP-r) reads

$$\begin{aligned}
\mathscr{L}(X, y, z, \mu, S) &= \langle F, X\rangle + \langle z_0, \mathcal{H}_0(X) + 1\rangle + \langle y_0, \mathcal{G}_0(X)\rangle + \sum_{k=1}^{\ell-1}\left[\left\langle \begin{bmatrix} y_{k,1} \\ y_{k,2}\end{bmatrix}, \mathcal{G}_k(X)\right\rangle + \langle z_k, \mathcal{H}_k(X)\rangle\right]\\
&\quad - \mu(\operatorname{tr} X - R^2) - \langle S, X\rangle\\
&= z_0 + R^2\mu + \left\langle F + \mathcal{G}_0^T(y_0) + \mathcal{H}_0^T(z_0) + \sum_{k=1}^{\ell-1}\left[\mathcal{G}_k^T(y_{k,1}, y_{k,2}) + \mathcal{H}_k^T(z_k)\right] - S - \mu I, X\right\rangle\\
&= z_0 + R^2\mu + \langle S(x,y) - S - \mu I, X\rangle
\end{aligned}$$

where we use the superscript "T" to indicate the adjoint operators. Setting $S = S(x,y) - \mu I \succeq 0$ yields the dual problem

$$\max_{y\ge 0,\ z,\ \mu\le 0} z_0 + R^2\mu \quad \text{s.t.} \quad S(y,z) \succeq \mu I. \tag{SDD}$$

Finally, we derive an explicit expression for the slack matrix

$$S(y,z) \equiv \frac{1}{2}\left[\begin{array}{c|cccc} s_0 & s_1^T & s_2^T & \cdots & s_\ell^T \\ \hline s_1 & S_{1,1} & S_{1,2} & & \\ s_2 & S_{1,2}^T & S_{2,2} & \ddots & \\ \vdots & & \ddots & \ddots & S_{\ell-1,\ell} \\ s_\ell & & & S_{\ell-1,\ell}^T & S_{\ell,\ell} \end{array}\right] = F + \mathcal{G}_0^T(y_0) + \mathcal{H}_0^T(z_0) + \sum_{k=1}^{\ell-1}\left[\mathcal{G}_k^T(y_{k,1}, y_{k,2}) + \mathcal{H}_k^T(z_k)\right].$$

Here, we write out the adjoint operators in terms of the scalar, vector, and matrix components of X:

$$\begin{aligned}
\langle X, F\rangle &= x_0 \cdot w_0 &&+ \langle x_\ell, w_\ell\rangle \\
\left\langle X, \mathcal{H}_0^T(z_0)\right\rangle &= x_0 \cdot (-z_0) \\
\left\langle X, \mathcal{G}_0^T(y_0)\right\rangle &= x_0 \cdot y_0(\|\hat{x}\|^2 - \rho^2) &&+ \langle x_1, -2y_0\hat{x}\rangle && + \langle X_{1,1}, y_0 I\rangle \\
\left\langle X, \mathcal{G}_k^T(y_{k,1}, y_{k,2})\right\rangle &= x_0 \cdot y_{k,2}^T b_k &&+ \left\langle \begin{bmatrix} x_k \\ x_{k+1} \end{bmatrix}, \begin{bmatrix} W_k^T y_{k,2} \\ -(y_{k,1} + y_{k,2}) \end{bmatrix} \right\rangle \\
\left\langle X, \mathcal{H}_k^T(z_k)\right\rangle &= &&+ \langle x_{k+1}, -Z_k b_k\rangle && + \left\langle \begin{bmatrix} X_{k,(k+1)} \\ X_{(k+1),(k+1)} \end{bmatrix}, \begin{bmatrix} -W_k^T Z_k \\ Z_k \end{bmatrix} \right\rangle
\end{aligned}$$

where $Z_k = \mathrm{diag}(z_k)$. Isolating the scalar terms x_0, we obtain the expression for s_0

$$s_0 = 2\left[w_0 + y_0(\|\hat{x}\|^2 - \rho^2) + \sum_{k=1}^{\ell-1} b_k^T y_{k,2} - z_0\right]$$

Isolating the vector terms $x_1, x_2, \ldots, x_\ell$, we see that the following indeed holds for $s_1, \ldots, s_\ell$

$$\begin{aligned}
s_1 = W_1^T y_{1,2} - 2\hat{x}y_0, \quad s_\ell = w_\ell - \left[Z_{(\ell-1)} b_{(\ell-1)} + y_{(\ell-1),1} + y_{(\ell-1),2}\right], \\
s_{k+1} = W_{k+1}^T y_{(k+1),2} - (Z_k b_k + y_{k,1} + y_{k,2}) \quad \text{for } k \in \{1, \ldots, \ell-2\}.
\end{aligned}$$

Finally, isolating the matrix terms $X_{i,j}$ for $i, j \in \{1, \ldots, \ell\}$, we have

$$S_{1,1} = 2y_0 I, \quad S_{k,k+1} = -W_k^T Z_k, \quad S_{k+1,k+1} = 2Z_k \quad \text{for } k \in \{1, \ldots, \ell-1\}.$$

E. Proof of the Main Results

E.1. Proof of Proposition 3.1: the dual lower bounds for (SDP-r)

Recall from the previous section, we have rewritten (SDP-r) into the following generic form

$$\min_{X \succeq 0,\ \mathrm{rank}(X) \le r} \langle F, X\rangle \quad \text{s.t.} \quad \mathcal{G}_0(X) \le 0, \quad \mathcal{G}_k(X) \le 0, \quad \mathcal{H}_0(X) + 1 = 0, \quad \mathcal{H}_k(X) = 0, \quad \mathrm{tr}(X) \le R^2$$

Now we are ready to prove Proposition 3.1.

Proof. Let $X^\star$ denote the global solution of (SDP-r) with rank $r \ge 1$ and $\mathrm{tr}(X) \le R^2$. Then, for any dual multipliers $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$ and $z = (z_0, \{z_k\}_{k=1}^{\ell-1})$ that satisfy $y \ge 0$, we have

$$\begin{aligned}
\phi[c] \ge \phi_r[c] &= \langle F, X^\star\rangle \\
&= \left\langle S(x,y) - \mathcal{G}_0^T(y_0) - \mathcal{H}_0^T(z_0) - \sum_{k=1}^{\ell-1}\left[\mathcal{G}_k^T(y_{k,1}, y_{k,2}) + \mathcal{H}_k^T(z_k)\right], X^\star \right\rangle \\
&= \langle S(x,y), X^\star\rangle - \underbrace{\left\langle \begin{bmatrix} \mathcal{G}_0(X^\star) \\ \vdots \\ \mathcal{G}_{\ell-1}(X^\star) \end{bmatrix}, y \right\rangle}_{\le 0} - \underbrace{\left\langle \begin{bmatrix} \mathcal{H}_0(X^\star) \\ \vdots \\ \mathcal{H}_{\ell-1}(X^\star) \end{bmatrix}, z \right\rangle}_{= -z_0} \\
&\ge z_0 + \langle S(x,y), X^\star\rangle \\
&\ge z_0 + R^2 \cdot \min\{0, \lambda_{\min}[S(y,z)]\}.
\end{aligned}$$

□

E.2. Proof of the dual lower bound for (BM-r)

we start by putting (BM-r) into the standard-form of nonlinear program (NLP). In the previous section, we have rewritten the original SDP problem (SDP-r) in the following generic form

$$\min_{X\succeq 0,\ \mathrm{rank}(X)\le r} \langle F, X\rangle \quad \text{s.t.} \quad \mathcal{G}_0(X)\le 0, \quad \mathcal{G}_k(X)\le 0, \quad \mathcal{H}_0(X)+1=0, \quad \mathcal{H}_k(X)=0, \quad \mathrm{tr}(X)\le R^2$$

Since every $(n+1)\times(n+1)$ positive semidefinite matrix X of rank at most r admits an $(n+1)\times n+1$ lower-triangular Cholesky factorization U that satisfies $X=UU^T$. Substituting

$$X=\begin{bmatrix} u_0^2 & u_0\cdot u^T \\ u_0\cdot u & uu^T+VV^T\end{bmatrix} = \left[\begin{array}{c|c} u_0 & 0 \\ \hline u_1 & V_1 \\ \vdots & \vdots \\ u_\ell & V_\ell \end{array}\right]\left[\begin{array}{c|c} u_0 & 0 \\ \hline u_1 & V_1 \\ \vdots & \vdots \\ u_\ell & V_\ell \end{array}\right]^T = UU^T,$$

as in (1), we obtain the generic form of our proposed Burer–Monteiro formulation (BM-r)

$$\min_{\|U\|\le R} \left\langle F, UU^T\right\rangle \quad \text{s.t.} \quad \mathcal{G}_0(UU^T)\le 0, \quad \mathcal{G}_k(UU^T)\le 0, \quad \mathcal{H}_0(UU^T)+1=0, \quad \mathcal{H}_k(UU^T)=0.$$

To solve (BM-r) as an instance of the standard-form nonlinear program (NLP). Define

$$x\equiv\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_\ell \\ \mathrm{vec}(V_1) \\ \vdots \\ \mathrm{vec}(V_\ell)\end{bmatrix}, \quad f(x)\equiv\left\langle F, UU^T\right\rangle, \quad g(x)\equiv\underbrace{\begin{bmatrix}\mathcal{G}_0(UU^T) \\ \mathcal{G}_1(UU^T) \\ \vdots \\ \mathcal{G}_{\ell-1}(UU^T)\end{bmatrix}}_{\mathcal{G}(UU^T)}, \quad h(x)\equiv\underbrace{\begin{bmatrix}\mathcal{H}_0(UU^T) \\ \mathcal{H}_1(UU^T) \\ \vdots \\ \mathcal{H}_{\ell-1}(UU^T)\end{bmatrix}}_{\mathcal{H}(UU^T)}+\underbrace{\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}}_{h_0}$$

we obtain the standard-form of nonlinear program (NLP)

$$\min_{\|x\|\le R} f(x) \quad \text{s.t.} \quad g(x)\le 0, \quad h(x)=0.$$

Similar to the previous section, let $y\ge 0$, z and $\mu\le 0$ denote the Lagrangian multiplier associated with $g(x)\le 0$, $h(x)=0$ and $\|x\|^2\le R^2$, respectively. The corresponding Lagrangian function reads

$$\mathscr{L}(x,y,z,\mu)=f(x)+y^Tg(x)+z^Th(x)-\mu(\|x\|^2-R^2)=z_0+\mu R^2+\left\langle S(x,y)-\mu I, U(x)U(x)^T\right\rangle \tag{2}$$

where $S(y,z)=F+\mathcal{G}^T(y)+\mathcal{H}^T(z)$ and $U(x)$ is a matricization operator

$$U(x)\equiv U(u_0,u,\mathrm{vec}(V))=\begin{bmatrix} u_0 & 0 \\ u & V\end{bmatrix}.$$

General-purpose nonlinear programming solvers work by computing a feasible primal point x and dual multipliers y,z,μ that satisfy the *first-order optimality* condition (known as the Karush–Kuhn–Tucker (KKT) conditions)

$$\nabla_x\mathscr{L}(x,y,z,\mu)=0, \quad y\ge 0, \quad y\odot g(x)=0, \quad \mu\le 0, \quad \mu\cdot(\|x\|^2-R^2)=0, \tag{FOC}$$

and also attempt to achieve the *second-order optimality* condition (known as the projected Hessian condition)

$$\dot{x}^T\nabla^2_{xx}\mathscr{L}(x,y,z,\mu)\dot{x}\ge 0 \quad \text{for all } \dot{x}\in\mathcal{C}(x,y) \tag{SOC}$$

in which the *critical cone* is defined

$$\mathcal{C}(x,y) = \left\{ \dot{x} : \begin{array}{rcl} \nabla g_i(x)^T \dot{x} & \geq 0 & \text{for all } i \text{ with } g_i(x) = 0, \\ \nabla g_i(x)^T \dot{x} & = 0 & \text{for all } i \text{ with } y_i > 0, \\ \nabla h_i(x)^T \dot{x} & = 0 & \text{for all } j, \\ 2x^T \dot{x} & \geq 0 & \text{if } \|x\|^2 = R^2, \\ 2x^T \dot{x} & = 0 & \text{if } \mu < 0. \end{array} \right\} \tag{3}$$

However, in the constrained optimization setting, a local minimum $x^\star$ does not need to satisfy (FOC) and (SOC)[6]. Solvers tend to fail catastrophically when converging towards a point that does not satisfy (FOC) and (SOC), either by diverging to infinity or cycling through nonsensical solutions. Instead, a notion of *constraint qualification* is required to ensure convergence. Of all possibilities, the LICQ is one of the stronger conditions that allow strong guarantees to be made.

Definition E.1 (LICQ)**.** We say that a given x satisfies the *linear independence constraint qualification* (LICQ) if the following holds

$$\nabla g(x)y + \nabla h(x)z + 2\mu \cdot x = 0, \quad g(x) \odot y = 0, \quad \mu \cdot (\|x\|^2 - R^2) = 0 \quad \iff \quad y = 0, \quad z = 0, \quad \mu = 0. \tag{LICQ}$$

One of our main theoretical contribution in this paper is to state the conditions for a point x to satisfy (LICQ).

Lemma E.2 (LICQ for (BM-r))**.** *If x satisfy the nonzero preactivation condition (NPCQ). Then, x satisfies (LICQ).*

We defer the proof of Lemma E.2 to the Appendix F. Lemma E.2 implies that for every local minimum $x^\star$, there exists a unique set of dual variables $(y^\star, z^\star, \mu^\star)$ such that $(x^\star, y^\star, z^\star \mu^\star)$ is guaranteed to satisfy (FOC) and (SOC); therefore, the nonlinear programming solvers are guaranteed to work.

Corollary E.3 (Dual lower bound of (BM-r))**.** *Let $x^\star = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell-1})$ denote a local minimum for the Burer–Monteiro problem (BM-r) that satisfy nonzero activation (NPCQ). Then, there exists an unique dual multipliers $y = (y_0, \{y_{k,1}, y_{k,2}\}_{k=1}^{\ell-1})$ and $z = (z_0, \{z_k\}_{k=1}^{\ell-1})$ that satisfy $y \geq 0$ provide the following lower-bound*

$$\phi[c] \geq \phi_r[c] \geq z_0 + R^2 \cdot \min\{0, \lambda_{\min}[S(y,z)]\}.$$

Proof. If $x^\star$ is a local minimum for (BM-r) that satisfy nonzero activation (NPCQ). Then, it follows from Lemma E.2 that the point x is first-order optimal, and the corresponding KKT equations (FOC) yields a unique set of dual multipliers (y, z, μ) that certify the above lower-bound on the global minimum. □

E.3. Proof for Theorem 4.4: Escape lifted saddle point.

Now, let us explain how LICQ leads to our desired results in Theorem 4.4. We start by proving two technique lemmas regarding the first- and second-optimality of (BM-r).

Lemma E.4 (First-order optimality)**.** *Let $x = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell-1})$ and y, z, μ satisfy $\nabla_x \mathscr{L}(x, y, z, \mu) = 0$. Then, the slack matrix $S(y,z) = F + \mathcal{G}^T(y) + \mathcal{H}^T(z)$ satisfies the following:*

- $S(y,z)U(x) = 0$.
- $S(y,z) - \mu I = \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix} (S_2 - \mu I) \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix}^T$ *for some matrix S_2.*

Proof. Let $S(y,z) = \begin{bmatrix} s_0 & s_1^T \\ s_1 & S_2 \end{bmatrix}$, the Lagrangian (2) can be written as the following

$$\begin{aligned} \mathscr{L}(x,y,z,\mu) &= z_0 + \mu R^2 + \left\langle \begin{bmatrix} u_0 & 0 \\ u & V \end{bmatrix} \begin{bmatrix} u_0 & 0 \\ u & V \end{bmatrix}^T, \begin{bmatrix} s_0 - \mu & s_1^T \\ s_1 & S_2 - \mu I \end{bmatrix} \right\rangle \\ &= z_0 + \mu(R^2 - u_0^2) + u_0^2 s_0 + 2u_0 s_1^T u + \left\langle uu^T + VV^T, S_2 - \mu I \right\rangle. \end{aligned}$$

[6]For an explicit counterexample, consider $f(x) = x$ and $g(x) = x^2$ and $h(x) = 0$.

The condition $\nabla_x \mathscr{L}(x, y, z, \mu) = 0$ is equivalent to setting the following three Jacobians to zero:

$$\frac{\partial \mathscr{L}}{\partial u_0} = 2(u_0(s_0 - \mu) + s_1^T u), \quad \frac{\partial \mathscr{L}}{\partial u} = 2(u_0 s_1^T + u^T(S_2 - \mu I)), \quad \frac{\partial \mathscr{L}}{\partial V} = 2V^T(S_2 - \mu I).$$

It follows by substituting the above into the equation below that

$$S(y, z)U(x) = \begin{bmatrix} s_0 - \mu & s_1^T \\ s_1 & S_2 - \mu I \end{bmatrix} \begin{bmatrix} u_0 & 0 \\ u & V \end{bmatrix} = \begin{bmatrix} u_0(s_0 - \mu) + s_1^T u & s_1^T V \\ s_1 u_0 + (S_2 - \mu I)u & (S_2 - \mu I)V \end{bmatrix} = 0,$$

where $s_1^T V = 0$ because $s_1^T = -u^T(S_2 - \mu I)/u_0$ and $(S_2 - \mu I)V = 0$.

Similarly, substituting $s_0 = -s_1^T u/u_0 + \mu$ and $s_1 = -(S_2 - \mu I)u/u_0$ yields

$$S(x, y) = \begin{bmatrix} s_0 & s_1^T \\ s_1 & S_2 \end{bmatrix} = \begin{bmatrix} u^T(S_2 - \mu I)u/u_0^2 + \mu & -u^T(S_2 - \mu I)/u_0 \\ -(S_2 - \mu I)u/u_0 & S_2 \end{bmatrix} = \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix} (S_2 - \mu I) \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix}^T + \mu I.$$

□

Lemma E.5 (Rank-deficient second-order optimality)**.** *Given y, z, let $x = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell-1})$ satisfy $\nabla_x \mathscr{L}(x, y, z, \mu) = 0$. If there exists unit vectors $\psi \in \mathbb{R}^r, \|\psi\| = 1$ and $(\xi_0, \xi_1) \in \mathbb{R} \times \mathbb{R}^n, \|(\xi_0, \xi_1)\| = 1$ such that*

$$V\psi = 0, \qquad 2 \begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix}^T (S(y, z) - \mu I) \begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix} = -\gamma < 0$$

where $S(y, z) = F + \mathcal{G}^T(y) + \mathcal{H}^T(z)$, then the vector $\dot{x} = (0, 0_n, \{\mathrm{vec}(\dot{V}_k)\}_{k=1}^{\ell-1})$ with $\dot{V}_k = [(u_k/u_0)\xi_0 - \xi_1]\psi^T$ satisfies

$$\nabla g(x)^T \dot{x} = 0, \qquad \nabla h(x)^T \dot{x} = 0, \qquad \dot{x}^T \nabla_{xx}^2 \mathscr{L}(x, y, z)\dot{x} = -\gamma.$$

Proof. Note that in general, a function of the form $f(x) = \langle F, UU^T \rangle$ with $U(x) = \begin{bmatrix} u_0 & 0 \\ u & V \end{bmatrix}$ has directional derivatives

$$\nabla f(x)^T \dot{x} = \left\langle F, U(x)U(\dot{x})^T + U(\dot{x})U(x)^T \right\rangle, \quad \dot{x}^T \nabla^2 f(x)\dot{x} = 2 \left\langle F, U(\dot{x})U(\dot{x})^T \right\rangle.$$

For our specific choice of $\dot{x}$, we can verify that

$$U(x)U(\dot{x})^T = \begin{bmatrix} u_0 & 0 \\ u & V \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & [(u/u_0)\xi_0 - \xi_1]\psi^T \end{bmatrix}^T = \begin{bmatrix} 0 & 0 \\ 0 & V\psi[(u/u_0)\xi_0 - \xi_1]^T \end{bmatrix} = 0.$$

Since $g_i(x) = \langle G_i, UU^T \rangle$ and $h_j(x) = h_{0,j} + \langle H_j, UU^T \rangle$ for some matrix G_i, H_j and scalar $h_{0,j}$, it then follows that $\nabla g_i(x)^T \dot{x} = 0$ and $\nabla h_j(x)^T \dot{x} = 0$ for all i and j.

Next, given that the Lagrangian (2) is written

$$\mathscr{L}(x, y, z, \mu) = \langle h_0, z \rangle + \mu R^2 + \left\langle S(x, y) - \mu I, U(x)U(x)^T \right\rangle,$$

For our specific choice of $\dot{x}$, the second-order directional derivative reads

$$\dot{x}^T \nabla_{xx}^2 \mathscr{L}(x, y, z, \mu)\dot{x} = 2 \left\langle S(x, y) - \mu I, U(\dot{x})U(\dot{x})^T \right\rangle = 2 \begin{bmatrix} 0 \\ \xi_2 \end{bmatrix}^T (S(x, y) - \mu I) \begin{bmatrix} 0 \\ \xi_2 \end{bmatrix}.$$

where $\xi_2 = (u/u_0)\xi_0 - \xi_1$. It follows from Lemma E.4 that for $\nabla_x \mathscr{L}(x, y, z, \mu) = 0$, the slack matrix satisfies $S(x, y) - \mu I = \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix} (S_2 - \mu I) \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix}^T$ and therefore

$$\begin{aligned} 2\begin{bmatrix} 0 \\ \xi_2 \end{bmatrix}^T (S(x,y) - \mu I) \begin{bmatrix} 0 \\ \xi_2 \end{bmatrix} &= 2\begin{bmatrix} 0 \\ \xi_2 \end{bmatrix}^T \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix} (S_2 - \mu I) \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix}^T \begin{bmatrix} 0 \\ \xi_2 \end{bmatrix} \\ &= 2\begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix}^T \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix} (S_2 - \mu I) \begin{bmatrix} u^T/u_0 \\ -I \end{bmatrix}^T \begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix} \\ &= 2\begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix}^T (S(x,y) - \mu I) \begin{bmatrix} \xi_0 \\ \xi_1 \end{bmatrix} \\ &= -\gamma. \end{aligned}$$

□

Lemma E.6 (Critical cone)**.** *Let Ω denote a set of feasible points of (BM-r) where (LICQ) hold. If $x \in \Omega$, and (y, z, μ) satisfy (FOC), then there exists a continuously differentiable path $x(t)$ with initial position $x(0) = x$ that satisfies*

$$f(x(t)) = \mathscr{L}(x(t), y, z), \quad x(t) \in \Omega \quad \textit{for all } t \in [0, \epsilon)$$

if and only if $\dot{x}(0) \in \mathcal{C}(x, y)$. Moreover, if $g(x)$ and $h(x)$ are k-times continuously differentiable, then $x(t)$ is also k-times continuously differentiable.

We are now ready to prove the escape result.

Proof. Let x be first-order optimal for (BM-r) with dual multipliers (y, z, μ). If x satisfies nonzero activation, and $\gamma = -\lambda_{\min}[S(y, z) - \mu I] > 0$, then the eigenvector $\xi = (\xi_0, \xi_1)$ that satisfies $\xi^T(S(y, z) - \mu I)\xi = -\gamma\|\xi\|^2$ implicitly defines an escape path

$$u_{k,+}(t) = u_k + O(t^2), \qquad V_{k,+}(t) = [V_k, 0] + t \cdot [0, (u_k/u_0)\xi_0 - \xi_k] + O(t^2)$$

so that $x_+(t) = (u_0, u(t), V(t))$ is feasible with sufficiently small $t \geq 0$, and for which the objective makes a decrement as follows

$$w_\ell^T u_{\ell,+}(t) = w_\ell^T u_\ell - 2t^2\gamma + O(t^3) \text{ for all } t \in [0, \epsilon).$$

□

F. Proof of Constraint Qualification

In this section, we provide the proof for Lemma E.2. Specifically, we show that the (LICQ) holds for (BM-r) under the assumption of nonzero preactivation (NPCQ). Throughout this section, we assume R is chosen large enough such that $\|x\| > R$ holds at optimality with a *strict inequality*, which in turn implies the corresponding dual variable $\mu = 0$.

We start by showing that (LICQ) holds for single neuron and single layer ReLU networks.

Lemma F.1 (Single neuron)**.** *Define $\alpha_0, \alpha, \overline{\alpha} \in \mathbb{R}$ and $\beta, \overline{\beta}, \in \mathbb{R}^{r-1}$. Let $x = (\alpha_0, \alpha, \overline{\alpha}, \beta, \overline{\beta})$ satisfy $g_1(x) \geq 0$, $g_2(x) \geq 0$ and $h_1(x) = 0$, where*

$$g_1(x) = \alpha_0\overline{\alpha}, \quad g_2(x) = \alpha_0(\overline{\alpha} - \alpha - \gamma\alpha_0), \quad h_1(x) = \overline{\alpha}(\overline{\alpha} - \alpha - \gamma\alpha_0) + \left\langle \overline{\beta}, \overline{\beta} - \beta \right\rangle.$$

Suppose that $\alpha_0^2 \neq 1$, $\alpha + \gamma\alpha_0 \neq 0$ and $\beta \neq 0$. Then, the following holds

$$\nabla g_1(x) y_1 + \nabla g_2(x) y_2 + \nabla h_1(x) z_1 = 0, \tag{4a}$$

$$g_1(x) \odot y_1 = g_2(x) \odot y_2 = 0, \tag{4b}$$

if and only if $y_1 = y_2 = z_1 = 0$.

Proof. Explicitly computing the gradient terms in (4a), we can write (4a) and (4b) as the following

$$\begin{bmatrix} \overline{\alpha} & \overline{\alpha}-\alpha-2\gamma\alpha_0 & -\overline{\alpha}\gamma \\ 0 & -\alpha_0 & -\overline{\alpha} \\ \alpha_0 & \alpha_0 & 2\overline{\alpha}-\alpha-\gamma\alpha_0 \\ 0 & 0 & -\overline{\beta} \\ 0 & 0 & 2\overline{\beta}-\beta \\ \alpha_0\overline{\alpha} & 0 & 0 \\ 0 & \alpha_0(\overline{\alpha}-\alpha-\gamma\alpha_0) & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0$$

The goal is to show that the above matrix has full column rank. To simplify our proof, we delete the first, the second and the fourth row as these rows are obviously dependent to the the rest of the rows. Deleting those three rows reveals the desired claim as equivalent to the following

$$\begin{bmatrix} \alpha_0 & \alpha_0 & 2\overline{\alpha}-\alpha-\gamma\alpha_0 \\ 0 & 0 & 2\overline{\beta}-\beta \\ \alpha_0\overline{\alpha} & 0 & 0 \\ 0 & \alpha_0(\overline{\alpha}-\alpha-\gamma\alpha_0) & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0. \tag{5}$$

Next, completing the square $h_1(x) = \overline{\alpha}(\overline{\alpha}-\alpha-\gamma\alpha_0) + \left\langle \overline{\beta}, \overline{\beta}-\beta \right\rangle = \|(\overline{\alpha},\overline{\beta}) - \frac{1}{2}(\alpha+\gamma\alpha_0,\beta)\|^2 - \|\frac{1}{2}(\alpha+\gamma\alpha_0,\beta)\|^2$ reveals that

$$h_1(x) = 0 \quad \Longrightarrow \quad \|(2\overline{\alpha}-\alpha-\gamma\alpha_0, 2\overline{\beta}-\beta)\| = \|(\alpha+\gamma\alpha_0,\beta)\| \tag{6}$$

If additionally $\alpha_0\overline{\alpha} = \max\{0, \alpha_0(\alpha+\gamma\alpha_0)\}$, i.e. when $g_1(x) = 0$ or $g_2(x) = 0$, or $g_1(x) = g_2(x) = 0$, then substituting $g_1(x)g_2(x) = \alpha_0^2\overline{\alpha}(\overline{\alpha}-\alpha-\gamma\alpha_0) = \overline{\alpha}(\overline{\alpha}-\alpha-\gamma\alpha_0) = 0$ into (6) further yields

$$\alpha_0\overline{\alpha} = \max\{0, \alpha_0(\alpha+\gamma\alpha_0)\}, \quad h_1(x) = 0 \quad \Longrightarrow \quad \|2\overline{\beta}-\beta\| = \|\beta\|. \tag{7}$$

Finally, from $|g_1(x) - g_2(x)| = |\alpha_0(\alpha+\gamma\alpha_0)| = |\alpha+\gamma\alpha_0| > 0$, it follows that we cannot jointly have both $g_1(x) = 0$ and $g_2(x) = 0$ at the same time. We proceed by analyzing that (5) holds true for the other three cases one at a time:

- If $g_1(x) = 0$ and $g_2(x) > 0$, then $y_2 = 0$. Substituting $y_2 = 0$ into (5), it follows from $\|2\overline{\beta}-\beta\| = \|\beta\| > 0$ via (7) and $\alpha_0^2 = 1$ that

$$\begin{bmatrix} \alpha_0 & 2\overline{\alpha}-\alpha-\gamma\alpha_0 \\ 0 & 2\overline{\beta}-\beta \\ \alpha_0\overline{\alpha} & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ z_1 \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} y_1 \\ z_1 \end{bmatrix} = 0.$$

- If $g_1(x) > 0$ and $g_2(x) = 0$, then $y_1 = 0$. Substituting $y_1 = 0$ into (5), it again follows from $\|2\overline{\beta}-\beta\| = \|\beta\| > 0$ via (7) and $\alpha_0^2 = 1$ that the following holds

$$\begin{bmatrix} \alpha_0 & 2\overline{\alpha}-\alpha-\gamma\alpha_0 \\ 0 & 2\overline{\beta}-\beta \\ \alpha_0(\overline{\alpha}-\alpha-\gamma\alpha_0) & 0 \end{bmatrix} \begin{bmatrix} y_2 \\ z_1 \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} y_2 \\ z_1 \end{bmatrix} = 0.$$

- Finally, if $g_1(x) > 0$ and $g_2(x) > 0$, then both $y_1 = y_2 = 0$. Substituting $y_1 = y_2 = 0$ into (5), it follows from $\|(2\overline{\alpha}-\alpha-\gamma\alpha_0, 2\overline{\beta}-\beta)\| = \|(\alpha+\gamma\alpha_0,\beta)\| > 0$ via (6) and $\alpha_0^2 = 1$ that

$$\begin{bmatrix} 2\overline{\alpha}-\alpha-\gamma\alpha_0 \\ 2\overline{\beta}-\beta \end{bmatrix} z_1 = 0 \qquad \Longleftrightarrow \qquad z_1 = 0.$$

□

Lemma F.2 (Single layer)**.** *Define* $u_0 \in \mathbb{R}$, $u \in \mathbb{R}^n$, $\overline{u} \in \mathbb{R}^{\overline{n}}$, $V \in \mathbb{R}^{n\times(r-1)}$ *and* $\overline{V} \in \mathbb{R}^{\overline{n}\times(r-1)}$. *Let* $x = (u_0, u, \overline{u}, \mathrm{vec}(V), \mathrm{vec}(\overline{V}))$ *satisfy* $g_1(x) \geq 0$, $g_2(x) \geq 0$ *and* $h_1(x) = 0$, *where*

$$g_1(x) = u_0 \cdot \overline{u}, \quad g_2(x) = u_0 \cdot (\overline{u} - Wu - bu_0), \quad h(x) = \mathrm{diag}[(\overline{u} - Wu - bu_0)\overline{u}^T + (\overline{V} - WV)\overline{V}^T].$$

Suppose that $u_0^2 = 1$, $\mathbf{e}_i^T(Wu + bu_0) \neq 0$ *and* $\mathbf{e}_i^T WV \neq 0$ *hold for all* $i \in \{1, 2, \ldots, \overline{n}\}$. *Then, the following holds*

$$\nabla g_1(x) y_1 + \nabla g_2(x) y_2 + \nabla h_1(x) z_1 = 0, \tag{8a}$$

$$g_1(x) \odot y_1 = g_2(x) \odot y_2 = 0, \tag{8b}$$

if and only if $y_1 = y_2 = z_1 = 0$.

Proof. Let $V = \begin{bmatrix} v_1 & \cdots v_{r-1} \end{bmatrix}$ and $\overline{V} = \begin{bmatrix} \overline{v}_1 & \cdots \overline{v}_{r-1} \end{bmatrix}$. Explicitly write out the gradient terms in (8a) and stack it together with (8b), we have

$$\begin{bmatrix} \overline{u}^T & (\overline{u} - Wu - 2bu_0)^T & -\overline{u}^T \mathrm{diag}(b) \\ 0 & -u_0 \cdot W^T & -W^T \mathrm{diag}(\overline{u}) \\ u_0 \cdot I & u_0 \cdot I & \mathrm{diag}(2\overline{u} - Wu - bu_0) \\ 0 & 0 & -W^T \mathrm{diag}(\overline{v}_1) \\ \vdots & \vdots & \vdots \\ 0 & 0 & -W^T \mathrm{diag}(\overline{v}_{r-1}) \\ 0 & 0 & \mathrm{diag}(2\overline{v}_1 - Wv_1) \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathrm{diag}(2\overline{v}_{r-1} - Wv_{r-1}) \\ u_0 \cdot \mathrm{diag}(\overline{u}) & 0 & 0 \\ 0 & u_0 \cdot \mathrm{diag}(\overline{u} - Wu - bu_0) & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0$$

Similar to Lemma F.1, deleting dependent rows allows us to restate the desired claim as the following

$$\begin{bmatrix} u_0 \cdot I & u_0 \cdot I & \mathrm{diag}(2\overline{u} - Wu - bu_0) \\ 0 & 0 & \mathrm{diag}(2\overline{v}_1 - Wv_1) \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathrm{diag}(2\overline{v}_{r-1} - Wv_{r-1}) \\ u_0 \cdot \mathrm{diag}(\overline{u}) & 0 & 0 \\ 0 & u_0 \cdot \mathrm{diag}(\overline{u} - Wu - bu_0) & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0 \quad \Longleftrightarrow \quad \begin{bmatrix} y_1 \\ y_2 \\ z_1 \end{bmatrix} = 0. \tag{9}$$

Collecting rows correspond to each $(\mathbf{e}_i^T y_1, \mathbf{e}_i^T y_2, \mathbf{e}_i^T z_1) = (y_{1,i}, y_{2,i}, z_{1,i})$, we see that (9) holds true if and only if the following holds true for all $i \in \{1, 2, \ldots, \overline{n}\}$:

$$\begin{bmatrix} \alpha_0 & \alpha_0 & 2\overline{\alpha}_i - \alpha_i - \gamma_i\alpha_0 \\ 0 & 0 & 2\overline{\beta}_i - \beta_i \\ \alpha_0\overline{\alpha}_i & 0 & 0 \\ 0 & \alpha_0(\overline{\alpha}_i - \alpha_i - \gamma_i\alpha_0) & 0 \end{bmatrix} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ z_{1,i} \end{bmatrix} = 0 \quad \Longleftrightarrow \quad \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ z_{1,i} \end{bmatrix} = 0. \tag{10}$$

where

$$\alpha_0 = u_0, \quad \alpha_i = \mathbf{e}_i^T Wu, \quad \overline{\alpha}_i = \mathbf{e}_i^T \overline{u}, \quad \beta_i^T = \mathbf{e}_i^T WV, \quad \overline{\beta}_i^T = \mathbf{e}_i^T \overline{V}, \quad \gamma_i = \mathbf{e}_i^T b.$$

By hypothesis, $u_0^2 = \alpha_0^2 = 1$, $\mathbf{e}_i^T(Wu + bu_0) = \alpha_i + \gamma_i\alpha_0 \neq 0$ and $\mathbf{e}_i^T WV = \beta_i^T \neq 0$, it then follows from Lemma F.1 that (10) holds true for all i. This proves the lemma. □

The results from Lemma F.1 and Lemma F.2 can be easily extended to the multiple layers case.

Lemma F.3 (Multiple layers). *Define $u_0 \in \mathbb{R}$, $u = (u_1, \dots, u_\ell) \in \mathbb{R}^n$, $V = (V_1, \dots, V_\ell) \in \mathbb{R}^{n\times(r-1)}$. Let $x = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell})$ satisfy $g_{k,1}(x) \geq 0$, $g_{k,2}(x) \geq 0$ and $h_k(x) = 0$ for all $k \in \{1, \dots, \ell-1\}$, where*

$$g_{k,1}(x) = u_0 \cdot u_{k+1}, \qquad g_{k,2}(x) = u_0 \cdot (u_{k+1} - W_k u_k - b_k u_0),$$
$$h_k(x) = \mathrm{diag}[(u_{k+1} - W_k u_k - b_k u_0)u_{k+1}^T + (V_{k+1} - WV_k)V_{k+1}^T].$$

Suppose that $u_0^2 = 1$, $\mathbf{e}_i^T(W_k u_k + b_k u_0) \neq 0$ and $\mathbf{e}_i^T W_k V_k \neq 0$ hold for all $k \in \{1, \dots, \ell-1\}$ and $i \in \{1, 2, \dots, n_{k+1}\}$. Then, the following holds

$$\sum_{k=1}^{\ell-1} [\nabla g_{k,1}(x) y_{k,1} + \nabla g_{k,2}(x) y_{k,2} + \nabla h_k(x) z_k] = 0, \tag{11a}$$

$$g_{k,1}(x) \odot y_{k,1} = g_{k,2}(x) \odot y_{k,2} = 0 \quad \textit{for all } k \in \{1, 2, \dots, \ell-1\}, \tag{11b}$$

if and only if $y_{k,1} = y_{k,2} = z_k = 0$ for all $k \in \{1, 2, \dots, \ell-1\}$.

Proof. Let us assume $r = 2$ without loss of generality. Similar to the proof in Lemma F.2, we start by writing (11a) and (11b) into a matrix-vector product. Let $\frac{\partial f(x)}{\partial y}$ denote the gradient of $f(x)$ with respect to variable y and let $x_k = (u_0, u_k, u_{k+1}, \mathrm{vec}(V_k), \mathrm{vec}(V_{k+1}))$. Analogous to the one layer case, for each $k \in \{1, \dots, \ell-1\}$, we define the following block matrix

$$\begin{bmatrix} \dfrac{\partial g_{1,k}(x)}{\partial x_k} & \dfrac{\partial g_{2,k}(x)}{\partial x_k} & \dfrac{\partial h_k(x)}{\partial x_k} \\ \mathrm{diag}(g_{1,k}(x)) & 0 & 0 \\ 0 & \mathrm{diag}(g_{2,k}(x)) & 0 \end{bmatrix} = \begin{bmatrix} a_k^T \\ B_k \\ C_k \\ D_k \\ E_k \\ F_k \end{bmatrix}, \qquad \mathbf{M}_k = \begin{bmatrix} C_k \\ E_k \\ F_k \end{bmatrix}.$$

Notice that $\mathbf{M}_k$ has the same structure as in (9). Each block above is assigned as the following

$$a_k^T = \frac{\partial(g_{k,1}, g_{k,2}, h_k)}{\partial u_0} = \begin{bmatrix} u_{k+1}^T & (u_{k+1} - Wu_k - 2b_k u_0)^T & -u_{k+1}^T \,\mathrm{diag}(b) \end{bmatrix},$$
$$B_k = \frac{\partial(g_{k,1}, g_{k,2}, h_k)}{\partial u_k} = \begin{bmatrix} 0 & -u_0 \cdot W_k^T & -W_k^T \,\mathrm{diag}(u_{k+1}) \end{bmatrix},$$
$$C_k = \frac{\partial(g_{k,1}, g_{k,2}, h_k)}{\partial u_{k+1}} = \begin{bmatrix} u_0 \cdot I & u_0 \cdot I & \mathrm{diag}(2u_{k+1} - W_k u_k - b_k u_0) \end{bmatrix},$$
$$D_k = \frac{\partial(g_{k,1}, g_{k,2}, h_k)}{\partial \,\mathrm{vec}(V_k)} = \begin{bmatrix} 0 & 0 & -W_k^T \,\mathrm{diag}(v_{k+1,1}) \\ \vdots & \vdots & \vdots \\ 0 & 0 & -W_k^T \,\mathrm{diag}(v_{k+1,r-1}) \end{bmatrix},$$
$$E_k = \frac{\partial(g_{k,1}, g_{k,2}, h_k)}{\partial \,\mathrm{vec}(V_{k+1})} = \begin{bmatrix} 0 & 0 & \mathrm{diag}(2v_{k+1,1} - W_k v_{k,1}) \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathrm{diag}(2v_{k+1,r-1} - W_k v_{k,r-1}) \end{bmatrix},$$
$$F_k = \begin{bmatrix} u_0 \cdot \mathrm{diag}(u_{k+1}) & 0 & 0 \\ 0 & u_0 \cdot \mathrm{diag}(u_{k+1} - W_k u_k - b_k u_0) & 0 \end{bmatrix}.$$

Since each $g_{k,1}(x), g_{k,2}(x), h_k(x)$ depends only on x_k. It follows that (11a) and (11b) can be written as a matrix-vector product in which the corresponding matrix admit a block tri-diagonal structure. This allows us to restated the desire claim

as the following

$$\begin{bmatrix} a_1^T & a_2^T & \cdots & a_{\ell-1}^T \\ B_1 & & & \\ C_1 & B_2 & & \\ & C_2 & \ddots & \\ & & \ddots & B_{\ell-1} \\ & & & C_{\ell-1} \\ D_1 & & & \\ E_1 & D_2 & & \\ & E_2 & \ddots & \\ & & \ddots & D_{\ell-1} \\ & & & E_{\ell-1} \\ F_1 & & & \\ & F_2 & & \\ & & \ddots & \\ & & & F_{\ell-1} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{\ell-1} \end{bmatrix} = 0 \qquad \iff \qquad \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{\ell-1} \end{bmatrix} = 0, \tag{12}$$

where $\lambda_k = (y_{k,1}, y_{k,2}, z_k)$.

We now process to show that (12) is true. Focusing our attention on the $(\ell-1)$-th block-row. Observe that the blocks $C_{\ell-1}$, $E_{\ell-1}$ and $F_{\ell-1}$ are the only nonzero blocks in their row; therefore, the left-hand side of (12) implies $\mathbf{M}_{\ell-1}\lambda_{\ell-1} = 0$. Given that $\mathbf{e}_i^T(W_{\ell-1}u_{\ell-1} + b_{\ell-1}u_0) \neq 0$ and $\mathbf{e}_i^T W_{\ell-1}V_{\ell-1} \neq 0$ hold for all $i \in \{1, 2, \ldots, n_\ell\}$ by hypothesis, it then follows from Lemma F.2 that $\mathbf{M}_{\ell-1}\lambda_{\ell-1} = 0 \iff \lambda_{\ell-1} = 0$.

Next, substituting $\lambda_{\ell-1} = 0$ back to the left-hand side of (12) to eliminate the $(\ell-1)$-th block-row. Repeat the same process for the $(\ell-2)$-th block-row all the way down to the first block-row to show that $\mathbf{M}_{\ell-2}\lambda_{\ell-2} = 0 \iff \lambda_{\ell-2} = 0$, $\mathbf{M}_{\ell-3}\lambda_{\ell-3} = 0 \iff \lambda_{\ell-3} = 0, \ldots,$ $\mathbf{M}_1\lambda_1 = 0 \iff \lambda_1 = 0$. This proves that the left-hand side of (12) does indeed imply the right-hand side under our stated hypotheses, as desired.

□

Theorem F.4 (LICQ for (BM-r))**.** *Define* $u_0 \in \mathbb{R}$, $u = (u_1, \ldots, u_\ell) \in \mathbb{R}^n$, $V = (V_1, \ldots, V_\ell) \in \mathbb{R}^{n\times(r-1)}$. *Let* $x = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell})$ *satisfy constraints in (BM-r),* $g_0(x) \geq 0$, $g_{k,1}(x) \geq 0$, $g_{k,2}(x) \geq 0$, $h_0(x) = 0$ *and* $h_k(x) = 0$ *for all* $k \in \{1, \ldots, \ell-1\}$, *where*

$$g_0(x) = \rho^2 - \|u_1 - \hat{x}u_0\|^2 - \|V_1\|^2, \qquad h_0(x) = u_0^2 - 1,$$
$$g_{k,1}(x) = u_0 \cdot u_{k+1}, \qquad g_{k,2}(x) = u_0 \cdot (u_{k+1} - W_k u_k - b_k u_0),$$
$$h_k(x) = \mathrm{diag}[(u_{k+1} - W_k u_k - b_k u_0)u_{k+1}^T + (V_{k+1} - WV_k)V_{k+1}^T].$$

Suppose that x satisfies (NPCQ), i.e. $\mathbf{e}_i^T(W_k u_k + b_k u_0) \neq 0$ *and* $\mathbf{e}_i^T W_k V_k \neq 0$ *hold for all* $k \in \{1, 2, \ldots, \ell-1\}$ *and* $i \in \{1, 2, \ldots, n_{k+1}\}$. *Then, x satisfies (LICQ) stated as the following:*

$$\nabla g_0(x)y_0 + \nabla h_0(x)z_0 + \sum_{k=1}^{\ell-1}[\nabla g_{k,1}(x)y_{k,1} + \nabla g_{k,2}(x)y_{k,2}] = 0, \tag{LICQ-a}$$

$$g_0 \odot y_0 = 0, \qquad g_{k,1}(x) \odot y_{k,1} = g_2(x) \odot y_{k,2} = 0 \quad \textit{for all } k \in \{1, \ldots, \ell-1\}, \tag{LICQ-b}$$

if and only if $y_0 = z_0 = 0$ *and* $y_{k,1} = y_{k,2} = z_k = 0$ for all $k \in \{0, 1, \ldots, \ell-1\}$.

Proof. Let us assume $r = 2$ and $\rho > 0$ without loss of generality. Similar to Lemma F.3, (LICQ-a) and (LICQ-b) admits a

tri-diagonal structure which allows us to restate the desired claim as the following

$$\begin{bmatrix} 2u_0 & a_0 & a_1^T & a_2^T & \cdots & a_{\ell-1}^T \\ & b_0 & B_1 & & & \\ & & C_1 & B_2 & & \\ & & & C_2 & \ddots & \\ & & & & \ddots & B_{\ell-1} \\ & & & & & C_{\ell-1} \\ & d_0 & D_1 & & & \\ & & E_1 & D_2 & & \\ & & & E_2 & \ddots & \\ & & & & \ddots & D_{\ell-1} \\ & & & & & E_{\ell-1} \\ & f_0 & & & & \\ & & F_1 & & & \\ & & & F_2 & & \\ & & & & \ddots & \\ & & & & & F_{\ell-1} \end{bmatrix} \begin{bmatrix} z_0 \\ y_0 \\ \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{\ell-1} \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} z_0 \\ y_0 \\ \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{\ell-1} \end{bmatrix} = 0, \tag{13}$$

in which $\lambda_k = (y_{k,1}, y_{k,2}, z_k)$. The blocks $(a_k, B_k, C_k, D_k, E_k)$ for all $k \in \{1, \ldots, \ell-1\}$ are defined in Lemma F.3, and the blocks (a_0, b_0, d_0, f_0) are written

$$a_0 = \frac{\partial g_0}{\partial u_0} = 2(\hat{x}^T u_1 - \|\hat{x}\|^2 u_0), \quad b_0 = \frac{\partial g_0}{\partial u_1} = 2(u_1 - \hat{x} u_0), \quad d_0 = \frac{\partial g_0}{\partial \operatorname{vec}(V_1)} = 2\operatorname{vec}(V_1), \quad f_0 = g_0(x).$$

Under the stated assumptions, we can verify the matrix at the left-hand side of (13) indeed has full column rank. First, we apply Lemma F.3 to show that

$$\mathbf{M}_{\ell-1}\lambda_{\ell-1} = 0 \iff \lambda_{\ell-1} = 0, \quad \mathbf{M}_{\ell-2}\lambda_{\ell-2} = 0 \iff \lambda_{\ell-2} = 0, \quad \ldots \quad , \mathbf{M}_1\lambda_1 = 0 \iff \lambda_1 = 0.$$

Substituting $\lambda_{\ell-1} = \lambda_{\ell-2} = \cdots = \lambda_1 = 0$ allows us to simplify (13) as

$$\begin{bmatrix} 2u_0 & a_0 \\ & b_0 \\ & d_0 \\ & f_0 \end{bmatrix} \begin{bmatrix} z_0 \\ y_0 \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} z_0 \\ y_0 \end{bmatrix} = 0. \tag{14}$$

To show that the matrix at the left-hand side of (14) has full column rank. We consider two cases:

- If $g_0(x) = 0$. It follows from $\|u_1 - \hat{x}u_0\|^2 + \|V_1\|^2 = \|(u_1 - \hat{x}u_0, \operatorname{vec}(V_1))\|^2 = \|\frac{1}{2}(b_0, d_0)\|^2 = \rho^2 > 0$ and $u_0^2 = 1$ that

$$\begin{bmatrix} 2u_0 & a_0 \\ & b_0 \\ & d_0 \\ & f_0 \end{bmatrix} \begin{bmatrix} z_0 \\ y_0 \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} z_0 \\ y_0 \end{bmatrix} = 0$$

- If $g_0(x) > 0$, then $y_0 = 0$. Substituting $y_0 = 0$ into (14), it then follows from $u_0^2 = 1$ that

$$2u_0 z_0 = 0 \qquad \Longleftrightarrow \qquad z_0 = 0.$$

□

We can now extend Theorem F.4 to show that (BM-ℓ_2) satisfies LICQ under an extra mild assumption, $\|V_1\| \neq 0$. The proof is summarized in the following corollary.

Corollary F.5 (LICQ for (BM-ℓ_2))**.** *Define u_0, u and V as in Theorem F.4. Let $x = (u_0, \{u_k\}_{k=1}^{\ell-1}, \{\mathrm{vec}(V_k)\}_{k=1}^{\ell})$ satisfy constraints in (BM-ℓ_2), $g_0(x) \geq 0$, $g_{k,1}(x) \geq 0$, $g_{k,2}(x) \geq 0$ and $h_k(x) = 0$ for all $k \in \{0, \dots, \ell-1\}$, where*

$$
\begin{aligned}
g_0(x) = \rho^2 - \|u_1 - \hat{x}u_0\|^2 - \|V_1\|^2, \qquad h_0(x) = u_0^2 - 1, \\
g_{0,1}(x) = u_0 \cdot u_1, \qquad g_{0,2}(x) = u_0 \cdot (u_0 - u_1)\,, \\
g_{k,1}(x) = u_0 \cdot u_{k+1}, \qquad g_{k,2}(x) = u_0 \cdot (u_{k+1} - W_k u_k - b_k u_0) \quad \textit{for all } k \in \{1, \dots, \ell-1\}, \\
h_k(x) = \mathrm{diag}[(u_{k+1} - W_k u_k - b_k u_0)u_{k+1}^T + (V_{k+1} - WV_k)V_{k+1}^T] \quad \textit{for all } k \in \{1, \dots, \ell-1\}.
\end{aligned}
$$

Suppose that x satisfies (NPCQ), and $\|V_1\| \neq 0$. Then, x satisfies (LICQ)

Proof. From Theorem F.4, to prove this corollary, it is suffice to show the following

$$
\begin{bmatrix}
2u_0 & a_0 & u_1^T & 2u_0 - u_1^T \\
 & b_0 & u_0 I & -u_0 I \\
 & d_0 & & \\
 & f_0 & & \\
 & & g_{0,1}(x) & \\
 & & & g_{0,2}(x)
\end{bmatrix}
\begin{bmatrix} z_0 \\ y_0 \\ y_{0,1} \\ y_{0,2} \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} z_0 \\ y_0 \\ y_{0,1} \\ y_{0,2} \end{bmatrix} = 0
$$

where (a_0, b_0, d_0, f_0) are defined in Theorem F.4. By hypothesis, $\|\frac{1}{2}d_0\| = \|V_1\| \neq 0$. This allows us to restate the desired claim as the following

$$
\begin{bmatrix}
2u_0 & u_1^T & 2u_0 - u_1^T \\
 & u_0 I & -u_0 I \\
 & g_{0,1}(x) & \\
 & & g_{0,2}(x)
\end{bmatrix}
\begin{bmatrix} z_0 \\ y_{0,1} \\ y_{0,2} \end{bmatrix} = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} z_0 \\ y_{0,1} \\ y_{0,2} \end{bmatrix} = 0. \tag{15}
$$

Notice that we cannot jointly have $\mathbf{e}_i^T g_{0,1}(x) = \mathbf{e}_i^T g_{0,2}(x) = 0$ for all $i \in \{1, \dots, n_1\}$. Hence, each pair of $(\mathbf{e}_i^T y_{0,1}, \mathbf{e}_i^T y_{0,2})$ has only three possible cases: $\mathbf{e}_i^T g_{0,1}(x) = 0, \mathbf{e}_i^T g_{0,2}(x) \neq 0$; $\mathbf{e}_i^T g_{0,1}(x) \neq 0, \mathbf{e}_i^T g_{0,2}(x) = 0$; and $\mathbf{e}_i^T g_{0,1}(x) \neq 0, \mathbf{e}_i^T g_{0,2}(x) \neq 0$. Of all three cases, it is clear that (15) is true because $u_0^2 = 1$.

□