Proceedings of Machine Learning Research vol 99:0–73, 2019 32nd Annual Conference on Learning Theory

# Learning Two Layer Rectified Neural Networks in Polynomial Time

**Ainesh Bakshi** ABAKSHI@CS.CMU.EDU
**Rajesh Jayaram** RKJAYARA@CS.CMU.EDU
**David P. Woodruff** DWOODRUF@CS.CMU.EEDU
*Carnegie Mellon University*

**Editors:** Alina Beygelzimer and Daniel Hsu

## Abstract

We consider the following fundamental problem in the study of neural networks: given input examples $x \in \mathbb{R}^d$ and their vector-valued labels, as defined by an underlying generative neural network, recover the weight matrices of this network. We consider two-layer networks, mapping $\mathbb{R}^d$ to $\mathbb{R}^m$, with a single hidden layer and $k$ non-linear activation units $f(\cdot)$, where $f(x) = \max\{x, 0\}$ is the ReLU activation function. Such a network is specified by two weight matrices, $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$, such that the label of an example $x \in \mathbb{R}^d$ is given by $\mathbf{U}^* f(\mathbf{V}^* x)$, where $f(\cdot)$ is applied coordinate-wise. Given $n$ samples $x^1, \ldots, x^n \in \mathbb{R}^d$ as a matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ and the label $\mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ of the network on these samples, our goal is to recover the weight matrices $\mathbf{U}^*$ and $\mathbf{V}^*$. More generally, our labels $\mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ may be corrupted by noise, and instead we observe $\mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$ where $\mathbf{E}$ is some noise matrix. Even in this case, we may still be interested in recovering good approximations to the weight matrices $\mathbf{U}^*$ and $\mathbf{V}^*$.

In this work, we develop algorithms and hardness results under varying assumptions on the input and noise. Although the problem is NP-hard even for $k = 2$, by assuming Gaussian marginals over the input $\mathbf{X}$ we are able to develop polynomial time algorithms for the approximate recovery of $\mathbf{U}^*$ and $\mathbf{V}^*$. Perhaps surprisingly, in the noiseless case our algorithms recover $\mathbf{U}^*, \mathbf{V}^*$ *exactly*, i.e. with no error, in *strongly* polynomial time. To the best of the our knowledge, this is the first algorithm to accomplish exact recovery for the ReLU activation function. For the noisy case, we give the first polynomial time algorithm that approximately recovers the weights in the presence of mean-zero noise $\mathbf{E}$. Our algorithms generalize to a larger class of *rectified* activation functions, $f(x) = 0$ when $x \leq 0$, and $f(x) > 0$ otherwise. Although our polynomial time results require $\mathbf{U}^*$ to have full column rank, we also give a fixed-parameter tractable algorithm (in $k$) when $\mathbf{U}^*$ does not have this property. Lastly, we give a fixed-parameter tractable algorithm for more arbitrary noise matrices $\mathbf{E}$, so long as they are independent of $\mathbf{X}$.

**Keywords:** neural networks, learning, polynomial time, ReLU, computational linear algebra

## 1. Introduction

Neural networks have achieved remarkable success in solving many modern machine learning problems which were previously considered to be intractable. With the use of neural networks now being wide-spread in numerous communities, the optimization of neural networks is an object of intensive study.

Common usage of neural networks involves running stochastic gradient descent (SGD) with simple non-linear activation functions, such as the extremely popular ReLU function, to learn an incredibly large set of weights. This technique has enjoyed immense success in solving complicated classification tasks with record-breaking accuracy. However, theoretically the behavior and convergence properties of SGD are very poorly understood, and few techniques are known which achieve provable bounds for the training of large neural networks. This is partially due to the hardness of the problem – there are numerous formulations where the problem is known to be NP-hard Blum and Rivest (1992); Judd (1988); Boob et al. (2018); Manurangsi and Reichman (2018). Nevertheless, given the importance and success in solving this problem in practice, it is important to understand the source of this hardness.

Typically a neural network can be written in the following form: $\mathbf{A} = \mathbf{U}^h(\cdots \mathbf{U}^3 f(\mathbf{U}^2 f(\mathbf{U}^1 \mathbf{X}))$, where $h$ is the depth of the network, $\mathbf{X} \in \mathbb{R}^{d\times n}$ is a matrix with columns corresponding to individual $d$-dimensional input samples, and $\mathbf{A}$ is the output labeling of $\mathbf{X}$. The functions $f$ are applied entry-wise to a matrix, and are typically non-linear. Perhaps the most popular activation used in practice is the ReLU, given by $f(x) = \max\{0, x\}$. Here each $\mathbf{U}^i$ is an unknown linear map, representing the "weights", which maps inputs from one layer to the next layer. In the reconstruction problem, when it is known that $\mathbf{A}$ and $\mathbf{X}$ are generated via the above model, the goal is to recover the matrices $\mathbf{U}^1, \ldots, \mathbf{U}^h$.

In this work, we consider the problem of learning the weights of two layer networks with a single non-linear layer. Such a network can be specified by two weight matrices $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ and $\mathbf{V}^* \in \mathbb{R}^{k\times d}$, such that, on a $d$-dimensional input vector $x \in \mathbb{R}^d$, the classification of the network is given by $\mathbf{U}^* f(\mathbf{V}^* x) \in \mathbb{R}^m$. Given a training set $\mathbf{X} \in \mathbb{R}^{d\times n}$ of $n$ examples, along with their labeling $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$, where $\mathbf{E}$ is a (possibly zero) noise matrix, the learning problem is to find $\mathbf{U}$ and $\mathbf{V}$ for which $\|\mathbf{U} - \mathbf{U}^*\|_F + \|\mathbf{V} - \mathbf{V}^*\|_F \leq \epsilon$

We consider two versions of this problem. First, in the noiseless (or realizable) case, we observe $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ precisely. In this setting, we demonstrate that exact recovery of the matrices $\mathbf{U}^*, \mathbf{V}^*$ is possible in polynomial time. Our algorithms, rather than exploiting smoothness of activation functions, exploit combinatorial properties of rectified activation functions. Additionally, we consider the more general noisy case, where we instead observe $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$, where $\mathbf{E}$ is a noise matrix which can satisfy various conditions. Perhaps the most common assumption in the literature Ge et al. (2018, 2017); Janzamin et al. (2015) is that $\mathbf{E}$ has mean 0 and is sub-Gaussian. Observe that the first condition is equivalent to the statement that $\mathbb{E}[\mathbf{A} \mid \mathbf{X}] = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$. While we primarily focus on designing polynomial time algorithms for this model of noise, in Section F we demonstrate fixed-parameter tractable (in the number $k$ of ReLUs) algorithms to learn the underlying neural network for a much wider class of noise matrices $\mathbf{E}$. We predominantly consider the *identifiable* case where $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ has full column rank, however we also provide supplementary algorithms for the exact case when $m < k$. Our algorithms are robust to the behavior of $f(x)$ for positive $x$, and therefore generalize beyond the ReLU to a wider class of rectified functions $f$ such that $f(x) = 0$ for $x \leq 0$ and $f(x) > 0$ otherwise.

It is known that stochastic gradient descent cannot converge to the ground truth parameters when $f$ is ReLU and $\mathbf{V}^*$ is orthonormal, even if we have access to an infinite number of samples Livni et al. (2014). This is consistent with empirical observations and theory, which states that over-parameterization is crucial to train neural networks successfully Hardt (2014); Soudry and Carmon (2016). In contrast, in this work we demonstrate that we can approximate the optimal parameters in the noisy case, and obtain the optimal parameters exactly in the realizable case, in polynomial time, without over-parameterization. In other words, we provide algorithms that do not succumb to spurious local minima, and can converge to the global optimum efficiently, without over-parametrization.

### 1.1. Our Contributions

We now state our results more formally. We consider 2-layer neural networks with ReLU-activation functions $f$. Such a neural network is specified by matrices $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ and $\mathbf{V}^* \in \mathbb{R}^{k\times d}$. We are given $d$-dimensional input examples $x^1, x^2 \dots x^n \in \mathbb{R}^d$, which form the columns of our input matrix $\mathbf{X}$, and also give the network's $m$-dimensional classification of $\mathbf{X}$, which is $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, where $f$ is applied entry-wise. We note that our formulation corresponds to having one non-linear layer.

**Worst Case Upper Bounds.** In the worst case setting, no properties are assumed on the inputs $\mathbf{X}, \mathbf{A}$. While this problem is generally assumed to be intractable, we show, perhaps surprisingly, that when $\text{rank}(\mathbf{A}) = k$ and $k = O(1)$, polynomial time exact algorithms do exist. One of our primary techniques throughout this work is the leveraging of combinatorial aspects of the ReLU function. For a row $f(\mathbf{V}^*\mathbf{X})_{i,*}$, we define a *sign pattern* of this row to simply be the subset of positive entries of the row. Thus, a sign pattern of a vector in $\mathbb{R}^n$ is simply given by the orthant of $\mathbb{R}^n$ in which it lies. We first prove an upper bound of $O(n^k)$ on the number of orthants which intersect with an arbitrary $k$-dimensional subspace of $\mathbb{R}^n$. Next, we show how to enumerate these sign patterns in time $n^{k+O(1)}$.

We use this result to give an $n^{O(k)}$ time algorithm for the neural network learning problem in the *realizable case*, where $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ for some fixed *rank-k* matrices $\mathbf{U}^*, \mathbf{V}^*$. After fixing a sign pattern of $f(\mathbf{V}^*\mathbf{X})$, we can effectively "remove" the non-linearity of $f$. Even so, the learning problem is still non-convex, and cannot be solved in polynomial time in the general case (even for fixed $k$). We show, however, that if the *rank* of $\mathbf{A}$ is $k$, then it is possible to use a sequence of linear programs to recover $\mathbf{U}^*, \mathbf{V}^*$ in polynomial time given the sign pattern, which allows for an $n^{O(k)}$ overall running time. Our theorem is stated below.

**Theorem 1** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{X} \in \mathbb{R}^{d\times n}$, such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and $\mathbf{A}$ is rank $k$, there is an algorithm that finds $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and runs in time poly$(n, m, d) \min\{n^{O(k)}, 2^n\}$.*

**Worst Case Lower Bounds.** Our upper bound relies crucially on the fact that $\mathbf{A}$ is rank $k$, which is full rank when $k \le d, m$. We demonstrate that an $O(n^k)$ time algorithm is no longer possible without this assumption by proving the NP-hardness of the realizable learning problem when $\text{rank}(\mathbf{A}) < k$, which holds even for $k$ as small as 2. Our hardness result is as follows.

**Theorem 2** *For a fixed $\alpha \in \mathbb{R}^{m\times k}, \mathbf{X} \in \mathbb{R}^{d\times n}, \mathbf{A} \in \mathbb{R}^{m\times n}$, the problem of deciding whether there exists a solution $\mathbf{V} \in \mathbb{R}^{k\times d}$ to $\alpha f(\mathbf{V}\mathbf{X}) = \mathbf{A}$ is NP-hard even for $k = 2$. Furthermore, for the case for $k = 2$, the problem is still NP-hard when $\alpha \in \mathbb{R}^{m\times 2}$ is allowed to be a variable.*

**Gaussian Inputs.** Since non-convex optimization problems are known to be NP-hard in general, it is, perhaps, unsatisfying to settle for worst-case results. To make such problems tractable, it is often assumed in the learning community that the input data is drawn from some underlying probability distribution. Therefore, we make the common step of assuming that the samples in $\mathbf{X}$ have a standard Gaussian distribution Zhong et al. (2017); Ge et al. (2017); Kalai et al. (2008); Plan and Vershynin (2013); Vempala (2010); Zhang et al. (2016). More generally, our algorithms work for arbitrary multi-variate Gaussian distributions over the columns of $\mathbf{X}$, as long as the covariance matrix is non-degenerate, i.e., full rank (see Remark 24). In this case, our running time and sample complexity will blow up by the condition number of the covariance matrix, which we can estimate first using standard techniques.

For simplicity, we state our results here for $\mathbf{\Sigma} = \mathbb{I}$, though, for the above reasons, all of our results for Gaussian inputs $\mathbf{X}$ extend to all full rank $\mathbf{\Sigma}$. Finally, we note that by combining our results with the recent results of Ge et al. (2018), we can extend our algorithms to arbitrary mixtures of Gaussians and symmetric distributions (see the discussion in Appendix C.3).

Furthermore, because many of our primary results utilize the combinatorial sparsity patterns of $f(\mathbf{VX})$, where $\mathbf{X}$ is a Gaussian matrix, we do not rely on the fact that $f(x)$ is linear for $x > 0$. For this reason, our results generalize easily to other *non-linear* rectified functions $f$. In other words, any function $f$ given by $f(x) = 0$ if $x \leq 0$ and $f(x) = \phi(x)$ otherwise, where $\phi(x) : [0, \infty] \to [0, \infty]$ is a continuous, injective function. In particular, our bounds do not change for polynomial valued $\phi(x) = x^c$ for $c \in \mathbb{N}$. For more details of this generalization, see Appendix G.1. Note, however, that our worst-case, non-distributional algorithms (stated earlier), where $\mathbf{X}$ is a fixed matrix, do not generalize to non-linear $\phi(x)$.

We first consider the noiseless setting, also referred to as the exact or realizable setting. Here $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ is given for rank $k$ matrices $\mathbf{U}^*$ and $\mathbf{V}^*$, where $\mathbf{X}$ has non-degenerate Gaussian marginals. The goal is then to recover the weights $(\mathbf{U}^*)^T, \mathbf{V}^*$ exactly up to a permutation of their rows (since one can always permute both sets of rows without effecting the output of the network). Note that for any positive diagonal matrix $\mathbf{D}$, $\mathbf{U}^* f(\mathbf{D}\mathbf{V}^* \mathbf{X}) = \mathbf{U}^* \mathbf{D} f(\mathbf{V}^* \mathbf{X})$ when $f$ is the ReLU. Thus recovery of $(\mathbf{U}^*)^T, \mathbf{V}^*$ is always only possible up to a permutation and positive scaling. We now state our main theorem for the exact recovery of the weights in the realizable (noiseless) setting.

**Theorem 3** *Suppose $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ where $\mathbf{U}^* \in \mathbb{R}^{m \times k}, \mathbf{V}^* \in \mathbb{R}^{k \times d}$ are both rank-$k$, and such that the columns of $\mathbf{X} \in \mathbb{R}^{d \times n}$ are mean 0 i.i.d. Gaussian. Then if $n = \Omega\left(\text{poly}(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))\right)$, then there is a poly$(n)$-time algorithm which recovers $(\mathbf{U}^*)^T, \mathbf{V}^*$ exactly up to a permutation of the rows with high probability.*

To the best of our knowledge, this is the first algorithm which learns the weight matrices of a two-layer neural network with ReLU activation *exactly* in the noiseless case and with Gaussian inputs $\mathbf{X}$. Our algorithm first obtains good approximations to the weights $\mathbf{U}^*, \mathbf{V}^*$, and concludes by solving a system of judiciously chosen linear equations, which we solve using Gaussian elimination. Therefore, we obtain exact solutions in polynomial time, without needing to deal with convergence guarantees of continuous optimization primitives. Furthermore, to demonstrate the robustness of our techniques, we show that using results introduced in the concurrent and independent work of Ge et. al. Ge et al. (2018), we can extend Theorem 3 to hold for inputs sampled from symmetric distributions (we refer the reader to Corollary 53). We note that Ge et al. (2018) recovers the weight matrices up to additive error $\epsilon$ and runs in poly $\left(\frac{1}{\epsilon}\right)$-time, whereas our algorithm has no $\epsilon$ dependency.

Observe, the *realizable* requirement, namely the requirement that $\mathbf{U}^*$ is rank $k$ forces $k \leq m$, i.e. the size of the output layer is at least as large as the hidden layer. We remark that this is in fact an important setting in practice. For instance, in autoencoders, the output layer is often much larger than the hidden layer, which in turn can shrink the input dimension. This also occurs in applications such as super resolution, which require a large output size. Moreover, most modern convolutional neural network architectures for ImageNet with Global Averaging Pool (GAP) also have this property. For example, in the ResNet architecture of He et al. (2016), the last hidden layer has 512 units, but the output size is 1000. Finally, there is a substantial body of literature which studies the compression of neural networks, and our algorithms with the rank constraint align with this literature. The goal in this setting is to approximate the behavior of a large, over-parameterized net with a more compact network. Assuming the existence of a good underlying network, our work can be seen as addressing this compression problem. Thus, we see the goal of optimizing over-parametrized networks as being disjoint from the goals of this work.

Also note that the runtime of our algorithm depends on the condition number $\kappa(\mathbf{V}^*)$ of $\mathbf{V}^*$, which is a fairly ubiquitous requirement in the literature for learning neural networks, and optimization in general Ge et al. (2018); Janzamin et al. (2015); Lee et al. (2015); Cohen et al. (2017); Arora et al. (2017); Zhong et al. (2017); Sedghi et al. (2016). To address this dependency, in Lemma 55 we give a lower bound which shows at least a linear dependence on $\kappa(\mathbf{V}^*)$ is necessary in the sample and time complexity. Next, we introduce an algorithm for approximate recovery of the weight matrices $\mathbf{U}^*, \mathbf{V}^*$ when $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$ for Gaussian marginals $\mathbf{X}$ and an i.i.d. sub-Gaussian mean-zero noise matrix $\mathbf{E}$ with variance $\sigma^2$.

**Theorem 4** *Let $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$ be given, where $\mathbf{U}^* \in \mathbb{R}^{m \times k}, \mathbf{V}^* \in \mathbb{R}^{k \times d}$ are rank-$k$, $\mathbf{E}$ is a matrix of i.i.d. mean-zero sub-Gaussian random variables with variance $\sigma^2$, and such that the columns of $\mathbf{X} \in \mathbb{R}^{d \times n}$ are i.i.d. Gaussian. Then given $n = \Omega\Big(poly\big(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma, \frac{1}{\epsilon}\big)\Big)$, there is an algorithm that runs in poly$(n)$ time and w.h.p. outputs $\mathbf{V}, \mathbf{U}$ such that $\|\mathbf{U} - \mathbf{U}^*\|_F \leq \epsilon$ and $\|\mathbf{V} - \mathbf{V}^*\|_F \leq \epsilon$*

Again, to the best of our knowledge, this work is the first which learns the weights of a 2-layer network in this noisy setting without additional constraints, such as the restriction that $\mathbf{U}$ be positive. Recent independent and concurrent work, using different techniques, achieves similar approximate recovery results in the noisy setting Ge et al. (2018). We note that the algorithm of Goel et. al. Goel and Klivans (2017) that Ge et al. (2018) uses, crucially requires the linearity of the ReLU for $x > 0$, and thus the work of Ge et al. (2018) does not generalize to the larger class of rectified functions which we handle. We also note that the algorithm of Ge et al. (2017) requires $\mathbf{U}^*$ to be non-negative. Finally, the algorithms presented in Janzamin et al. (2015) work for activation functions that are thrice differentiable and can only recover rows of $\mathbf{V}^*$ up to $\pm 1$ scaling. Note, for the ReLU activation function, we need to resolve the signs of each row.

**Fixed-Parameter Tractable Algorithms.** For several harder cases of the above problems, we are able to provide Fixed-Parameter Tractable algorithms. First, in the setting where the "labels" are vector valued, i.e., $m > 1$, we note prior results, not restricted to ReLU activation, require the rank of $\mathbf{U}^*$ to be $k$ Ge et al. (2018); Janzamin et al. (2015); Ge et al. (2017). This implies that $m \geq k$, namely, that the output dimension of the neural net is at least as large as the number $k$ of hidden neurons.

Perhaps surprisingly, however, we show that even when $\mathbf{U}^*$ does not have full column rank (which always occurs when $m < k$, we can still recover $\mathbf{U}^*$ exactly in the realizable case, as long as no two columns are non-negative scalar multiples of each other. Note that this allows for columns of the form $[u, -u]$ for $u \in \mathbb{R}^m$ as long as $u$ is non-zero. Our algorithm for doing so is fixed parameter tractable in the condition number of $\mathbf{V}^*$ and the number of hidden neurons $k$. Our results rely on proving bounds on the sample complexity in order to obtain all $2^k$ possible sparsity patterns of the $k$-dimensional columns of $f(\mathbf{V}^*\mathbf{X})$.

**Theorem 5** *Suppose $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ for $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ for any $m \geq 1$ such that no two columns of $\mathbf{U}^*$ are non-negative scalar multiples of each other, and $\mathbf{V}^* \in \mathbb{R}^{k\times n}$ has rank$(\mathbf{V}^*) = k$, and $n > \kappa^{O(k)}$poly$(dkm)$. Then there is an algorithm which recovers $\mathbf{U}^*, \mathbf{V}^*$ exactly with high probability in time $\kappa^{O(k)}$poly$(d, k, m)$.*

Furthermore, we generalize our results in the noisy setting to *arbitrary* error matrices $\|\mathbf{E}\|$, so long as they are independent of the Gaussians $\mathbf{X}$. In this setting, we consider a slightly different objective function, which is to find $\mathbf{U}, \mathbf{V}$ such that $\mathbf{U}f(\mathbf{V}\mathbf{X})$ approximates $\mathbf{A}$ well, where the measure is to compete against the optimal generative solution $\|\mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) - \mathbf{A}\|_F = \|\mathbf{E}\|_F$. Our results are stated below.

**Theorem 6** *Let $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ be given, where $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ are rank-$k$, and $\mathbf{E} \in \mathbb{R}^{m\times n}$ is any matrix independent of $\mathbf{X}$. Then there is an algorithm which outputs $\mathbf{U} \in \mathbb{R}^{m\times k}, \mathbf{V} \in \mathbb{R}^{k\times d}$ in time $(\kappa/\epsilon)^{O(k^2)}$poly$(n, d, m)$ such that with probability $1 - \exp(-\sqrt{n})$ we have $\|\mathbf{A} - \mathbf{U}f(\mathbf{V}\mathbf{X})\|_F \leq \|\mathbf{E}\|_F + O\Big(\Big[\sigma_{\max}\epsilon\sqrt{nm}\|\mathbf{E}\|_2\Big]^{1/2}\Big)$, where $\|\mathbf{E}\|_2$ is the spectral norm of $\mathbf{E}$.*

Note that the above error bounds depend on the flatness of the spectrum of $\mathbf{E}$. In particular, our bounds give a $(1+\epsilon)$ approximation whenever the spectral norm of $\mathbf{E}$ is a $\sqrt{m}$ factor smaller than the Frobenius norm, as is in the case for a wide class of random matrices Vershynin (2010). When this is not the case, we can scale $\epsilon$ by $1/\sqrt{m}$, to get an $(m\kappa/\epsilon)^{O(k^2)}$-time algorithm which gives a $(1+\epsilon)$ approximation for any error matrix $\mathbf{E}$ independent of $\mathbf{X}$ such that $\|\mathbf{E}\|_F = \Omega(\epsilon\|\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})\|_F)$.

**Sparse Noise.** Finally, we show that for *sparse noise*, when the network is *low-rank* we can reduce the problem to the problem of exact recovery in the noiseless case. Here, by low-rank we mean that $m > k$. It has frequently been observed in practice that many pre-trained neural-networks exhibit correlation and a low-rank structure Denil et al. (2013); Denton et al. (2014). Thus, in practice it is likely that $k$ need not be as large as $m$ to well-approximate the data. For such networks, we give a polynomial time algorithm for Gaussian $\mathbf{X}$ for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$. Our algorithm assumes that $\mathbf{U}^*$ has orthonormal columns, and satisfies an *incoherence* property, which is fairly standard in the numerical linear algebra community Candes and Romberg (2007); Candès and Recht (2009); Keshavan et al. (2010); Candès et al. (2011); Jain et al. (2013); Hardt (2014). Formally, assume $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ where $\mathbf{X}$ is i.i.d. Gaussian, and $\mathbf{E}$ is obtained from the following sparsity procedure. First, fix any matrix $\overline{\mathbf{E}}$, and randomly choose a subset of $nm - s$ entries for some $s < nm$, and set them equal to 0. The following result states that we can exactly recover $\mathbf{U}^*, \mathbf{V}^*$ in polynomial time even when $s = \Omega(mn)$.

**Theorem 7** *Let $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ be rank $k$ matrices, where $\mathbf{U}^*$ has orthonormal columns, $\max_{i\in[m]} \|(\mathbf{U}^*)^T e_i\|_2^2 \leq \frac{\mu k}{m}$ for some $\mu$, and $k \leq \frac{m}{\overline{\mu}\log^2(n)}$, where $\overline{\mu} = O\big((\kappa(\mathbf{V}^*))^2\sqrt{k\log(n)}\mu +$*

$\mu + (\kappa(\mathbf{V}^*))^4 \log(n)\big)$. *Here* $\kappa(\mathbf{V}^*)$ *is the condition number of* $\mathbf{V}^*$. *Let* $\mathbf{E}$ *be generated from the* $s$*-sparsity procedure with* $s = \gamma nm$ *for some constant* $\gamma > 0$ *and let* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}\mathbf{X}) + \mathbf{E}$. *Suppose the sample complexity satisfies* $n = poly(d, m, k, \kappa(\mathbf{V}^*))$. *Then on i.i.d. Gaussian input* $\mathbf{X}$ *there is a* $poly(n)$ *time algorithm that recovers* $\mathbf{U}^*, \mathbf{V}^*$ *exactly up to a permutation and positive scaling with high probability.*

### 1.2. Related Work

Recently, there has been a flurry of work developing provable algorithms for learning the weights of a neural network under varying assumptions on the activation functions, input distributions, and noise models Sedghi et al. (2016); Arora et al. (2016); Goel et al. (2016); Manurangsi and Reichman (2018); Zhong et al. (2017); Ge et al. (2018, 2017); Zhong et al. (2017); Tian (2017a); Li and Yuan (2017b); Brutzkus and Globerson (2017); Soltanolkotabi (2017); Goel et al. (2018); Du and Goel (2018); Du et al. (2018); Allen-Zhu et al. (2018); Vempala and Wilmes (2018). In addition, there have been a number of works which consider lower bounds for these problems under a similar number of varying assumptions Goel et al. (2016); Livni et al. (2014); Zhang et al. (2016); Sedghi et al. (2016); Arora et al. (2016); Boob et al. (2018); Manurangsi and Reichman (2018). We describe the main approaches here, and how they relate to our problem.

**Learning ReLU Networks without noise.** In the noiseless setting with Gaussian input, the results of Zhong et al. (2017) utilize a similar strategy as ours. Namely, they first apply techniques from tensor decomposition to find a good initialization of the weights, whereafter they can be learned to a higher degree of accuracy using other methods. At this point our techniques diverge, as they utilize gradient descent on the initialized weights, and demonstrate good convergence properties for *smooth* activation functions. However, their results do not give convergence guarantees for non-smooth activation functions, including the ReLU and the more general class of rectified functions considered in this work. In this work, once we are given a good initialization, we utilize combinatorial aspects of the sparsity patterns of ReLU's, as well as solving carefully chosen linear systems, to obtain exact solutions.

Li and Yuan (2017a) also analyze stochastic gradient descent, and demonstrate good convergence properties when the weight matrix $\mathbf{V}^*$ is known to be close to the identity, and $\mathbf{U}^* \in \mathbb{R}^{1\times k}$ is the all 1's vector. In Tian (2017b), stochastic gradient descent convergence is also analyzed when $\mathbf{U}^* \in \mathbb{R}^{1\times k}$ is the all 1's vector, and when $\mathbf{V}^*$ is orthonormal. Moreover, Tian (2017b) does not give bounds on sample complexity, and requires that a good initialization point is already given. For uniformly random and sparse weights in $[-1, 1]$, Arora et al. (2014) provide polynomial time learning algorithms. In Brutzkus and Globerson (2017), the learning of *convolutional neural networks* is considered, where they demonstrate global convergence of gradient descent, but do not provide sample complexity bounds.

**Learning ReLU Networks with noise.** Ge et al. (2017) considers learning a ReLU network with a single output dimension $\mathbf{A} = u^T f(\mathbf{V}\mathbf{X}) + \mathbf{E}$ where $u \in \mathbb{R}^k$ is restricted to be entry-wise positive and $\mathbf{E}$ is a zero-mean sub-Gaussian noise vector. In this setting, it is shown that the weights $u, \mathbf{V}$ can be approximately learned in polynomial time when the input $\mathbf{X}$ is i.i.d. Gaussian. However, in contrast to the algorithms in this work, the algorithm of Ge et al. (2017) relies heavily on the non-negativity of $u$ Ge (October, 2018), and thus cannot generalize to arbitrary $u$. Janzamin et al. (2015) utilize tensor decompositions to approximately learn the weights in the presence of mean

zero sub-Gaussian noise, when the activation functions are smooth and satisfy the property that $f(x) = 1 - f(-x)$. Using similar techniques, Sedghi et al. (2016) provide a polynomial time algorithm to approximate the weights, if the weights are sparse.

A more recent result of Ge et al. demonstrates polynomial time algorithms for learning weights of two-layer ReLU networks in the presence of mean zero sub-gaussian noise, when the input is drawn from a mixture of a symmetric and Gaussian distribution Ge et al. (2018). We remark that the results of Ge et al. (2018) were independently and concurrently developed, and utilize substantially different techniques than ours that rely crucially on the linearity of the ReLU for $x > 0$ Ge (October, 2018). For these reasons, their algorithms do not generalize to the larger class of rectified functions which are handled in this work. To the best of the our knowledge, for the case of Gaussian inputs, this work and Ge et al. (2018) are the first to obtain polynomial time learning algorithms for this noisy setting. A recent line of work addresses the problem of learning the weights of an over-parameterized neural network with arbitrary noise under varying assumptions on the input Du et al. (2018); Allen-Zhu et al. (2018). These algorithms initialize the weight matrices randomly, and then train the network with gradient descent. We note that over-parameterization is crucial for the success of their algorithms, and thus the goals of this line of work are disjoint from ours.

**Agnostic Learning.** A variety of works study learning ReLU's in the more general *agnostic* learning setting, based off Valiant's original PAC learning model Valiant (1984). The agnostic PAC model allows for arbitrary noisy and distributions over observations, and the goal is to output a hypothesis function which approximates the output of the neural network. Note that this does not necessarily entail learning the weights of an underlying network. For instance, Arora et al. (2016) gives an algorithm with $O(n^d)$ running time to minimize the empirical risk of a two-layer neural network. A closer analysis of the generalization bounds required in this algorithm for PAC learning is given in Manurangsi and Reichman (2018), which gives a $2^{\text{poly}(k/\epsilon)}\text{poly}(n, m, d, k)$ time algorithm under the constraints that $\mathbf{U}^* \in \{1, -1\}^k$ is given a fixed input, and both the input examples $\mathbf{X}$ and the weights $\mathbf{V}^*$ are restricted to being in the unit ball. In contrast, our $(\kappa/\epsilon)^{O(k^2)}$ time algorithm for general error matrices $\mathbf{E}$ improves on their complexity whenever $\kappa = O(2^{\text{poly}(k)})$, and moreover can handle arbitrarily large $\mathbf{V}^*$ and unknown $\mathbf{U}^* \in \mathbb{R}^{m \times k}$. We remark, however, that our loss function is different from that of the PAC model, and is in fact roughly equivalent to the empirical loss considered in Arora et al. (2016).

Note that the above algorithms *properly* learn the networks. That is, they actually output weight matrices $\mathbf{U}, \mathbf{V}$ such that $\mathbf{U}f(\mathbf{V}\mathbf{X})$ approximates the data well under some measure. A relaxation if this setting is *improper* learning, where the output of the learning algorithm can be any efficiently computable function, and not necessarily the weights of neural network. Several works have been studied that achieve polynomial running times under varying assumptions about the network parameters, such as Goel et al. (2016); Goel and Klivans (2017). The algorithm of Goel and Klivans (2017), returns a "clipped" polynomial. In addition, Zhang et al. (2016) gives polynomial time improper learning algorithms for multi-layer neural networks under several assumptions on the weights and activation functions.

**Hardness.** Hardness results for learning networks have an extensive history in the literature Judd (1988); Blum and Rivest (1992). Originally, hardness was considered for threshold activation functions $f(x) \in \{1, -1\}$, where it is known that even for two ReLU's the problem is NP-hard Blum and Rivest (1992). Very recently, there have been several concurrent and independent lower bounds developed for learning ReLU networks. The work of Boob et al. (2018) has demonstrated the hardness

of a neural network with the same number of nodes as the hard network in this paper, albeit with two applications of ReLU's (i.e., two non-linear layers) instead of one. Note that the hardness results of this work hold for even a single non-linear layer. Also concurrently and independently, a recent result of Manurangsi and Reichman (2018) appears to demonstrate the same NP-hardness as that in this paper, albiet using a slightly different reduction. The results of Manurangsi and Reichman (2018) also demonstrate that *approximately* learning even a single ReLU is NP-hard. In addition, there are also NP-hardness results with respect to improper learning of ReLU networks Goel et al. (2016); Livni et al. (2014); Zhang et al. (2016) under certain complexity-theoretic assumptions.

**Sparsity.** One of the main techniques of our work involves analyzing the sparsity patterns of the vectors in the rowspan of $\mathbf{A}$. Somewhat related reasoning has been applied by Spielman, Wang, and Wright to the dictionary learning problem Spielman et al. (2012). Here, given a matrix $\mathbf{A}$, the problem is to recover matrices $\mathbf{B}, \mathbf{X}$ such that $\mathbf{A} = \mathbf{B}\mathbf{X}$, where $\mathbf{X}$ is sparse. They argue the uniqueness of such a factorization by proving that, under certain conditions, the sparsest vectors in the row span of $\mathbf{A}$ are the precisely rows of $\mathbf{X}$. This informs their later algorithm for the exact recovery of these sparse vectors using linear programming.

### 1.3. Our Techniques

One of the primary technical contributions of this work is the utilization of the combinatorial structure of sparsity patterns of the rows of $f(\mathbf{V}\mathbf{X})$, where $f$ is a rectified function, to solve learning problems. Here, a sparsity pattern refers to the subset of coordinates of $f(\mathbf{V}\mathbf{X})$ which are non-zero, and a rectified function $f$ is one which satisfies $f(x) = 0$ for $x \leq 0$, and $f(x) > 0$ otherwise.

**Arbitrary Input.** For instance, given $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ where $\mathbf{U}^*, \mathbf{V}^*$ are full rank and $f$ is the ReLU, one approach to recovering the weights is to find $k$-linearly vectors $v_i$ such that $f(v_i\mathbf{X})$ span precisely the rows of $\mathbf{A}$. Without the function $f(\cdot)$, one could accomplish this by solving a linear system. Of course, the non-linearity of the activation function complicates matters significantly. Observe, however, that if the sparsity pattern of $f(\mathbf{V}^*\mathbf{X})$ was known before hand, one could simply "remove" $f$ on the coordinates where $f(\mathbf{V}^*\mathbf{X})$ is non-zero, and solve the linear system here. On all other coordinates, one knows that $f(\mathbf{V}^*\mathbf{X})$ is 0, and thus finding a linearly independent vector in the right row span can be solved with a linear system. Of course, naively one would need to iterate over $2^n$ possible sparsity patterns before finding the correct one. However, one can show that any $k$-dimensional subspace of $\mathbb{R}^n$ can intersect at most $n^k$ orthants of $\mathbb{R}^n$, and moreover these orthants can be enumerated in $n^k\text{poly}(n)$ time given the subspace. Thus the rowspan of $\mathbf{A}$, being $k$-dimensional, can contain vectors with at most $n^k$ patterns. This is the primary obervation behind our $n^k\text{poly}(n)$-time algorithm for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in the noiseless case (for arbitrary $\mathbf{X}$).

As mentioned before, the prior result requires $\mathbf{A}$ to be rank-$k$, otherwise the row span of $f(\mathbf{V}\mathbf{X})$ cannot be recovered from the row span of $\mathbf{A}$. We show that this difficulty is not merely a product of our specific algorithm, by demonstrating that even for $k$ as small as 2, if $\mathbf{U}^*$ is given as input then it is NP-hard to find $\mathbf{V}^*$ such that $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) = \mathbf{A}$, thus ruling out any general $n^k$ time algorithm for the problem. For the case of $k = 2$, the problem is still NP-hard even when $\mathbf{U}^*$ is not given as input, and is a variable.

**Gaussian Input.** In response to the aformentioned hardness results, we relax to the case where the input $\mathbf{X}$ has Gaussian marginals. In the noiseless case, we *exactly* learn the weights $\mathbf{U}^*, \mathbf{V}^*$ given $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ (up to a positive scaling and permutation). As mentioned, our results

utilize analysis of the sparsity patterns in the row-span of $\mathbf{A}$. One benefit of these techniques is that they are largely insensitive to the behavior of $f(x)$ for positive $x$, and instead rely on the rectified property $f(\cdot)$. Hence, this can include even exponential functions, and not solely the ReLU.

Our exact recovery algorithms proceed in two steps. First, we obtain an approximate version of the matrix $f(\mathbf{V}^*\mathbf{X})$. For a good enough approximation, we can exactly recover the sparsity pattern of $f(\mathbf{V}^*\mathbf{X})$. Our main insight is, roughly, that the only sparse vectors in the row span of $\mathbf{A}$ are precisely the rows of $f(\mathbf{V}^*\mathbf{X})$. Specifically, we show that the only vectors in the row span which have the same sparsity pattern as a row of $f(\mathbf{V}^*\mathbf{X})$ are scalar multiples of that row. Moreover, we show that no vector in the row span of $\mathbf{A}$ is supported on a strict subset of the support of a given row of $f(\mathbf{V}^*\mathbf{X})$. Using these facts, we can then set up a judiciously designed linear system to find these vectors, which allows us to recover $f(\mathbf{V}^*\mathbf{X})$ and then $\mathbf{V}^*$ exactly. By solving linear systems, we avoid using iterative continuous optimization methods, which recover a solution up to additive error $\epsilon$ and would only provide rates of convergence in terms of $\epsilon$. In contrast, Gaussian elimination yields exact solutions in a polynomial number of arithmetic operations. Therefore, our algorithms give exact solutions for $\mathbf{U}^*, \mathbf{V}^*$ in *strongly-polynomial time*, whereas, for instance, it is not known whether optimization problems such as linear programming can be solved in strongly polynomial time.

The first step, finding a good approximation of $f(\mathbf{V}^*\mathbf{X})$, can be approached from multiple angles. In this work, we demonstrate two different techniques to obtain these approximations, the first being Independent Component Analysis (ICA), and the second being tensor decomposition. To illustrate the robustness of our exact recovery procedure once a good estimate of $f(\mathbf{V}^*\mathbf{X})$ is known, we show in Appendix C.3 how we can bootstrap the estimators of recent, concurrent and independent work Ge et al. (2018), to improve them from approximate recovery to exact recovery.

**Independent Component Analysis.** In the restricted case when $\mathbf{V}^*$ is orthonormal, we show that our problem can be modeled as a special case of *Independent Component Analysis* (ICA). The ICA problem approximately recovers a subspace $\mathbf{B}$, given that the algorithm observes samples of the form $y = \mathbf{B}x + \zeta$, where $x$ is i.i.d. and drawn from a distribution that has moments bounded away from Gaussians, and $\zeta$ is a Gaussian noise vector. Intuitively, the goal of ICA is to find a linear transformation of the data such that each of the coordinates or features are as independent as possible. By rotational invariance of Gaussians, in this case $\mathbf{V}^*\mathbf{X}$ is also i.i.d. Gaussian, and we know that the columns of $f(\mathbf{V}^*\mathbf{X})$ have independent components and moments bounded away from a Gaussian. Thus, in the orthonormal case, our problem is well suited for the ICA framework.

**Tensor Decomposition.** A second, more general approach to approximating $f(\mathbf{V}^*\mathbf{X})$ is to utilize techniques from *tensor decomposition*. Our starting point is the generative model considered by Janzamin et. al. Janzamin et al. (2015), which matches our setting, i.e., $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$. The main idea behind this algorithm is to construct a tensor that is a function of both $\mathbf{A}, \mathbf{X}$ and captures non-linear correlations between them. A key step is to show that the resulting tensor has low CP-rank and the low-rank components actually capture the rows of the weight matrix $\mathbf{V}^*$. Intuitively, working with higher order tensors is necessary since matrix decompositions are only identifiable up to orthogonal components, whereas tensors have identifiable non-orthogonal components, and we are specifically interested in recovering approximations for non-orthonormal $\mathbf{V}^*$.

Next, we run a tensor decomposition algorithm to recover the low-rank components of the resulting tensor. While computing a tensor decomposition is NP-hard in general Hillar and Lim (2013), there is a plethora of work on special cases, where computing such decompositions is tractable

Bhaskara et al. (2014); Song et al. (2016); Wang and Anandkumar (2016); Goyal et al. (2014); Ge and Ma (2015); Barak and Moitra (2016). Tensor decomposition algorithms have recently become an invaluable algorithmic primitive and with applications in statistical and machine learning Janzamin et al. (2015, 2014); Ge et al. (2017); Anandkumar et al. (2014); Barak et al. (2015).

However, there are several technical hurdles involved in utilizing tensor decompositions to obtain estimates of $\mathbf{V}^*$. The first is that standard analysis of these methods utilizes a generalized version of *Stein's Lemma* to compute the expected value of the tensor, which relies on the smoothness of the activation function. Thus, we first approximate $f(\cdot)$ closely using a Chebyshev polynomial $p(\cdot)$ on a sufficiently large domain. However, we cannot algorithmically manipulate the input to demand that $\mathbf{A}$ instead be generated as $\mathbf{U}^* p(\mathbf{V}^*\mathbf{X})$. Instead, we add a small mean-zero Gaussian perturbation to our samples and analyze the variation distance between $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{G}$ and $\mathbf{U}^* p(\mathbf{V}^*\mathbf{X}) + \mathbf{G}$. For a good enough approximation $p$, this variation distance will be too small for any algorithm to distinguish between them, thus standard arguments imply the success of tensor decomposition algorithms when given the inputs $\mathbf{A} + \mathbf{G}$ and $\mathbf{X}$.

Next, a key step is to construct a non-linear transformation of the input by utilizing knowledge about the underlying density function for the distribution of $\mathbf{X}$, which we denote by $p(x)$. The non-linear function considered is the so-called Score Function, defined in Janzamin et al. (2014), which is the normalized $m$-th order derivative of the input probability distribution function $p(x)$. Computing the score function for an arbitrary distribution can be computationally challenging. However, as mentioned in Janzamin et al. (2014), we can use Hermite polynomials that help us compute a closed form for the score function, in the special case when $x \sim \mathcal{N}(0, \mathbb{I})$.

**Sign Ambiguity.** A further complication arises due to the fact that this form of tensor decomposition is agnostic to the signs of $\mathbf{V}$. Namely, we are guaranteed vectors $v_i$ from tensor decomposition such that $\|v_i - \xi_i \mathbf{V}^*_{i,*}\|_F < \epsilon$, where $\xi_i \in \{1, -1\}$ is some unknown sign. Prior works have dealt with this issue by considering restricted classes of smooth activation functions which satisfy $f(x) = 1 - f(-x)$ Janzamin et al. (2015). For such functions, one can compensate for not knowing the signs by allowing for an additional affine transformation in the neural network. Since we consider non-affine networks and rectified functions $f(\cdot)$ which do not satisfy this restriction, we must develop new methods to recover the signs $\xi_i$ to avoid the exponential blow-up needed to simply guess them. For the noiseless case, if $v_i$ is close enough to $\xi_i \mathbf{V}^*_{i,*}$, we can employ our previous results on the uniqueness of sparsity patterns in the row-span of $\mathbf{A}$. Namely, we can show that the sparsity pattern of $f(\xi v_i)$ will in fact be feasible in the row-span of $\mathbf{A}$, whereas the sparsity pattern of $f(-\xi v_i)$ will not, from which we recover the signs $\xi_i$ via a linear system.

In the presence of noise, however, the problem becomes substantially more complicated. Since we do not have the true row-span of $f(\mathbf{V}^*\mathbf{X})$, but instead a noisy row-span given by $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$, we cannot recover the $\xi_i$'s by feasibility arguments involving sparsity patterns. Our solution to the sign ambiguity in the noisy case is a projection-based scheme. Our scheme for determining $\xi_i$ involves constructing a $2k-2$ dimensional subspace $S$, spanned by vectors of the form $f(\pm v_j \mathbf{X})$ for all $j \neq i$. We augment this subspace as $S^1 = S \cup \{f(v_i \mathbf{X})\}$ and $S^{-1} = S \cup \{f(-v_i \mathbf{X})\}$. We then claim that the length of the projections of the rows of $\mathbf{A}$ onto the $S^\xi$ will be *smaller* for $\xi = \xi_i$ than for $\xi = -\xi_i$. Thus by averaging the projections of the rows of $\mathbf{A}$ onto these subspaces and finding the subspace which has the smaller projection length on average, we can recover the $\xi_i$'s with high probability. Our analysis involves bounds on projections onto perturbed subspaces, and a

spectral analysis of the matrices $f(\mathbf{WX})$, where $\mathbf{W}$ is composed of up to $2k$ rows of the form $\mathbf{V}^*_{i,*}$ and $-\mathbf{V}^*_{i,*}$.

**FPT Algorithms.** In addition to our polynomial time algorithms, we also demonstrate how various seemingly intractable relaxations to our model, within the Gaussian input setting, can be solved in fixed-parameter tractable time in the number $k$ of hidden units, and the condition numbers $\kappa$ of $\mathbf{U}^*$ and $\mathbf{V}^*$. Our first result demonstrates that, in the noiseless case, exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ is still possible even when $\mathbf{U}^*$ is not rank $k$. Note that the assumption that $\mathbf{U}^*$ is rank $k$ is required in many other works on learning neural networks Ge et al. (2017, 2018); Janzamin et al. (2015); Sedghi et al. (2016)

We demonstrate that taking $\text{poly}(d)\kappa^{O(k)}$ columns of $\mathbf{X}$, where $\kappa$ is the condition number of $\mathbf{V}^*$, is sufficient to obtain 1-sparse vectors in the columns of $f(\mathbf{V}^*\mathbf{X})$. As a result, we can look for column of $\mathbf{A}$ which are positive scalar multiples of each other, and conclude that any such pair will indeed be a positive scaling of a column of $\mathbf{U}^*$ with probability 1. This allows for exact recovery of $\mathbf{U}^*$ for any $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ and $m \geq 1$, as long as no two columns of $\mathbf{U}^*$ are positive scalar multiples of each other. Thereafter, we can recover $\mathbf{V}^*$ by solving a linear system on the subset of 1-sparse columns of $f(\mathbf{VX})$, and argue that the resulting constraint matrix is full rank. The result is a $\text{poly}(d, k, m)\kappa^O(k)$ time algorithm for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$. Our second FPT result involves a substantial generalization of the class of error matrices $\mathbf{E}$ which we can handle. In fact, we allow arbitrary $\mathbf{E}$, so long as they are independent of the input $\mathbf{X}$. Our primary technical observation is as follows. Suppose that we were given $f(v\mathbf{X}) + \mathbf{E}$, where $\mathbf{E}$ is an arbitrary, possibly very large, error vector, and $v \in \mathbb{R}^d$. Then one can look at the sign of each entry $i$, and consider it to be a noisy observation of which side of a halfspace the vector $\mathbf{X}_{*,i}$ lies within. In other words, we couch the problem as a noisy half-space learning problem, where the half-space is given by the hyperplane normal to $v$, and the labeling of $\mathbf{X}_{*,i}$ is the sign of $(f(v\mathbf{X}) + \mathbf{E})_i$.

Now while the error on each entry will be large, resulting in nearly half of the labelings being flipped incorrectly, because $\mathbf{E}$ is *independent* of $\mathbf{X}$, we are able to adapt recent techniques in noisy-halfspace learning to recover $v$ in polynomial time. In order to utilize these techniques without knowing anything about $\mathbf{E}$, we must first *smooth out* the error $\mathbf{E}$ by adding a large Gaussian matrix. The comparatively small value of $f(v\mathbf{X})$ is then able to shift the observed distribution of signs sufficiently to have non-trivial correlation with the true signs. Taking polynomially many samples, our algorithms detect this correlation, which will allow for accurate recovery of $v$. To even obtain a matrix of the form $f(v\mathbf{X}) + \mathbf{E}$, where $v$ is a row of $\mathbf{V}^*$, we can guess the pseudo-inverse of $\mathbf{U}^*$ to compute $(\mathbf{U}^*)^\dagger\mathbf{A}$, which we show can be done accurately enough for our purposes in time $(\kappa/\epsilon)^{O(k^2)}$, which dominates the overall runtime of the algorithm.

## 2. Sketch of Polynomial Time Exact Learning Algorithms

To illustrate some key techniques used in our algorithms, we briefly describe our algorithm for exact recovery of the weight matrices $\mathbf{U}^*, \mathbf{V}^*$ given $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and $\mathbf{X}$, where $\mathbf{X}$ is i.i.d. Gaussian input. We focus here on the noiseless case, where $\mathbf{A}$ is observed without corruption. Note that by exact recovery, we always mean up to a permutation and positive scaling of the rows of $(\mathbf{U}^*)^T$ and $\mathbf{V}^*$, since one can always apply these without changing the output of the neural net. For simplicity, we describe a restricted version of our algorithm which assumes that $\mathbf{V}^*$ has orthonormal rows. Our more general algorithm, which has no assumptions on $\mathbf{U}^*, \mathbf{V}^*$, is built using similar ideas. In this setting of orthonormal $\mathbf{V}^*$, we show how to use Independent Component Analysis (ICA) to recover

the matrix $\mathbf{U}^*$ approximately. Intuitively, for a fixed matrix $\mathbf{U}^*$, given independent samples of the form $\mathbf{U}^* x$ where $x \in \mathbb{R}^k$ has independent coordinates and has fourth moment bounded away from Gaussian, ICA returns a matrix $\widehat{\mathbf{U}}$ such that $\|\widehat{\mathbf{U}} - \mathbf{U}^*\|_F$ is small. Observe, by rotational invariance, for orthonormal $\mathbf{V}^*$, the columns of $\mathbf{V}^*\mathbf{X}$ will still be independent and Gaussian. Since applying $f$ to $\mathbf{V}^*\mathbf{X}$ does not change the independence, and the resulting distribution has fourth moment bounded away from Gaussian, ICA will indeed recover $\mathbf{U}^*$ approximately.

An abbreviated version of our algorithm is given in Algorithm 1. We first run ICA on the "samples", which are the columns of $\mathbf{A}$. This gives us an approximation $\widehat{\mathbf{U}}$ of $\mathbf{U}^*$. Next, we recover an approximation to $f(\mathbf{V}^*\mathbf{X})$ by simply computing $\widehat{\mathbf{U}}^\dagger\mathbf{A}$, where $\widehat{\mathbf{U}}^\dagger$ is the pseduo-inverse of $\widehat{\mathbf{U}}$, and setting all entries below a threshold $\tau$ to be 0. We denote the resulting matrix by $\widehat{f(\mathbf{V}\mathbf{X})}$. If the error $\|\widehat{\mathbf{U}} - \mathbf{U}^*\|_F$ is small enough, we can argue that the resulting *sign pattern* of $\widehat{f(\mathbf{V}\mathbf{X})}$, i.e. the subset of non-zero entries, will be exactly the same as the sign pattern of the true $f(\mathbf{V}^*\mathbf{X})$ with high probability. Next, we utilize our main combinatorial lemma (33), which demonstrates that this sign pattern is unique among the vectors in the row span of $\mathbf{A}$. Thus, if we can find $k$ row vectors $\mathbf{W} \in \mathbb{R}^{k\times n}$ such that $\mathbf{A}$ spans the rows of $\mathbf{W}$, and $\mathbf{W}$ and $\widehat{f(\mathbf{V}\mathbf{X})}$ have the same sign pattern, then we can conclude that $\mathbf{W}$ must in fact be equal to $f(\mathbf{V}^*\mathbf{X})$. To complete the proof, we demonstrate that such a matrix $\mathbf{W}$ can be obtained by solving a system of linear equations with Gaussian elimination, which gives a *strongly* polynomial time algorithm for exact recovery (which does not depend on any rates of convergence, unlike iterative methods). Once $f(\mathbf{V}^*\mathbf{X})$ is exactly recovered, exact recovery of $\mathbf{V}^*$ and then $\mathbf{U}^*$ is straightforward through solving another two such linear system.

**Algorithm 1: Exact Neural Net Learning via ICA (abbreviated)**

`Input :` Matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{X} \in \mathbb{R}^{r\times n}$ such that each entry in $\mathbf{X} \sim \mathcal{N}(0,1)$.

1. Run the ICA on $\mathbf{A}$ to recover $\widehat{\mathbf{U}}$ such that $\|\widehat{\mathbf{U}} - \mathbf{U}^*\|_F \leq \epsilon$ (for some permutation and positive scaling of $\mathbf{U}^*$).
2. Let $\tau$ be a threshold. Consider the matrix $\widehat{f(\mathbf{V}\mathbf{X})}$ obtained by setting all entries of $\widehat{\mathbf{U}}^\dagger\mathbf{A}$ smaller than $\tau$ to be 0. Let $S_j$ be the sign pattern of $\widehat{f(\mathbf{V}\mathbf{X})}_{j,*}$.
3. For all $j \in [k]$, solve the following linear system in unknowns $w_j \in \mathbb{R}^k$ such that $w_j \neq 0$ and $\forall \ell \in [n], (w_j\mathbf{A})_\ell \geq 0$ and $\forall i \notin S_j,\ (w_j\mathbf{A})_i = 0$. Let $\mathbf{W} \in \mathbb{R}^{k\times n}$ be the matrix where the $i$-th row is given by $w_i\mathbf{A}$.
4. $\mathbf{W}$ is equal to $f(\mathbf{V}^*\mathbf{X})$ (up to permutation and positive scalings), so we can recover $\mathbf{V}^*$ and then $\mathbf{U}^*$ exactly by solving two linear systems using Gaussian Elimination.

Our general algorithm for exact recovery of $\mathbf{U}^*$ and $\mathbf{V}^*$ does not require $\mathbf{V}^*$ to be orthonormal. However, for non-orthonormal $\mathbf{V}^*$, the components of $f(\mathbf{V}^*\mathbf{X})$ are no longer independent, and thus ICA no longer works to recover $\mathbf{U}^*$. Thus, instead of using ICA, in our general algorithm we replace the first step of Algorithm 1 with a tensor decomposition procedure, which similarly outputs a good approximation $\widehat{\mathbf{U}}$ of $\mathbf{U}^*$ (without any assumptions on $\mathbf{U}^*, \mathbf{V}^*$). Once $\widehat{\mathbf{U}}$ is obtained, we can proceed similarly, arguing about the uniqueness of sign patterns and solving a linear system to obtain $\mathbf{W}$, and ultimately $\mathbf{V}^*$ and $\mathbf{U}^*$.

## Acknowledgments

The authors thank the partial support by the National Science Foundation under Grant No. CCF-1815840. Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing. The authors would also like to thank Anima Anandkumar, Mark Bun, Rong Ge, Sam Hopkins and Rina Panigrahy for useful discussions.

## References

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962*, 2018.

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. *arXiv preprint arXiv:1611.01491*, 2016.

Sanjeev Arora, Rong Ge, Ankur Moitra, and Sushant Sachdeva. Provable ica with unknown gaussian noise, with implications for gaussian mixtures and autoencoders. In *Advances in Neural Information Processing Systems*, pages 2375–2383, 2012.

Sanjeev Arora, Aditya Bhaskara, Rong Ge, and Tengyu Ma. Provable bounds for learning some deep representations. In *International Conference on Machine Learning*, pages 584–592, 2014.

Sanjeev Arora, Rong Ge, Tengyu Ma, and Andrej Risteski. Provable learning of noisy-or networks. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1057–1066. ACM, 2017.

Robert B. Ash. Lecture notes 21-25 in statistics, finding the density. https://faculty.math.illinois.edu/~r-ash/Stat/StatLec21-25.pdf.

Boaz Barak and Ankur Moitra. Noisy tensor completion via the sum-of-squares hierarchy. In *Conference on Learning Theory*, pages 417–445, 2016.

Boaz Barak, Jonathan A Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 143–151. ACM, 2015.

Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 594–603. ACM, 2014.

Avrim Blum and Ronald L. Rivest. Training a 3-node neural network is np-complete. *Neural Networks*, 5(1):117–127, 1992.

Digvijay Boob, Santanu S Dey, and Guanghui Lan. Complexity of training relu neural network. *arXiv preprint arXiv:1809.10787*, 2018.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Alon Brutzkus and Amir Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. *arXiv preprint arXiv:1702.07966*, 2017.

Wlodzimierz Bryc. *The normal distribution: characterizations with applications*, volume 100. Springer Science & Business Media, 2012.

Emmanuel Candes and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

Michael B Cohen, Aleksander Madry, Dimitris Tsipras, and Adrian Vladu. Matrix scaling and balancing via box constrained newton's method and interior point methods. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 902–913. IEEE, 2017.

Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

Keith Conrad. Expository papers: Universal identities. http://www.math.uconn.edu/~kconrad/blurbs/linmultialg/univid.pdf.

A. DasGupta. *Asymptotic Theory of Statistics and Probability*. Springer Texts in Statistics. Springer New York, 2008. ISBN 9780387759708. URL https://books.google.com/books?id=9ByccYe5aI4C.

Misha Denil, Babak Shakibi, Laurent Dinh, Nando De Freitas, et al. Predicting parameters in deep learning. In *Advances in neural information processing systems*, pages 2148–2156, 2013.

Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, pages 1269–1277, 2014.

Simon S Du and Surbhi Goel. Improved learning of one-hidden-layer convolutional neural networks with overlaps. *arXiv preprint arXiv:1805.07798*, 2018.

Simon S Du, Jason D Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. *arXiv preprint arXiv:1811.03804*, 2018.

Alan Frieze, Mark Jerrum, and Ravi Kannan. Learning linear transformations. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 359–368. IEEE, 1996.

Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)*, 51(6):1025–1041, 2004.

Rong Ge. Personal communication. October, 2018.

Rong Ge and Tengyu Ma. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *arXiv preprint arXiv:1504.05287*, 2015.

Rong Ge, Jason D Lee, and Tengyu Ma. Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501*, 2017.

Rong Ge, Rohith Kuditipudi, Zhize Li, and Xiang Wang. Learning two-layer neural networks with symmetric inputs. *arXiv preprint arXiv:1810.06793*, 2018.

Surbhi Goel and Adam Klivans. Learning depth-three neural networks in polynomial time. *arXiv preprint arXiv:1709.06010*, 2017.

Surbhi Goel, Varun Kanade, Adam Klivans, and Justin Thaler. Reliably learning the relu in polynomial time. *arXiv preprint arXiv:1611.10258*, 2016.

Surbhi Goel, Adam Klivans, and Raghu Meka. Learning one convolutional layer with overlapping patches. *arXiv preprint arXiv:1802.02547*, 2018.

Navin Goyal, Santosh Vempala, and Ying Xiao. Fourier pca and robust tensor decomposition. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 584–593. ACM, 2014.

David Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566, 2011.

Allan Gut. An intermediate course in probability. chapter 5. Springer Publishing Company, Incorporated, 2009.

Moritz Hardt. Understanding alternating minimization for matrix completion. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 651–660. IEEE, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

Daniel Hsu and Sham M Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 11–20. ACM, 2013.

Aapo Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE transactions on Neural Networks*, 10(3):626–634, 1999.

Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM (JACM)*, 53(3):307–323, 2006.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674. ACM, 2013.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Score function features for discriminative learning: Matrix and tensor framework. *arXiv preprint arXiv:1412.2863*, 2014.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473*, 2015.

J Stephen Judd. Neural network design and the complexity of learning. Technical report, CALIFORNIA INST OF TECH PASADENA DEPT OF COMPUTER SCIENCE, 1988.

Sham M Kakade, Varun Kanade, Ohad Shamir, and Adam Kalai. Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems*, pages 927–935, 2011.

Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE transactions on information theory*, 56(6):2980–2998, 2010.

B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *Ann. Statist.*, 28(5):1302–1338, 10 2000. doi: 10.1214/aos/1015957395. URL https://doi.org/10.1214/aos/1015957395.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1049–1065. IEEE, 2015.

Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with relu activation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 597–607. Curran Associates, Inc., 2017a. URL http://papers.nips.cc/paper/6662-convergence-analysis-of-two-layer-neural-networks-with-relu-activation.pdf.

Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems*, pages 597–607, 2017b.

Yi-Kai Liu, Animashree Anandkumar, Dean P Foster, Daniel Hsu, and Sham M Kakade. Two svds su ffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. In *Neural Information Processing Systems (NIPS)*, 2012.

Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863, 2014.

Michael W Mahoney et al. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

Pasin Manurangsi and Daniel Reichman. The computational complexity of training relu(s). *arXiv preprint arXiv:1810.04207*, 2018.

Nimrod Megiddo. On the complexity of polyhedral separability. *Discrete & Computational Geometry*, 3(4):325–337, 1988.

Lingsheng Meng and Bing Zheng. The optimal perturbation bounds of the moorepenrose inverse under the frobenius norm. *Linear Algebra and its Applications*, 432(4):956 – 963, 2010. ISSN 0024-3795. doi: https://doi.org/10.1016/j.laa.2009.10.009. URL http://www.sciencedirect.com/science/article/pii/S0024379509005230.

Marco Mondelli and Andrea Montanari. On the connection between learning two-layers neural networks and tensor decomposition. *arXiv preprint arXiv:1802.07301*, 2018.

Yaniv Plan and Roman Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. *IEEE Transactions on Information Theory*, 59(1):482–494, 2013.

Mark Rudelson and Roman Vershynin. Non-asymptotic theory of random matrices: extreme singular values. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pages 1576–1602. World Scientific, 2010.

Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 143–152. IEEE, 2006.

Hanie Sedghi and Anima Anandkumar. Provable methods for training neural networks with sparse connectivity. *arXiv preprint arXiv:1412.2693*, 2014.

Hanie Sedghi, Majid Janzamin, and Anima Anandkumar. Provable tensor methods for learning mixtures of generalized linear models. In *Artificial Intelligence and Statistics*, pages 1223–1231, 2016.

Mahdi Soltanolkotabi. Learning relus via gradient descent. In *Advances in Neural Information Processing Systems*, pages 2007–2017, 2017.

Zhao Song, David Woodruff, and Huan Zhang. Sublinear time orthogonal tensor decomposition. In *Advances in Neural Information Processing Systems*, pages 793–801, 2016.

Daniel Soudry and Yair Carmon. No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361*, 2016.

Daniel A Spielman, Huan Wang, and John Wright. Exact recovery of sparsely-used dictionaries. In *Conference on Learning Theory*, pages 37–1, 2012.

Yuandong Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. *arXiv preprint arXiv:1703.00560*, 2017a.

Yuandong Tian. Symmetry-breaking convergence analysis of certain two-layered neural networks with relu nonlinearity. 2017b.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Santosh Vempala and John Wilmes. Polynomial convergence of gradient descent for training one-hidden-layer neural networks. *CoRR*, abs/1805.02677, 2018. URL http://arxiv.org/abs/1805.02677.

Santosh S Vempala. Learning convex concepts from gaussian distributions with pca. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 124–130. IEEE, 2010.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

M.J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019. ISBN 9781108498029. URL https://books.google.com/books?id=8C8nuQEACAAJ.

Yining Wang and Anima Anandkumar. Online and differentially-private tensor decomposition. In *Advances in Neural Information Processing Systems*, pages 3531–3539, 2016.

David P Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014.

Yuchen Zhang, Jason D Lee, and Michael I Jordan. l1-regularized neural networks are improperly learnable in polynomial time. In *International Conference on Machine Learning*, pages 993–1001, 2016.

Kai Zhong, Zhao Song, Prateek Jain, Peter L Bartlett, and Inderjit S Dhillon. Recovery guarantees for one-hidden-layer neural networks. *arXiv preprint arXiv:1706.03175*, 2017.

**Roadmap**

In Section A we introduce our $n^{O(k)}$ time exact algorithm when $\text{rank}(A) = k$ and arbitrary $\mathbf{X}$, for recovery of rank-$k$ matrices $\mathbf{U}^*, \mathbf{V}^*$ such that $\mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) = \mathbf{A}$. In this section, we also demonstrate that for a very wide class of distributions for *random* matices $\mathbf{X}$, the matrix $\mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ is in fact full rank with high probability, and therefore can be solved with our exact algorithm. Then, in Section B, we prove NP-hardness of the learning problem when $\text{rank}(A) < k$. Next, in Section C, we give a polynomial time algorithm for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in the case when $\mathbf{X}$ has Gaussian marginals in the realizable setting. Section C.1 develops our Independenct Component Analysis Based algorithm, whereas Section C.2 develops our more general exact recovery algorithm. In Section C.3, we show how recent concurrent results can be bootstrapped via our technqiues to obtain exact recovery for a wider class of distributions.

In Section D, we demonstrate how to extend our algorithm to the case where $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$ where $\mathbf{E}$ is mean $0$ i.i.d. sub-Gaussian noise. Then in Section E, we give a fixed-paramater

tractable (FPT) (in $k$ and $\kappa(\mathbf{V}^*)$) for the exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in the case where $\mathbf{U}^*$ does not have full column rank. We give our second FPT algorithm in Section F, which finds weights which approximate the optimal network for arbitrary error matrices $\mathbf{E}$ that are independent of $\mathbf{X}$. In Section G, we demonstrate how the weights of certain *low-rank* networks, where $k < d, m$, can be recovered exactly in the presence of a class of arbitrary sparse noise in polynomial time. Finally, in Appendix G.1, we give further details on generalizing the ReLU to the class of rectified activation functions.

**Preliminaries**

For a positive integer $k$, we write $[k]$ to denote the set $\{1, 2, \ldots, k\}$. We use the term *with high probability* (w.h.p.) in a parameter $r > 1$ to describe an event that occurs with probability $1 - \frac{1}{\text{poly}(r)}$. For a real $r$, we will often use the shorthand $\text{poly}(r)$ to denote a sufficiently large constant degree polynomial in $r$. Since for simplicity we do not seek to analyze or optimize the polynomial running time of our algorithms, we will state many of our error bounds within technical lemmas as $\frac{1}{\text{poly}(r)}$ where $r$ constitutes some set of relevant parameters, with the understanding that this polynomial can be made arbitrarily large by increasing the sample complexity $n$ of our algorithms by a polynomial factor.

In this work we use boldface font $\mathbf{A}, \mathbf{V}, \mathbf{U}, \mathbf{W}$ to denote matrices, and non-boldface font $x, y, u, v$ to denote vectors. For a vector $x$, we use $\|x\|_2$ to denote the $\ell_2$ norm of $x$. For any matrix $\mathbf{W}$ with $p$ rows and $q$ columns, for all $i \in [p]$, let $\mathbf{W}_{i,*}$ denote the $i$-th row of $\mathbf{W}$, for all $j \in [q]$ let $\mathbf{W}_{*,j}$ denote the $j$-th column and let $\mathbf{W}_{i,j}$ denote the $i, j$-th entry of $\mathbf{W}$. Further, the singular value decomposition of $\mathbf{W}$, denoted by $\texttt{SVD}(\mathbf{W}) = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, is such that $\mathbf{U}$ is a $p \times r$ matrix with orthonormal columns, $\mathbf{V}^T$ is a $r \times q$ matrix with orthonormal rows and $\mathbf{\Sigma}$ is an $r \times r$ diagonal matrix, where $r$ is the rank of $\mathbf{W}$. The entries along the diagonal are the singular values of $\mathbf{W}$, denoted by $\sigma_{\max} = \sigma_1(\mathbf{W}) \geq \sigma_2(\mathbf{W}) \geq \ldots \geq \sigma_r(\mathbf{W}) = \sigma_{\min}(\mathbf{W})$. We write $\|\mathbf{W}\|_F = \left(\sum_{p,q} \mathbf{W}_{p,q}^2\right)^{1/2}$ to denote the Frobenius norm of $\mathbf{W}$, and

$$\|\mathbf{W}\|_2 = \text{sup}_x \frac{\|\mathbf{A}x\|_2}{\|x\|_2} = \sigma_{\max}(\mathbf{W})$$

to denote the spectral norm. We will write $\mathbb{I}_k$ to denote the $k \times k$ square identity matrix. We use the notation $\texttt{Proj}_{\mathbf{W}}(w)$ to denote the projection of the vector $w$ onto the *row-span* of $\mathbf{W}$. In other words, if $x^* = \arg\min_x \|x\mathbf{W} - w\|_2$, then $\texttt{Proj}_{\mathbf{W}}(w) = x^*\mathbf{W}$. We now recall the condition number of a matrix $\mathbf{W}$.

**Definition 8** *For a rank $k$ matrix $\mathbf{W} \in \mathbb{R}^{p \times q}$, let $\sigma_{\max}(\mathbf{W}) = \sigma_1(\mathbf{W}) \geq \sigma_2(\mathbf{W}) \geq \cdots \geq \sigma_k(\mathbf{W}) = \sigma_{\min}(\mathbf{W})$ be the non-zero singular values of $\mathbf{W}$. Then the condition number $\kappa(\mathbf{W})$ of $\mathbf{W}$ is given by*

$$\kappa(\mathbf{W}) = \frac{\sigma_{\max}(\mathbf{W})}{\sigma_{\min}(\mathbf{W})}$$

*Note that if $\mathbf{W}$ has full column rank (i.e., $k = q$), then if $\mathbf{W}^\dagger$ is the pseudo-inverse of $\mathbf{W}$ we have $\mathbf{W}^\dagger\mathbf{W} = \mathbb{I}_q$ and*

$$\kappa(\mathbf{W}) = \|\mathbf{W}^\dagger\|_2\|\mathbf{W}\|_2$$

*where $\|\mathbf{W}\|_2 = \sigma_1(\mathbf{W})$ is the spectral norm of $\mathbf{W}$. Similarly if $\mathbf{W}$ has full row rank (i.e. $k = p$), then $\mathbf{W}\mathbf{W}^\dagger = \mathbb{I}_p$ and*

$$\kappa(\mathbf{W}) = \|\mathbf{W}^\dagger\|_2\|\mathbf{W}\|_2$$

A real $m$-th order tensor is $\mathbf{T} \in \otimes^m \mathbb{R}^d$ is the outer product of $m$ $d$-dimensional Euclidean spaces. A third order tensor $\mathbf{T} \in \otimes^3 \mathbb{R}^d$ is defined to be rank-1 if $\mathbf{T} = w \cdot a \otimes b \otimes c$ where $a, b, c \in \mathbb{R}^d$. Further, $\mathbf{T}$ has Candecomp/Parafac (CP) rank-$k$ if it can be written as the sum of $k$ rank-1 tensors, i.e.,

$$\mathbf{T} = \sum_{i=1}^{k} w_i a_i \otimes b_i \otimes c_i$$

is such that $w_i \in \mathbb{R}, a_i, b_i, c_i \in \mathbb{R}^d$. Next, given a function $f(x) : \mathbb{R}^d \to \mathbb{R}$, we use the notation $\nabla_x^m f(x) \in \otimes^m \mathbb{R}^d$ to denote the $m$-th order derivative operator w.r.t. the variable $x$, such that

$$[\nabla_x^m f(x)]_{i_1, i_2, \ldots i_m} = \frac{\partial f(x)}{\partial x_{i_1} \partial x_{i_2} \ldots \partial x_{i_m}}$$

.

In the context of the ReLU activation function, a useful notion to consider is that of a sign pattern, which will be used frequently in our analysis.

**Definition 9** *For any matrix dimensions $p, q$, a sign pattern is simply a subset of $[p] \times [q]$. For a matrix $\mathbf{W} \in \mathbb{R}^{p \times q}$, we let $\mathtt{sign}(\mathbf{W})$ be the sign pattern defined by*

$$\mathtt{sign}(\mathbf{W}) = \{(i, j) \in [p] \times [q] \mid \mathbf{W}_{i,j} > 0\}$$

Intuitively, in the context of rectified activation functions, the sign pattern is an important notion since $\mathtt{sign}(\mathbf{W})$ is invariant under application of $f$, in other words $\mathtt{sign}(\mathbf{W}) = f(\mathtt{sign}(\mathbf{W}))$. We similarly define a *sparsity-pattern* of a matrix $\mathbf{W} \in \mathbb{R}^{p \times q}$ as a subset of $[p] \times [q]$ where $\mathbf{W}$ is non-zero. Note that a sign and sparsity pattern of $\mathbf{W}$, taken together, specify precisely where the strictly positive, negative, and zero-valued entries are in $\mathbf{W}$.

We use the notation $\mathcal{N}(\mu, \sigma^2)$ to denote the Gaussian distribution with mean $\mu$ and variance $\sigma^2$. More generally, we write $\mathcal{N}(\mu, \mathbf{\Sigma})$ to denote a $k$-dimensional multi-variate Gaussian distribution with mean $\mu \in \mathbb{R}^k$ and variance $\mathbf{\Sigma} \in \mathbb{R}^{k \times k}$. We make use of the 2-stability of the Gaussian distribution several times in this work, so we now recall the following definition of stable random variables. We refer the reader to Indyk (2006) for a further discussion of such distributions.

**Definition 10** *A distribution $\mathcal{D}_p$ is said to be $p$-stable if whenever $\mathbf{X}_1, \ldots, \mathbf{X}_n \sim \mathcal{D}_p$ are drawn independently, we have*

$$\sum_{i=1}^{n} a_i \mathbf{X}_i \sim \|a\|_p \mathbf{X}$$

*for any fixed vector $a \in \mathbb{R}^n$, where $\mathbf{X} \sim \mathcal{D}_p$ is again distributed as a $p$-stable random variable. In particular, the Gaussian random variables $\mathcal{N}(0, \sigma^2)$ are $p$-stable for $p = 2$ (i.e., $\sum_i a_i g_i = \|a\|_2 g$, where $g, g_1, \ldots, g_n \sim \mathcal{N}(0, \sigma^2)$).*

Finally, we remark that in this paper, we will work in the common real RAM model of computation, where arithmetic operations on real numbers can be performed in constant time.

## Appendix A. Exact solution when rank$(\mathbf{A}) = k$

In this section, we consider the exact case of the neural network recovery problem. Given an input matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ of examples, and a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ of classifications, the exact version of the recovery problem is to obtain rank-$k$ matrices $\mathbf{U}^*, \mathbf{V}^*$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, if such matrices exist. In this section we demonstrate the existence of an $n^{O(k)}$poly$(md)$-time algorithm for exact recovery when rank$(\mathbf{A}) = k$. We demonstrate that this assumption is likely necessary in Section B, where we show that if rank$(\mathbf{A}) < k$ then the problem is NP-hard even for any $k \geq 2$ when the matrix $\mathbf{U}$ is given as input, and NP-hard for $k = 2$ when $\mathbf{U}^*$ is allowed to be a variable. This rules out the existence of a general $n^{O(k)}$ time algorithm for this problem.

The main theorem we prove in this section is that there is an algorithm with running time dominated by $\min\{n^{O(k)}, 2^n\}$ such that it recovers the underlying matrices $\mathbf{U}^*$ and $\mathbf{V}^*$ exactly. Intuitively, we begin by showing a structural result that there are at most $n^{O(k)}$ sign patterns that lie in the row space of $f(\mathbf{V}^*\mathbf{X})$ and we can efficiently enumerate over them using a linear program. For a fixed sign pattern in this set, we construct a sequence of $k$ linear programs (LP) such that the $i$-th LP finds a vector $y^i$, $f(y^i)$ is in the row span of $f(\mathbf{V}^*\mathbf{X})$, subject to the fixed sign pattern, and the constraint that $f(y^i)$ is not a linear combination of $f(y^1), f(y^2), \ldots f(y^{i-1})$. We note that $f(y^i)$ being linearly independent is not a linear constraint, but we demonstrate how it can be linearized in a straightforward manner.

Crucially, our algorithm relies on the fact that we have the row-span of $f(\mathbf{V}^*\mathbf{X})$. Note that this is implied by the assumption that $\mathbf{A}$ is rank $k$. Knowing the rowspan allows us to design the constraints in the prior paragraph, and thus solve the LP to recover the rows of $f(\mathbf{V}^*\mathbf{X})$. On the other hand, if the rank of $\mathbf{A}$ is less than $k$, then it no longer seems possible to efficiently determine the row span of $f(\mathbf{V}^*\mathbf{X})$. In fact, our NP-Hardness result of Section B demonstrates that, given $\mathbf{U}^*$ as input, if the rank of $\mathbf{A}$ is strictly less than $k$, the problem of determining the exact row-span of $f(\mathbf{V}^*\mathbf{X})$ is NP-Hard. The main result of this section is then as follows.

**Theorem 11** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{X} \in \mathbb{R}^{d\times n}$, there is an algorithm that finds $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and runs in time poly$(nmd) \min\{n^{O(k)}, 2^n\}$, if rank$(\mathbf{A}) = k$.*

Let $\mathbf{V}' \in \mathbb{R}^{k\times n}$ be a basis for the row-span of $\mathbf{A}$. For two matrices $\mathbf{Y}, \mathbf{Z}$ of the same dimension, we will write $\mathbf{Y} \overset{\text{row}}{\simeq} \mathbf{Z}$ if the row spans of $\mathbf{Y}$ and $\mathbf{Z}$ are the same. The first step in our algorithm is to obtain a *feasible set* $\mathcal{S}$ of sign patterns, within which the true sign pattern of $f(\mathbf{V}^*\mathbf{X})$ must lie.

**Lemma 12** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{X} \in \mathbb{R}^{d\times n}$, such that rank$(\mathbf{A}) = k$, there is an algorithm which runs in time $\min\{n^{O(k)}, 2^n\}$ and returns a set of sign patterns $\mathcal{S} \subset 2^{[m]\times[n]}$ with $|\mathcal{S}| = \min\{n^{O(k)}, 2^n\}$ such that for any rank-$k$ matrices $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and any row $i \in [k]$, $\texttt{sign}((\mathbf{V}^*\mathbf{X})_i) = \texttt{sign}(S)$ for some $S \in \mathcal{S}$.*

**Proof**

Recall, $\mathbf{A}$ is rank $k$. Thus there is a subset $\mathbf{V}' \in \mathbb{R}^{k\times n}$ of $k$ rows of $\mathbf{A}$ which span all the rows of $\mathbf{A}$. Critically, here we require that the rank of $\mathbf{A}$ is $k$ and thus the row space of $\mathbf{A}$ is the same as that of $f(\mathbf{V}^*\mathbf{X})$. Since $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and $\mathbf{V}', f(\mathbf{V}^*\mathbf{X})$ have the same dimensional row space, the row spaces of $\mathbf{V}'$ and $f(\mathbf{V}^*\mathbf{X})$ are precisely the same, and so there must be an invertible change of basis matrix $\mathbf{W}$ such that $\mathbf{W}\mathbf{V}' = f(\mathbf{V}^*\mathbf{X})$. Now note that $\texttt{sign}(\mathbf{V}^*\mathbf{X}) = \texttt{sign}(f(\mathbf{V}^*\mathbf{X})) = \texttt{sign}(\mathbf{W}\mathbf{V}')$, and thus it suffices to return a set of sign patterns $\mathcal{S}$ which contains $\texttt{sign}(\mathbf{W}\mathbf{V}')$.

Therefore, consider any fixed sign pattern $S \subset [n]$, and fix a row $j \in [k]$, and consider the following feasibility linear program in the variables $w_j$

$$(w_j \mathbf{V}')_i \geq 1, \quad \text{for all } i \in \texttt{sign}(S)$$

$$(w_j \mathbf{V}')_i \leq 0, \quad \text{for all } i \notin \texttt{sign}(S)$$

Note that if the sign pattern $S$ is feasible by some $w_j \mathbf{V}'$, then the above LP will be feasible with a suitably large positive scaling to $w_j$. Now the LP has $k$ variables and $n$ constraints, and thus a solution is obtained by choosing the $w_j$ that makes a subset of $k$ linearly independent constraints tight. Observe in any such LP of the above form, there are at most $2n$ possible constraints that can ever occur. Thus if $S$ is realizable as the sign pattern of some $w_j \mathbf{V}'$, then it is obtained by the unique solution to a system which chooses to make $k$ of these constraints tight. Formally, if $S, b$ are the constraints for which $w_j S \geq b$ in the LP, then a solution is given by $w_j S' = b'$ where $S', b'$ are a subset of $k$ of the constraints. Since there are at most $\binom{2n}{k} = O(n^k)$ such possible choices, it follows that there are at most $O(\min\{n^{O(k)}, 2^n\})$ realizable sign patterns, and these can be enumerated in $O(\min\{n^{O(k)}, 2^n\})$ time by simply checking the sign pattern which results from the solution (if one exists) to $w_j S' = b'$ taken over all subsets $S', b'$ of constraints of size $k$.

■

**Algorithm 2 : Iterative LP$\big(\mathbf{X}, S, y^1, y^2, \ldots y^{i-1}\big)$.**

`Input`: Matrix $\mathbf{X}$, a sign pattern $S$, vectors $y^1, y^2, \ldots y^{i-1}$ such that $f(y^1), f(y^2), \ldots f(y^{i-1})$ are linearly independent.

1. Let $y^i, z^i, w^i$ be variables in $\mathbb{R}^n$.
2. Let $\mathbf{Q} \in \mathbb{R}^{(i-1)\times n}$ be a matrix such that for all $j \in [i-1]$, $\mathbf{Q}_{j,*} = f(y^j)$. Construct the projection matrix $\mathbf{P}^{i-1}$ onto span $\big\{f(y^1), f(y^2), ..., f(y^{i-1})\big\}$. Note, the projection matrix is given by $\mathbf{P}^{i-1} = \mathbf{Q}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}$.
3. Define $f_S(y^i)$ w.r.t. the sign pattern $S$ such that

$$f_S(y^i_j) = \begin{cases} (y^i_j) & \text{if } j \in S \\ 0 & \text{otherwise} \end{cases}$$

`Output`: A feasible solution to the following LP:

$$\begin{array}{lll} \forall j \in [n] & y^i_j \geq 1, & \text{if } j \in S \\ \forall j \in [n] & y^i_j \leq 0, & \text{if } j \notin S \\ & y^i = w^i \mathbf{X} & \\ & f_S(y^i) = z^i \mathbf{V}' & \\ & f_S(y^i)(I - \mathbf{P}^{i-1}) \neq 0 & \end{array}$$

Given access to the set of candidate sign patterns, $S \in \mathcal{S}$, and vectors $y^1, y^2, ..., y^{i-1} \in \mathbb{R}^n$, we can define the following iterative feasibility linear program, that at each iteration $i$ finds a vector $y^i$

which is equal to some vector in the row span of $\mathbf{X}$, and such that $f(y^1), f(y^2), \ldots, f(y^i)$ are all linearly independent and in the row span of $\mathbf{A}$.

**Remark 13** *Observe, while the last constraint is not a linear constraint, it can be made linear by running $2n$ consecutive LP's, such that, for $t \in [n]$, in the $2t$-th LP we replace the constraint $f_S(y^i)(\mathbb{I} - \mathbf{P}^{i-1}) \neq 0$ above with*

$$\left[f_S(y^i)\left(\mathbb{I} - \mathbf{P}^{i-1}\right)\right]_t \geq 1$$

*and in the $(2t-1)$-th LP we replace constraint $f_S(y^i)\left(\mathbb{I} - \mathbf{P}^{i-1}\right) \neq 0$ with*

$$\left[f_S(y^i)\left(\mathbb{I} - \mathbf{P}^{i-1}\right)\right]_t \leq -1$$

*Note, the modified constraints are linear in the variables $y^i$. If there is a vector $y^i$ which satisfies the above constraints such that $f_S(y^i)(\mathbb{I} - \mathbf{P}^{i-1}) \neq 0$, then by scaling $y^i, w^i, z^i$ all by a sufficiently large positive constant, then $y^i$ will also satisfy one of the $2n$ LPs described above, thus giving a solution to the original feasibility problem by returning the first feasible solution returned among the $2n$ new LPs.*

Using Algorithm 2 as a sub-routine, we iterate over all sign patterns $S \in \mathcal{S}$, such that we recover a linearly independent set of $k$ vectors $f(y^1), f(y^2), \ldots f(y^k)$. Let $\mathbf{Y}$ be a matrix such that the $j$-th row corresponds to $y^j$. We then set up and solve two linear systems in $\mathbf{U}$ and $\mathbf{V}$, given by $\mathbf{A} = \mathbf{U}f(\mathbf{Y})$ and $\mathbf{Y} = \mathbf{V}\mathbf{X}$. We show that the solutions to the linear system correspond to $\mathbf{U}^*$ and $\mathbf{V}^*$. Here, we note that since the optimal $\mathbf{U}^*$ and $\mathbf{V}^*$ are solutions to a linear system, we can recover them exactly.

**Algorithm 3 : ExactNeuralNet$(\mathbf{A}, \mathbf{X}, \mathcal{S})$.**

`Input`: Matrices $\mathbf{A}, \mathbf{X}$, a set of sign patterns $\mathcal{S}$.

1. For $i = 1, 2, \ldots, k$
   (a) $t = 1$.
   (b) While$(t \leq |\mathcal{S}|)$
      i. If `Iterative LP`$\left(\mathbf{X}, S_t, y^1, y^2, ..., y^{i-1}\right)$ is feasible, let $y^i$ be the output, and set $t = |\mathcal{S}| + 1$.
      ii. Else $t \leftarrow t + 1$.
2. Let $\mathbf{Y} \in \mathbb{R}^{k \times n}$ be the matrix with $j$-th row equal to $y^j$ and let $S$ be the corresponding sign pattern.
3. Let $\mathbf{U}^*$ be the solution to the linear system in $\mathbf{U}$ given by $\mathbf{A} = \mathbf{U}f_S(\mathbf{Y})$.
4. Let $\mathbf{V}^*$ be the solution to the linear system in $\mathbf{V}$ given by $\mathbf{Y} = \mathbf{V}\mathbf{X}$.

`Output`: $\mathbf{U}^*, \mathbf{V}^*$.

**Lemma 14** *For any $i \in [k]$ vectors $y^1, y^2, ..., y^{i-1} \in \mathbb{R}^n$ and $S \in \mathcal{S}$, let $y^i$ be a feasible solution to `Iterative LP`$\left(\mathbf{X}, S, y^1, y^2, ..., y^{i-1}\right)$. Then all of the following hold:*

1. $y^i$ *is in the row span of* $\mathbf{X}$.
2. $f(y^i)$ *is in the row span of* $\mathbf{A}$.
3. $f(y^i)$ *is independent of* $f(y^1), f(y^2), ..., f(y^{i-1})$.

**Proof** The first condition follows due to the third constraint $y^i = w^i\mathbf{X}$. The first and second constraint ensure that $f_S(y^i) = f(y^i)$, thus along with the fourth constraint and the fact that $\mathbf{V}'$ spans the rows of $\mathbf{A}$, the second condition follows. For the last condition, it suffices to show that if $\|f(y^i)(\mathbb{I} - \mathbf{P}^{i-1})\| \geq 1$ then $f(y^i)$ is not in the span of $\{f(y^1), \ldots, f(y^{i-1})\}$. Now if $f(y^i)(\mathbb{I} - \mathbf{P}^{i-1}) = z \neq 0$, then $f(y^i) = z + \texttt{Proj}_{i-1}(f(y^i))$, where $\texttt{Proj}_{i-1}(f(y^i))$ is the projection of $f(y^i)$ onto the subspace spanned by $\{f(y^1), \ldots, f(y^{i-1})\}$. If $f(y^i)$ was in this subspace, then we would have $\texttt{Proj}_{i-1}(f(y^i)) = y^i$, but this is impossible since $z \neq 0$, which completes the proof. ■

**Lemma 15** *Suppose that there exist matrices* $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ *with* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$. *Then in the above algorithm, for each* $i \in [k]$ `Iterative LP`$\big(\mathbf{X}, S_t, y^1, y^2, ..., y^{i-1}\big)$ *will be feasible for at least one* $S_t \in \mathcal{S}$.

**Proof** The proof is by induction. For $i = 1$, since $f(\mathbf{V}^*\mathbf{X})$ has rank $k$ and spans the rows of $\mathbf{A}$, it follows that there must be some $j \in [k]$ such that the $j$-th row $f(\mathbf{V}^*\mathbf{X})_j$ of $f(\mathbf{V}\mathbf{X})$ is in in the row span of $\mathbf{V}'$, and clearly $(\mathbf{V}^*\mathbf{X})_j$ is in the row span of $\mathbf{X}$. The last constraint is of the LP non-existent since $i = 1$. Furthermore, $(\mathbf{V}^*\mathbf{X})_j$ has some sign pattern $S^*$, and it must be that $S^* \in \mathcal{S}$ by construction of $\mathcal{S}$. Then there exists a positive constant $c > 0$ such that $(c\mathbf{V}^*\mathbf{X})_j$ satisfies the last constraints of `Iterative LP`$\big(\mathbf{X}, S^*, y^1, y^2, ..., y^{i-1}\big)$ (made linear as described in Remark 13), and multiplying $(\mathbf{V}^*\mathbf{X})_j$ by a positive constant does not affect the fact that $(c\mathbf{V}^*\mathbf{X})_j$ is in the row space of $\mathbf{X}$ and $f(c\mathbf{V}^*\mathbf{X})_j$ is in the row space of $\mathbf{A}$ by closure of subspaces under scalar multiplication. Thus the `Iterative LP`$\big(\mathbf{X}, S^*, y^1, y^2, ..., y^{i-1}\big)$ has a feasible point.

Now suppose we have feasible points $y^1, \ldots, y^{i-1}$, with $i \leq k$. Note that this guarantees that $f(y^1), \ldots, f(y^{i-1})$ are linearly independent. Since $f(\mathbf{V}^*\mathbf{X})$ spans the $k$-dimensional row-space of $\mathbf{A}$, there must be some $j$ with $f(\mathbf{V}^*\mathbf{X})_j$ that is linearly independent of $f(y^1), \ldots, f(y^{i-1})$ such that $f(\mathbf{V}^*\mathbf{X})_j$ is in the row span of $\mathbf{A}$. Then $(\mathbf{V}^*\mathbf{X})_j$ is in the row span of $\mathbf{X}$, and similarly $(\mathbf{V}\mathbf{X})_j$ has some sign pattern $S^*$, and after multiplication by a suitably large constant it follows that the `Iterative LP`$\big(\mathbf{X}, S^*, y^1, y^2, ..., y^{i-1}\big)$ will be feasible. The proposition follows by induction. ■

**Proof of Theorem 11.** By Proposition 14, $f(y^1), \ldots, f(y^k)$ are independent, and give a solution to $f(\mathbf{V}\mathbf{X}) \overset{\text{row}}{\simeq} \mathbf{A}$. Thus we can find a $\mathbf{U} \in \mathbb{R}^{d\times k}$ in polynomial time via $d$ independent linear regression problems that solves $\mathbf{U}f(\mathbf{V}\mathbf{X}) = \mathbf{A}$. By Proposition 12, there are at most $\min\{n^{O(k)}, 2^n\}$ sign patterns in the set $\mathcal{S}$, and solving for each iteration of `Iterative LP` takes poly$(nm)$-time. Thus the total time is poly$(nmd)\min\{n^{O(k)}, 2^n\}$ as stated.

### A.1. Rank$(\mathbf{A}) = k$ for random matrices $\mathbf{X}$.

We conclude this section with the observation that if the input $\mathbf{X}$ is drawn from a large class of independent distributions, then the resulting matrix $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ will in fact be rank $k$ with high probability if $\mathbf{U}^*$ and $\mathbf{V}^*$ are rank $k$. Therefore, Algorithm 3 recovers $\mathbf{U}^*, \mathbf{V}^*$ in poly$(nmd)\min\{n^{O(k)}, 2^n\}$ for all such input matrices $\mathbf{X}$.

**Lemma 16** *Suppose* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ *for rank* $k$ *matrices* $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ *and* $\mathbf{V}^* \in \mathbb{R}^{k\times d}$*, where* $\mathbf{X} \in \mathbb{R}^{d\times n}$ *is a matrix of random variables such that each column* $\mathbf{X}_{*,i}$ *is drawn i.i.d. from a distribution* $\mathcal{D}$ *with continuous p.d.f.* $p(x) : \mathbb{R}^d \to \mathbb{R}$ *such that* $p(x) > 0$ *almost everywhere in* $\mathbb{R}^d$*, and such that*

$$\inf_{v\in\mathbb{R}^d} Pr_{x\sim\mathcal{D}}\big[\langle v, x\rangle > 0\big] > 10k\log(k/\delta)/n$$

*Then rank*$(A) = k$ *with probability* $1 - O(\delta)$*.*

**Proof** By Sylverster's rank inequality, it suffices to show $f(\mathbf{V}^*\mathbf{X})$ is rank $k$. By symmetry and i.i.d. of the $\mathbf{X}_{ij}$'s in a fixed row $i$, each entry $f(\mathbf{V}^*\mathbf{X})_{ij}$ is non-zero with probability at least $10k\log(k/\delta)/n$ independently (within the row $i$). Then by Chernoff bounds, a fixed row $(\mathbf{V}^*\mathbf{X})_{i,*}$ will have at least $k$ positive entries with probability at least $1 - 2^{-k\log(k/\delta)}$, and we can then union bound over all $k$ rows to hold with probability at least $1-O(\delta)$. Thus one can pick a $k\times k$ submatrix $\mathbf{W}$ of $f(\mathbf{V}^*\mathbf{X})$ such that, under some permutation $\mathbf{W}'$ of the columns of $\mathbf{W}$, the diagonal of $\mathbf{W}'$ is non-zero.

Since $\mathbf{V}^*$ is rank $k$, $\mathbf{V}^*$ is a surjective linear mapping of the columns of $\mathbf{X}$ from $\mathbb{R}^d$ to $\mathbb{R}^k$. Since $p(x) > 0$ almost everywhere, it follows that $p_{\mathbf{V}^*}(x) > 0$ almost everywhere, where $p_{\mathbf{V}^*}(x)$ is the continuous pdf of a column of $\mathbf{V}^*\mathbf{X}$. Then if $\mathbf{X}'$ is any matrix of $k$ columns of $\mathbf{X}$, by independence of the columns, if $p_{k\times k} : \mathbb{R}^{k^2} \to \mathbb{R}$ is the joint pdf of all $k^2$ variables in $\mathbf{V}^*\mathbf{X}'$, it follows that $p_{k\times k}(x) > 0$ for all $x \in \mathbb{R}^{k^2}$. Thus, by conditioning on any sign pattern $S$ of $\mathbf{V}^*\mathbf{X}'$, this results in a new pdf $p^S_{k\times k}$, which is simply $p_{k\times k}$ where the domain is restricted to an orthant $\Omega$ of $R^{k^2}$. Since $p_{k\times k}$ is continuous and non-zero almost everywhere, it follows that the support of the pdf $p^S_{k\times k} : \Omega \to \mathbb{R}$ is all of $\Omega$. In particular, the Lesbegue measure of the support $\Omega$ inside of $\mathbb{R}^{k^2}$ is non-zero (note that this would not be true if $\mathbf{V}^*$ has rank $k' < k$, as the support on each column would then be confined to a subspace of $\mathbb{R}^k$, which would have Lesbegue measure zero in $\mathbb{R}^k$).

Now after conditioning on a sign pattern, $\det(\mathbf{W}')$ is a non-zero polynomial in $s$ random variables, for $k \le s \le k^2$, and it is well known that such a function cannot vanish on any non-empty open set in $\mathbb{R}^s$ (see e.g. Theorem 2.6 of Conrad, and note the subsequent remark on replacing $\mathbb{C}^s$ with $\mathbb{R}^s$). It follows that the set of zeros of $\det(\mathbf{W}')$ contain no open set of $\mathbb{R}^s$, and thus has Lesbegue measure 0 in $\mathbb{R}^s$. By the remarks in the prior paragraph, we know that the Lesbegue measure (taken over $\mathbb{R}^s$) of the support of the joint distribution on the $s$ variables is non-zero (after restricting to the orthant given by the sign pattern). In particular, the set of zeros of $\det(\mathbf{W}')$ has Lesbegue measure 0 inside of the support of the joint pdf of the non-zero variables in $\mathbf{W}'$. We conclude that the joint density of the variables of $\mathbf{W}'$, after conditioning on a sign pattern, integrated over the set of zeros of $\det(\mathbf{W}')$ will be zero, meaning that $\mathbf{W}'$ will have full rank almost surely, conditioned on the sign pattern event in the first paragraph when held with probability $1 - O(\delta)$. ■

**Remark 17** *Note that nearly all non-degenerate distributions* $\mathcal{D}$ *on* $d$*-dimensional vectors will satisfy* $\inf_{v\in\mathbb{R}^d} Pr_{x\sim\mathcal{D}}\big[\langle v, x\rangle > 0\big] = c = \Omega(1)$*. For instance any multi-variate Gaussian distribution with non-degenreate (full-rank) covariance matrix* $\mathbf{\Sigma}$ *will satisfy this bound with* $c = 1/2$*, and this will also hold for any symmetric i.i.d. distribution over the entries of* $x \sim \mathcal{D}$*. Thus it will suffice to take* $n = \Omega(k\log(k/\delta))$ *for the result to hold.*

**Corollary 18** *Let* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ *for rank* $k$ *matrices* $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ *and* $\mathbf{V}^* \in \mathbb{R}^{k\times d}$*, where* $\mathbf{X} \in \mathbb{R}^{d\times n}$ *is a matrix of random variables such that each column* $\mathbf{X}_{*,i}$ *is drawn i.i.d. from a*

*distribution $\mathcal{D}$ with continuous p.d.f. $p(x) : \mathbb{R}^d \to \mathbb{R}$ such that $p(x) > 0$ almost everywhere in $\mathbb{R}^d$, and such that*

$$\inf_{v \in \mathbb{R}^d} Pr_{x \sim \mathcal{D}}\big[\langle v, x\rangle > 0\big] = \Omega(k \log(1/\delta)/n)$$

*Then, there exists an algorithm such that, with probability $1 - O(\delta)$, recovers $\mathbf{U}^*$, $\mathbf{V}^*$ exactly and runs in time poly$(n, m, d, k) \min\{n^{O(k)}, 2^n\}$.*

## Appendix B. NP-Hardness

The goal of this section is to prove that the problem of deciding whether there exists $\mathbf{V} \in \mathbb{R}^{k \times d}$ that solves the equation $\alpha f(\mathbf{VX}) = w$ for fixed input $\alpha \in \mathbb{R}^{m \times k}, X \in \mathbb{R}^{d \times n}, A \in \mathbb{R}^{m \times n}$, is NP-hard. We will first prove the NP-hardness of a geometric separability problem, which will then be used to prove NP-hardness for the problem of deciding the feasibility of $\alpha f(\mathbf{VX}) = w$. Our hardness reduction is from a variant of Boolean SAT, used in Megiddo (1988) to prove NP-hardness of a similar geometric seperability problem, called *reversible 6-SAT*, which we will now define. For a Boolean formula $\psi$ on variables $\{u_1, \ldots, u_n, \overline{u_1}, \ldots, u_n\}$ (where $\overline{u_i}$ is the negation of $u_i$), let $\overline{\psi}$ be the formula where every variable $u_i$ and $\overline{u_i}$ appearing in $\psi$ is replaced with $\overline{u_i}$ and $u_i$ respectively. For instance, if $\psi = (u_1 \vee u_2 \vee \overline{u_3}) \wedge (\overline{u_2} \vee u_3)$ then $\overline{\psi} = (\overline{u_1} \vee \overline{u}_2 \vee u_3) \wedge (u_2 \vee \overline{u_3})$.

**Definition 19** *A Boolean formula $\psi$ is said to be reversible if $\psi$ and $\overline{\psi}$ are both either satisfiable or not satisfiable.*

The reverse 6-SAT problem is then to, given a reversible Boolean formula $\psi$ where each conjunct has exactly six literals per clause, determine whether or not $\psi$ is satisfiable. Observe, if $\xi$ is a satisfying assignment to the variables of a reversible formula $\psi$, then $\overline{\xi}$, obtained by negating each assignment of $\xi$, is a satisfying assignment to $\overline{\psi}$. The following can be found in Megiddo (1988).

**Proposition 20 (NP-Hardness of Reversible 6-SAT Megiddo (1988))** *Given a reversible formula $\psi$ in conjunctive normal form where each clause has exactly six literals, it is NP-hard to decide whether $\psi$ is satisfiable.*

We now introduce the following *ReLU-seperability* problem, and demonstrate NP-hardness via a reduction from reversible 6-SAT.

**Definition 21 (ReLU-separability.)** *Given two sets $P = \{p_1, ..., p_r\}, Q = \{q_1, \ldots, q_s\}$ of vectors in $R^d$, the ReLU-seperability is to find vectors $x, y \in \mathbb{R}^d$ such that*

- *For all $p_i \in P$, both $p_i^T x \leq 0$ and $p_i^T y \leq 0$.*
- *For all $q_i \in Q$, we have $f(q_i^T x) + f(q_i^T y) = 1$ where $f(\cdot) = \max(\cdot, 0)$ is the ReLU function.*

*We say that an instance of ReLU-seperability is satisfiable if there exists such an $x, y \in \mathbb{R}^d$ that satisfy the above conditions.*

**Proposition 22** *It is NP-Hard to decide whether an instance of ReLU-seperability is satisfiable.*

**Proof**

Let $u_1, \ldots, u_n$ be the variables of the reversible 6-SAT instance $\psi$, and set $d = n + 2$, and let $x, y$ be the solutions to the instance of ReLU separability which we will now describe. The vector $x$ will be such that $x_i$ represents the truth value of $u_i$, and $y_i$ represents the truth value of $\overline{x_i} = \overline{u_i}$. For $j \in [n+2]$, let $e_j \in \mathbb{R}^{n+2}$ be the standard basis vector with a 1 in the $j$-th coordinate and 0 elsewhere. For each $i \in [n]$, we insert $e_i$ and $-e_i$ into $Q$. This ensures that $f(x_i) + f(y_i) = 1$ and $f(-x_i) + f(-y_i) = 1$. This occurs iff either $x_i = 1$ and $y_i = -1$ or $x_i = -1$ and $y_i = 1$, so $y_i$ is the negation of $x_i$. In other words, the case $x_i = 1$ and $y_i = -1$ means $u_i$ is true and $\overline{u_i}$ is false, and the case $x_i = -1$ and $y_i = 1$ means $u_i$ is false and $\overline{u_i}$ is true. Now suppose we have a clause of the form $u_1 \vee \overline{u_2} \vee u_3 \vee u_4 \vee \overline{u_5} \vee u_6$ in $\psi$. Then this clause can be represented equivalently by the inequality $x_1 - x_2 + x_3 + x_4 - x_5 + x_6 \geq -5$.

To represent this affine constraint, we add additional constraints that force $x_{n+1} + x_{n+2} = 1/2$ and $y_{n+1} + y_{n+2} = 1/2$ (note that the $n+1$, and $n+2$ coordinates do not correspond to any of the $n$ variables $u_i$). We force this as follows. Add $e_{n+1}$ and $e_{n+2}$ to $Q$, and add $-2e_{n+1}$, $-2e_{n+2}$ to $Q$. This forces $f(x_i)+f(y_i) = 1$ and $f(-2x_i)+f(-2y_i) = 1$ for each $i \in \{n+1, n+2\}$. For each $i \in \{n+1, n+2\}$ there are only two solutions, either $x_i = 1$ and $y_i = -1/2$ or $x_i = -1/2$ and $y_i = 1$. Finally, we add the vector $e_{n+1} + e_{n+2}$ to $Q$, which forces $f(x_{n+1} + x_{n+2}) + f(y_{n+1} + y_{n+2}) = 1$. Now if $x_{n+1} = 1$, then $x_{n+2}$ must be $-1/2$ since otherwise there is no solution to $2 + f(\cdot) = 1$, and we know $x_{n+2} \in \{1, -1/2\}$. This forces $y_{n+2} = 1$, which forces $x_{n+1} + x_{n+2} = 1/2 = y_{n+1} + y_{n+2}$, and a symmetric argument goes through when one assumes $y_{n+1} = 1$. This lets us write affine inequalities as follows. For the clause $u_1 \vee \overline{u_2} \vee u_3 \vee u_4 \vee \overline{u_5} \vee u_6$, we can write the corresponding equation $x_1 - x_2 + x_3 + x_4 - x_5 + x_6 \geq -5$ precisely as a point constraint, which for us is $(-1, 1, -1, -1, 1, -1, 0, 0, \ldots, 0, -10, -10) \in P$ (the two $-10$'s are in coordinate positions $n+1$ and $n+2$). Now this also forces the constraint $y_1 - y_2 + y_3 + y_4 - y_5 + y_6 \geq -5$, but since the formula is reversible so we can assume WLOG that $\overline{u_1} \vee u_2 \vee \overline{u_3} \vee \overline{u_4} \vee u_5 \vee \overline{u_6}$ is also a conjunct and so the feasible set is not affected, and the first $n$ coordinates of any solution $x$ will indeed correspond to a satisfying assignment to $\psi$ if one exists. Since reversible 6-SAT is NP-hard by Proposition 20, the stated result holds. ■

**Theorem 23** *For a fixed $\alpha \in \mathbb{R}^{m \times k}, \mathbf{X} \in \mathbb{R}^{d \times n}, \mathbf{A} \in \mathbb{R}^{m \times n}$, the problem of deciding whether there exists a solution $\mathbf{V} \in \mathbb{R}^{k \times d}$ to $\alpha f(\mathbf{VX}) = \mathbf{A}$ is NP-hard even for $k = 2$. Furthermore, for the case for $k = 2$, the problem is still NP-hard when $\alpha \in \mathbb{R}^{m \times 2}$ is allowed to be a variable.*

**Proof**

Now we show the reduction from ReLU-separability to our problem. Given an instance $(P, Q)$ of ReLU separability as in Definition 21, set $\alpha = [1, 1]$, and $w = [0, 0, \ldots, 0, 1, 1, \ldots, 1]$ so $w_i = 0$ for $i \leq r$ and $w_i = 1$ for $r < i \leq r + s$. Let $\mathbf{X} = [p_1,\ p_2,\ \ldots, p_r,\ q_1, \ldots,\ q_s] \in \mathbb{R}^{d \times (r+s)}$. Now suppose we have a solution $\mathbf{V} = [x, y]^T \in \mathbb{R}^{(r+s) \times 2}$ to $\alpha f(\mathbf{VX}) = w$. This means $f(p_i^T x) + f(p_i^t y) = 0$ for all $p_i \in P$, so it must be that both $p_i^T x \leq 0$and $p_i^T y \leq 0$. Also, we have $f(q_i^T x) + f(q_i^t y) = 1$ for all $q_i \in Q$. These two facts together mean that $x, y$ are a solution to ReLU-separability. Conversely, if solutions $x, y$ to ReLU separability exist, then for all $p_i \in P$, both $p_i^T x \leq 0$ and $p_i^T y \leq 0$ implies $f(p_i^T x) + f(p_i^t y) = 0$, and for all $q_i \in Q$ we get $f(q_i^T x) + f(q_i^T y) = 1$, so $\mathbf{V} = [x, y]^T$ is a solution to our factoring problem. Using the NP-hardness of ReLU-separability by Proposition 22, the result follows. Note here that $k = 2$ is a constant, but for larger $\alpha \in \mathbb{R}^{m \times k}$

with $m$ rows and $k$ columns, we can pad the new entries with zeros to reduce the problem to the aforementioned one, which completes the proof for a fixed $\alpha$.

Now for $k = 2$ and $\alpha$ a variable, we add the following constraints to reduce to the case of $\alpha = [1, 1]$, after which the result follows. First, we add 2 new columns and 1 new row to $\mathbf{X}$, giving $\mathbf{X}' \in \mathbb{R}^{(d+1)\times(r+s+2)}$. We set

$$\mathbf{X}' = \begin{bmatrix} \mathbf{X} & \mathbf{0} & \mathbf{0} \\ \mathbf{0}^T & 1 & -1 \end{bmatrix}$$

Where $\mathbf{X}$ is as in the last paragraph, where $\mathbf{0}$ is a column vector of the appropriate dimensions above. Also, we set $\mathbf{A}' = [\mathbf{A}, 1, 1] \in \mathbb{R}^{r+s+4}$. Let $\mathbf{V} = [x, y]^T$ as before. This ensures that $\alpha_1 f(x_{d+1}) + \alpha_2 f(y_{d+1}) = 1$ and $\alpha_1 f(-x_{d+1}) + \alpha_2 f(-y_{d+1}) = 1$. As before, we cannot have that both $(x_{d+1})$ and $(y_{d+1})$ are negative, or that both are positive, as then one of the two constraints would be impossible. WLOG, $(y_{d+1}) < 0$. Then we have $\alpha_1 f(x_{d+1}) = 1$, which ensures $\alpha_1 > 0$, and $\alpha_2 f(-y_{d+1}) = 1$, which ensures $\alpha_2 > 0$.

Now suppose we have a solution to $\mathbf{V} = [x, y]^T$ and $\alpha \in \mathbb{R}^2$ to this new problem with $\mathbf{X}', \mathbf{A}'$. Then we can set $x' = x/\alpha_1$ and $y' = y/\alpha_2$, and $\alpha' = [1, 1]$, and we argue that we have recovered a solution $[x', y']$ to ReLU separability. Note that $[1, 1]f([x', y']^T\mathbf{X}') = \mathbf{A}'$, since we can always pull a positive diagonal matrix in and out of $f$. Then restricting to the first $r + s$ columns of $\mathbf{X}', \mathbf{A}'$, we see that $[1, 1]f([x', y']^T\mathbf{X}) = \mathbf{A}$, thus $[x', y']$ are a solution to the neural-net learning problem as in the first paragraph, so as already seen we have that $x', y'$ is a solution to ReLU-separability. Similarly, any solution $x, y$ to ReLU separability can easily be extended to our learning problem by simply using $\mathbf{V} = \begin{bmatrix} x & 1 \\ y & -1 \end{bmatrix}$ and $\alpha = [1, 1]$, which completes the proof. ■

## Appendix C. A Polynomial Time Exact Algorithm for Gaussian Input

In this section, we study an exact algorithm for recovering the weights of a neural network in the realizable setting, i.e., the labels are generated by a neural network when the input is sampled from a Gaussian distribution. We also show that we can use independent and concurrent work of Ge et. al. Ge et al. (2018) to extend our algorithms to the input being sampled from a symmetric distribution. Our model is similar to non-linear generative models such as those for neural networks and generalized linear models already well-studied in the literature Sedghi et al. (2016); Sedghi and Anandkumar (2014); Kakade et al. (2011); Mondelli and Montanari (2018), but with the addition of the ReLU activation function $f$ and the second layer of weights $\mathbf{U}^*$. In other words, we receive as input i.i.d. Gaussian[1] input $\mathbf{X} \in \mathbb{R}^{d\times n}$ and the generated output is $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, where $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ and $\mathbf{V}^* \in \mathbb{R}^{k\times d}$. For the remainder of the section, we assume that both $\mathbf{V}^*$ and $\mathbf{U}^*$ are rank $k$. Note that this implies that $d \geq k$ and $k \leq m$. In Section E, however, we show that if we allow for a larger $((\kappa(\mathbf{V}^*))^{O(k)})$ sample complexity, we can recover $\mathbf{U}^*$ even when it is not full rank.

We note that the generative model considered in Sedghi and Anandkumar (2014) matches our setting, however, it requires the function $f$ to be differentiable and $\mathbf{V}^*$ to be sparse. In contrast, we focus on $f$ being ReLU. The ReLU activation function has gained a lot of popularity recently and is ubiquitous in applications Comon (1994); Hyvarinen (1999); Frieze et al. (1996); Hyvärinen and Oja (2000); Arora et al. (2012); Liu et al. (2012); Hsu and Kakade (2013). As mentioned in

1. See Remark 24

Sedghi et. al. Sedghi and Anandkumar (2014), if we make no assumptions on $\mathbf{V}^*$, the resulting optimal weight matrix is not identifiable. Here, we make no assumptions on $\mathbf{U}^*$ and $\mathbf{V}^*$ apart from them being full rank and show an algorithm that runs in polynomial time. The main technical contribution is then to recover the optimal $\mathbf{U}^*$ and $\mathbf{V}^*$ exactly, and not just up to $\epsilon$-error. By solving linear systems at the final step of our algorithms, as opposed to iterative continuous optimization methods, our algorithms terminate after a polynomial number of arithmetic operations.

Formally, suppose there exist fixed rank-$k$ matrices $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, and $\mathbf{X}$ is drawn from an i.i.d. Gaussian distribution. Note that we can assume that each row $\mathbf{V}_i^*$ of $\mathbf{V}^*$ satisfies $\|\mathbf{V}_i^*\|_2 = 1$ by pulling out a diagonal scaling matrix $\mathbf{D}$ with positive entries from $f$, and noting $\mathbf{U}^* f(\mathbf{D}\mathbf{V}^*\mathbf{X}) = (\mathbf{U}^*\mathbf{D})f(\mathbf{V}^*\mathbf{X})$. Our algorithm is given as input both $\mathbf{A}$ and $\mathbf{X}$, and tasked with recovering the underlying generative neural network $\mathbf{U}^*, \mathbf{V}^*$. In the context of training neural networks, we consider $\mathbf{X}$ to be the feature vectors and $\mathbf{A}$ to be the corresponding labels. Note $\mathbf{U}^*, \mathbf{V}^*$ are oblivious to $\mathbf{X}$, and are fixed prior to the generation of the random matrix $\mathbf{X}$. In this section we present an algorithm that is polynomial in all parameters, i.e., in the rank $k$, the condition number of $\mathbf{U}^*$ and $\mathbf{V}^*$, denoted by $\kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*)$ and $n, m, d$.

Given an approximate solution to $\mathbf{U}^*$, we show that there exists an algorithm that outputs $\mathbf{U}^*, \mathbf{V}^*$ exactly and runs in time polynomial in all parameters. We begin by giving an altenative algorithm for orthonormal $\mathbf{V}^*$ based on *Independent Component Analysis*. We believe that this perspective on learning neural networks may be useful beyond our results. Next, we will give a general algorithm for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ which does not require $\mathbf{V}^*$ to be orthonormal. This algorithm is based on the completely different approach of tensor decomposition, yet yields the same polynomial running time for exact recovery in the noiseless case. We now pause for a brief aside on the generalization of our results to the non-identity covariance case.

**Remark 24** *While our results are stated for when the columns of $\mathbf{X}$ are Gaussian with identity covariance, they can naturally be extended to $\mathbf{X}$ with arbitrary non-degenerate (full-rank) covariance $\mathbf{\Sigma}$, by noting that $\mathbf{X} = \mathbf{\Sigma}^{1/2}\mathbf{X}'$ where $\mathbf{X}'$ is i.i.d. Gaussian, and then implicitly replacing $\mathbf{V}^*$ with $\mathbf{V}^*\mathbf{\Sigma}^{1/2}$ so that $f(\mathbf{V}^*\mathbf{X}) = f((\mathbf{V}^*\Sigma^{1/2})\mathbf{X}')$, and noting that $\kappa(\mathbf{V}^*\Sigma^{1/2})$ blows up by a $\sqrt{\kappa(\mathbf{\Sigma})}$ factor from $\kappa(\mathbf{V}^*)$. All our remaining results, which do not require $\mathbf{V}^*$ to be orthonormal, hold with the addition of polynomial dependency on $\sqrt{\kappa(\mathbf{\Sigma})}$, by just thinking of $\mathbf{V}^*$ as $\mathbf{V}^*\Sigma^{1/2}$ instead.*

We use the sample covariance as our estimator for the true covariance $\mathbf{\Sigma}$ and have the following guarantee:

**Lemma 25** *(Estimating Covariance of $\mathbf{X}$ Vershynin (2018).) Let $\mathbf{X} \in \mathbb{R}^{d\times N}$ such that for all $i \in [N]$, $\mathbf{X}_{*,i} \sim \mathcal{N}(0, \mathbf{\Sigma})$. Let $\mathbf{\Sigma}_N = \frac{1}{N}\sum_{i\in[N]} \mathbf{X}_{*,i}\mathbf{X}_{*,i}^T$. With probability at least $1 - 2e^{-\delta}$,*

$$\|\mathbf{\Sigma} - \mathbf{\Sigma}_n\|_2 \leq c\frac{d+\delta}{N}\|\mathbf{\Sigma}\|_2$$

*for a fixed constant $c$.*

We can then estimate $\Sigma$ using a holdout set of $N = \Omega(n^2\delta^2)$ samples, which suffices to get an accurate estimate of the covariance matrix. We point out that, other than the tensor decomposition algorithm of Section C.2 and the noisy half-space learning routine in Section F, our algorithms do not even need to estimate the covariance matrix $\mathbf{\Sigma}$ in the multivariate case in order to approximately (or exactly) recover $\mathbf{U}^*, \mathbf{V}^*$. With regards to our tensor decomposition algorithms, while our estimator for the covariance introduces small error in the computation of the Score Function and the

resulting tensor decomposition, this can be handled easily in the perturbation analysis of Theorem 40 (refer to Remark 4 in Janzamin et al. (2015)). For our half-space learning algorithm in Section F, the error caused by estimating $\Sigma$ is negligible, and can be added to the "adversarial" error $\mathbf{B}$ of Theorem 69 which is already handled.

In the following warm-up Section C.1, where it is assumed that $\mathbf{V}^*$ is orthonormal, we cannot allow $\mathbf{X}$ to have arbitrary covariance, since then $\mathbf{V}^*\mathbf{\Sigma}^{1/2}$ would not be orthonormal. However, in the more general algorithm which follows in Section C.2, arbitrary non-degenerate covariance $\mathbf{\Sigma}$ is allowed.

### C.1. An Independent Component Analysis Algorithm for Orthonormal $\mathbf{V}^*$

We begin with making the simplifying assumption that the optimal $\mathbf{V}^*$ has orthonormal rows, as a warm-up to our more general algorithm. Note, if $\mathbf{V}^*$ is orthonormal and $\mathbf{X}$ is standard normal, then by 2-stability of Gaussian random variables, $\mathbf{V}^*\mathbf{X}$ is a matrix of i.i.d. Gaussian random variables. Since Gaussian random variables are symmetric around the origin, each column of $f(\mathbf{V}^*\mathbf{X})$ is sparse, has i.i.d entries, and has moments bounded away from Gaussians. Using these facts, we form a connection to the Independent Component Analysis (ICA) problem, and use standard algorithms for ICA to recover an approximation to $\mathbf{U}^*$.

The ICA problem approximately recovers a subspace $\mathbf{B}$, given that the algorithm observes samples of the form $y = \mathbf{B}x + \mathbf{E}$, where $x$ is i.i.d. and drawn from a distribution that has moments bounded away from Gaussians and $\mathbf{E}$ is Gaussian noise. The ICA problem has a rich history of theoretical and applied work Comon (1994); Frieze et al. (1996); Hyvarinen (1999); Hyvärinen and Oja (2000); Frieze et al. (2004); Liu et al. (2012); Arora et al. (2012); Hsu and Kakade (2013). Intuitively, the goal of ICA is to find a linear transformation of the data such that each of the coordinates or features are as independent as possible. For instance, if the dataset is generated as $y = \mathbf{B}x$, where $\mathbf{B}$ is an unknown affine transformation and $x$ has i.i.d. components, with no noise added, then applying $\mathbf{B}^{-1}$ to $y$ recovers the independent components exactly, as long as $x$ is non-Gaussian. Note, if $x \sim \mathcal{N}(0, \mathbb{I}_m)$, then by rotational invariance of Gaussians, we can only hope to recover $\mathbf{B}$ up to a rotation and the identity matrix suffices as a solution.

**Definition 26** *(Independent Component Analysis.) Given $\epsilon > 0$ and samples of the form $y_i = \mathbf{B}x_i + \mathbf{E}_i$, for all $i \in [n]$, such that $\mathbf{B} \in \mathbb{R}^{m \times m}$ is unknown and full rank, $x_i \in \mathbb{R}^m$ is a vector random variable with independent components and has fourth moments strictly less than that of a Gaussian, the ICA problem is to recover an additive error approximation to $\mathbf{B}$, i.e., recover a matrix $\widehat{\mathbf{B}}$ such that $\|\widehat{\mathbf{B}} - \mathbf{B}\|_F \leq \epsilon$.*

We use the algorithm provided in Arora et. al. Arora et al. (2012) as a black box for ICA. We note that our input distribution is rectified Gaussian, which differs from the one presented in Arora et al. (2012). Observe, our distribution is invariant to permutations and *positive* scaling, is sub-Gaussian, and has moments that are bounded away from Gaussian. The argument in Arora et al. (2012) extends to our setting, as conveyed to us via personal communication Ge (October, 2018). We have the following formal guarantee :

**Theorem 27** *(Provable ICA, Arora et al. (2012) and Ge (October, 2018).) Suppose we are given samples of the form $y_i = \mathbf{B}x_i + \mathbf{E}_i$ for $i = 1, 2, \ldots, n$, where $\mathbf{B} \in \mathbb{R}^{m \times m}$, the vector $x_i \in \mathbb{R}^m$ has i.i.d. components and has fourth moments strictly bounded away from Gaussian, and $\mathbf{E}_i \in \mathbb{R}^m$ is*

*distributed as $\mathcal{N}(0, \mathbb{I}_m)$, there exists an algorithm that with high probability recovers $\widehat{\mathbf{B}}$ such that $\|\widehat{\mathbf{B}} - \mathbf{B}\mathbf{\Pi}\mathbf{D}\|_F \leq \epsilon$, where $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{D}$ is a diagonal matrix such that it is entry-wise positive. Further, the sample complexity is $n = poly\left(\kappa(\mathbf{B}), \frac{1}{\epsilon}\right)$ and the running time is $poly(n, m)$.*

**Algorithm 4 : ExactNeuralNet**$(\mathbf{A}, \mathbf{X})$

`Input :` Matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{X} \in \mathbb{R}^{r \times n}$ such that each entry in $\mathbf{X} \sim \mathcal{N}(0, 1)$.

1. Let $\mathbf{T} \in \mathbb{R}^{k \times m}$ be a matrix such that for all $i \in [k]$, $j \in [m]$, $\mathbf{T}_{i,j} \sim \mathcal{N}(0, 1)$. Let $\mathbf{TA}$ be the matrix obtained by applying the sketch to $\mathbf{A}$.
2. Consider the ICA problem where we receive samples of the form $\mathbf{TA} = \mathbf{T}\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$.
3. Run the ICA algorithm, setting $\epsilon = \frac{1}{\text{poly}(m,d,k,\kappa(\mathbf{U}^*))}$, to recover $\widehat{\mathbf{TU}}$ such that $\|\widehat{\mathbf{TU}} - \mathbf{T}\mathbf{U}^*\mathbf{\Pi}\mathbf{D}\|_F \leq \frac{1}{\text{poly}(m,d,k,\kappa(\mathbf{U}^*))}$.
4. Let $\overline{\mathbf{X}}$ be the first $\ell = \text{poly}(d, m, k, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))$ columns of $\mathbf{X}$, and let $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}})$. Let $\tau = \frac{1}{\text{poly}(\ell)}$ be a threshold. Then for all $i \in [k], j \in [\ell]$, set
$$\widehat{f(\mathbf{V}\overline{\mathbf{X}})}_{i,j} = \begin{cases} 0 & \text{if } \left((\widehat{\mathbf{TU}})^{-1}\mathbf{T}\overline{\mathbf{A}}\right)_{i,j} \leq \tau \\ \left((\widehat{\mathbf{TU}})^{-1}\mathbf{T}\overline{\mathbf{A}}\right)_{i,j} & \text{otherwise} \end{cases}$$
5. Let $S_j$ be the sparsity pattern of the vector $\widehat{f(\mathbf{V}\overline{\mathbf{X}})}_{j,*}$. For all $j \in [k]$, and $r \in [k]$, solve the following linear system of equations in the unknowns $x_j^r \in \mathbb{R}^k$.
$$\begin{aligned} \forall i \in [\ell] \setminus S_j \quad & (x_j^r \overline{\mathbf{A}})_i = 0, \\ & (x_j^r)_r = 1 \end{aligned}$$
Where $(x_j^r)_r$ is the $r$-th coordinate of $x_j^r$.
6. Set $w_j$ to be the first vector $x_j^r$ such that a solution exists to the above linear system.
7. Let $\mathbf{W} \in \mathbb{R}^{k \times \ell}$ be the matrix where the $i$-th row is given by $w_i\overline{\mathbf{A}}$. Flip the signs of the rows of $\mathbf{W}$ so that $\mathbf{W}$ has no strictly negative entries.
8. For each $i \in [k]$, solve the linear system $(\mathbf{W}_{i,*})_{S_i} = \mathbf{V}_{i,*}\overline{\mathbf{X}}_{S_i}$ for $\mathbf{V} \in \mathbb{R}^{k \times d}$, where the subscript $S_i$ means restricting to the columns of $S_i$. Normalize $\mathbf{V}$ to have unit norm rows. Finally, solve the linear system $\mathbf{A} = \mathbf{U}f(\mathbf{V}\mathbf{X})$ for $\mathbf{U}$, using Gaussian Elimination.

`Output :` $\mathbf{U}, \mathbf{V}$.

We remark that ICA analyses typically require $\mathbf{B}$ to be a square matrix, and recall that$\mathbf{U}^*$ is $m \times k$ for $m \geq k$. To handle this, we sketch our samples using a dense Gaussian matrix with

exactly $k$ columns, and show this sketch is rank preserving. We will denote the resulting matrix by $\mathbf{TU}^*$.

**Lemma 28** *(Rank Preserving Sketch.) Let $\mathbf{T} \in \mathbb{R}^{k \times m}$ be a matrix such that for all $i \in [k]$, $j \in [m]$, $\mathbf{T}_{i,j} \sim \mathcal{N}(0,1)$. Let $\mathbf{U}^* \in \mathbb{R}^{m \times k}$ such that rank$(\mathbf{U}^*) = k$ and $m > k$. Then, $\mathbf{TU}^* \in \mathbb{R}^{k \times k}$ has rank $k$. Further, with probability at least $1-\delta$, $\kappa(\mathbf{TU}^*) \leq (k^2 m/\delta)\kappa(\mathbf{U}^*)$.*

**Proof** Let $\mathbf{M\Sigma N}^T$ be the SVD of $\mathbf{U}^*$, such that $\mathbf{M} \in \mathbb{R}^{m \times k}$ and $\mathbf{\Sigma N}^T \in \mathbb{R}^{k \times n}$. Since columns of $\mathbf{M}$ are orthonormal and Gaussians are rotationally invariant, $\mathbf{TM} \in \mathbb{R}^{k \times k}$ is i.i.d. standard normal. Further, $\mathbf{\Sigma N}^T$ has full row rank and thus has a right inverse, i.e., $\mathbf{N\Sigma}^{-1}$. Then, rank$(\mathbf{TU}) =$ rank$(\mathbf{TM\Sigma N}^T) \leq$ rank$(\mathbf{TM})$. Further $\mathbf{TM} = \mathbf{TU\Sigma}^{-1}$, and therefore rank$(\mathbf{TM}) =$ rank$(\mathbf{TU\Sigma}^{-1}) \leq$ rank$(\mathbf{TU})$. Recall, $\mathbf{TM}$ is a $m \times k$ matrix of standard Gaussian random variables and has a non-zero determinant with probability 1.

Next, $\kappa(\mathbf{TU}^*) \leq \kappa(\mathbf{T})\kappa(\mathbf{U}^*)$. Note $\mathbf{T}$ is at least $k+1 \times k$ and by Theorem 3.1 in Rudelson and Vershynin (2010), with probability $1-\delta$, $\sigma_{\min}(\mathbf{T}) \geq k\delta$. Similarly, by Proposition 2.4 Rudelson and Vershynin (2010), with probability $1 - 1/e^{\Omega(1/\delta)}$, $\sigma_{\max}(\mathbf{T}) \leq km/\delta$. Union bounding over the two events, with probability at least $1 - 1/\text{poly}(k)$, $\kappa(\mathbf{T}) \leq \text{poly}(k)$ and thus $\kappa(\mathbf{TU}^*) \leq \kappa(\mathbf{U})k^2 m/\delta$. ■

Algorithmically, we sketch the samples $\mathbf{TA}$ such that they are of the form $\mathbf{TU}^* f(\mathbf{V}^*\mathbf{X})$. By Lemma 28, $\mathbf{TU}^*$ is a square matrix and has rank $k$. Since $\mathbf{V}$ is orthonormal, each column of $f(\mathbf{V}^*\mathbf{X})$ has entries that are i.i.d. $\max\{\mathcal{N}(0,1), 0\}$. Note, the samples $\mathbf{TA}$ now fit the ICA framework, the noise $\mathbf{E} = 0$, and thus we can approximately recover $\mathbf{U}^*$, without even looking at the matrix $\mathbf{X}$. Here, we set $\epsilon = \frac{1}{\text{poly}(m,d,k,\kappa(\mathbf{U}^*))}$ to get the desired running time. Recall, given the polynomial depedence on $1/\epsilon$, we cannot recover $\mathbf{U}^*$ exactly.

**Corollary 29** *(Approximate Recovery using ICA.) Given $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{X} \in \mathbb{R}^{d \times n}$, and a sketching matrix $\mathbf{T} \in \mathbb{R}^{k \times m}$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and for all $i \in [k]$, $j \in [m]$, $\mathbf{T}_{i,j} \sim \mathcal{N}(0,1)$, there exists an algorithm that outputs an estimator to $\widehat{\mathbf{TU}^*}$ such that $\|\widehat{\mathbf{TU}} - \mathbf{TU}^*\mathbf{\Pi D}\|_F \leq \frac{1}{poly(m,d,k,\kappa(\mathbf{U}^*))}$, where $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{D}$ is strictly positive diagonal matrix. Further, the running time is poly $(m, d, k, \kappa(\mathbf{U}^*))$.*

**Exact Recovery:** By Corollary 29, running ICA on $\mathbf{TA} = \mathbf{TU}^* f(\mathbf{V}^*\mathbf{X})$, we recover $\mathbf{TU}^*$ approximately up to a permutation and positive scaling of the column. Note that we can disregard the permutation by simply assuming $\mathbf{V}$ has been permuted to agree with the $\mathbf{\Pi}$. Let $\widehat{\mathbf{TU}}$ be our estimate of $\mathbf{TU}^*$. We then restrict our attention to the first $\ell =$ poly $(d, m, k, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))$ columns of $\mathbf{X}$, and call this submatrix $\overline{\mathbf{X}}$, and $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}})$. We then multiply $\mathbf{T}\overline{\mathbf{A}}$ by the inverse $(\widehat{\mathbf{TU}})^{-1}$, which we show allows us to recover $\mathbf{D}^{-1} f(\mathbf{V}^*\overline{X})$ up to additive $\epsilon$ error where $\epsilon$ is at most $O\left(\frac{1}{\text{poly}(d,m,k,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*)}\right)$. Since the sketch $\mathbf{T}$ will preserve rank, $\mathbf{TU}$ will have an inverse, and thus $(\widehat{\mathbf{TU}})$ will be invertible (we can always perturbe the entries of our estimate by $1/\text{poly}(n)$ to ensure this). The inverse can then be computed in a polynomial number of arithmetic operations via Gaussian elimination. By a simple thresholding argument, we show that after rounding off the entries below $\tau = 1/\text{poly}(\ell)$ in $(\widehat{\mathbf{TU}})^{-1}\mathbf{T}\overline{\mathbf{A}}$, we in fact recover the *exact* sign pattern of $f(\mathbf{V}^*\overline{\mathbf{X}})$.

Our main insight is now that the only sparse vectors in the row space of $\overline{\mathbf{A}}$ are precisely the rows (up to positive a scaling) of $f(\mathbf{V}^*\overline{\mathbf{X}})$. Specifically, we show that the only vectors in the row

span of $\mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}})$ which have the same sign and sparsity pattern as a row of $f(\mathbf{V}^*\overline{\mathbf{X}})$ are positive scalings of the rows of $f(\mathbf{V}^*\overline{\mathbf{X}})$. Here, by *sparsity pattern*, we mean the subset of entries of a row that are non-zero. Since each row of $f(\mathbf{V}^*\mathbf{X})$ is non-negative, the sign and sparsity patterns of $f(\mathbf{V}^*\overline{\mathbf{X}})$ together specify where the non-zero entries are (which are therefore strictly positive).

Now after exact recovery of the sign pattern of $f(\mathbf{V}^*\overline{\mathbf{X}})$, we can set up a linear system to find a vector in the row span of $\mathbf{A}$ with this sign pattern, thus recovering each row of $f(\mathbf{V}^*\overline{\mathbf{X}})$ exactly. Critically, we exploit the combinatorial structure of ReLUs together with the fact that linear systems can be solved in a polynomial number of arithmetic operations. This allows for exact recovery of $\mathbf{U}^*$ thereafter. Recall that we assume the rows of $\mathbf{V}^*$ have unit length, which removes ambiguity in the positive scalings used for the rows of $\mathbf{V}*$ (and similarly the columns of $\mathbf{U}^*$).

We begin by showing that the condition number of $\mathbf{V}^*$ is inversely proportional to the minimum angle between the rows of $\mathbf{V}^*$, if they are interpreted as vectors in $\mathbb{R}^d$. This will allow us to put a lower bound on the number of disagreeing sign patterns between rows of $f(\mathbf{V}^*\overline{\mathbf{X}})$ in Lemma 31. We will then use these results to prove the uniqueness of the sign and sparsity patterns of the rows of $f(\mathbf{V}^*\overline{\mathbf{X}})$ in Lemma 33.

**Lemma 30** *Let $\theta_{\min} \in [0, \pi]$ be the smallest angle between the lines spanned by two rows of the rank $k$ matrix $\mathbf{V} \in \mathbb{R}^{k\times d}$ which unit norm rows, in other words $\theta_{\min} = \min_{i,j} \arccos(\langle \mathbf{V}_{i,*}, \mathbf{V}_{j,*}\rangle)$ where $\arccos$ takes values in the principle range $[0, \pi]$. Then $\kappa(\mathbf{V}) > \frac{c}{\theta_{\min}}$ for some constant $c$.*

**Proof** Let $i, j$ be such that $\arccos(|\langle \mathbf{V}_{i,*}, \mathbf{V}_{j,*}\rangle|) = \theta_{\min}$. Let $\mathbf{V}^-$ be the pseudo-inverse of $\mathbf{V}$. Since $\mathbf{V}$ has full row rank, it follows that $\mathbf{V}(\mathbf{V}^-)^T = I_k$, thus $\langle \mathbf{V}_{i,*}, \mathbf{V}^-_{j,*}\rangle = 0$ and $\langle \mathbf{V}_{j,*}, \mathbf{V}^-_{j,*}\rangle = 1$. The first fact implies that $\mathbf{V}^-_{j,*}$ is orthonormal to $\mathbf{V}_{i,*}$, and the second that $\cos(\theta(\mathbf{V}_{j,*}, \mathbf{V}^-_{j,*})) = (\|\mathbf{V}^-_{j,*}\|_2)^{-1}$ where $\theta(\mathbf{V}_{j,*}, \mathbf{V}^-_{j,*})$ is the angle between $\mathbf{V}_{j,*}$ and $\mathbf{V}^-_{j,*}$.

Now let $x = \mathbf{V}_{i,*}, y = \mathbf{V}^-_{j,*}/\|\mathbf{V}^-_{j,*}\|, z = \mathbf{V}_{j,*}$. Note that $x, y, z$ are all points on the unit sphere in $r$ dimensions, and since scaling does not effect the angle between two vectors, we have $\theta(x, y) = \theta(\mathbf{V}_{i,*}, \mathbf{V}^-_{j,*})$. We know $\theta(x, y) = \pi/2$, and $\theta_{\min} = \theta(x, z)$, so the law of cosines gives $\cos(\theta(y, z)) = \frac{2-\|y-z\|_2^2}{2}$. We have $\|y - z\|_2 = \|(y - x) - (z - x)\|_2 \geq |\sqrt{2} - \|z - x\|_2|$. Again by the law of cosines, we have $\|z - x\|_2^2 = 2 - 2\cos(\theta_{\min})$. Since $\cos(x) \approx 1 - \Theta(x^2)$ for small $x$ (consider the Taylor expansion), it follows that $\|z - x\|_2 \leq c'\theta_{\min}$ for some constant $c'$. So $\|y - z\|_2^2 \geq 2 - 2\sqrt{2}\|z - x\|_2 + \|z - x\|_2^2 \geq 2 - c''\theta_{\min}$ for another constant $c''$. It follows that

$$\cos(\theta(y, z)) \leq \frac{c''\theta_{\min}}{2}$$

From which we obtain $\|\mathbf{V}^-_{j,*}\|_2 \geq 2/(c''\theta_{\min})$. It follows that $\sigma_1(\mathbf{V}^-) \geq \|e_{j,*}^T\mathbf{V}^-\|_2 = \|\mathbf{V}^-_{j,*}\|_2 \geq \frac{2}{c''\theta_{\min}}$. Since the rows of $\mathbf{V}$ have unit norm, we have $\sigma_1(\mathbf{V}) \geq 1$, so $\kappa(\mathbf{V}) = \sigma_1(\mathbf{V})\sigma_1(\mathbf{V}^-) \geq \frac{2}{c''\theta_{\min}}$ which is the desired result setting $c = \frac{2}{c''}$. ■

**Lemma 31** *Fix any matrix $\mathbf{V} \in \mathbb{R}^{k\times d}$ with unit norm rows. Let $\mathbf{X} \in \mathbb{R}^{d\times\ell}$ be an i.i.d. Gaussian matrix for any $\ell \geq$ tpoly$(k, \kappa)$, where $\kappa = \kappa(\mathbf{V})$. For every pair $i, j \in [k]$ with $i \neq j$, with probability $1 - 1/$poly$(\ell)$ there are at least $t$ coordinates $p \in [\ell]$ such that $(\mathbf{VX})_{i,p} < 0$ and $(\mathbf{VX})_{j,p} > 0$.*

**Proof** We claim that $\Pr[(\mathbf{VX})_{i,p} < 0, (\mathbf{VX})_{j,p} > 0] = \Omega(1/\kappa)$. To see this, Consider the 2-dimensional subspace $H$ spanned by $\mathbf{V}_{i,*}$ and $\mathbf{V}_{j,*}$. Let $\theta$ be the angle between $\mathbf{V}_{i,*}$ and $\mathbf{V}_{j,*}$ in the plane $H$. Then the event in question is the event that a random Gaussian vector, when projected onto this plane $H$, lies between two vectors with angle $\theta$ between each other. By the rotational invariance and spherical symmetry of Gaussians (see, e.g. Bryc (2012)), this probability is $\frac{\theta}{2\pi}$. Since $\kappa(\mathbf{V}) > \frac{c}{\theta_{\min}} = \Omega(\frac{1}{\theta})$ by Lemma 30, it follows that a random gaussian splits $\mathbf{V}_{i,*}$ and $\mathbf{V}_{j,*}$ with probability $\Omega(1/\kappa)$ as desired.

Thus on each column $p$ of $f(\mathbf{VX})$, $f(\mathbf{V}_{i,*}\mathbf{X}_{*,p}) < 0$ and $f(\mathbf{V}_{j,*}\mathbf{X}_{*,p}) > 0$ with probability at least $\Omega(1/\kappa)$. Using the fact that the entries in separate columns of $\mathbf{VX}$ are independent, by Chernoff bounds, with probability greater than $1 - k^2\exp(\Omega(\ell/\kappa)) > 1 - 1/\text{poly}(\ell)$, after union bounding over all $O(k^2)$ ordered pairs $i, j$, we have that $f(\mathbf{V}_{i,*}\mathbf{X}) < 0$ and $f(\mathbf{V}_{j,*}\mathbf{X}) > 0$ on at least $\Omega(\ell/\kappa) > t$ coordinates.

■

**Lemma 32** *Let $\mathbf{Z}_i$ be the $i$-th column of $(\mathbf{VX})$, where $\mathbf{V}$ has rank $k$. Then the covariance of the coordinates of $\mathbf{Z}_i$ are given by the $k \times k$ posiitve definite covariance matrix $\mathbf{VV}^T$, and the joint density function is given by:*

$$p(\mathbf{Z}_{i,1}, \ldots, \mathbf{Z}_{i,k}) = \frac{\exp\left(-\frac{1}{2}\mathbf{Z}_i^T(\mathbf{VV}^T)^{-1}\mathbf{Z}_i\right)}{\sqrt{(2\pi)^k \det(\mathbf{VV}^T)}}$$

*In particular, the joint probability density of any subset of entries of $\mathbf{VX}$ is smooth and everywhere non-zero.*

**Proof** Since $\mathbf{Z}_i = \mathbf{V}(\mathbf{X}_i^T)^T$, where $\mathbf{X}_i^T$ are i.i.d. normal random variables, t is well known that the covariance is given by $\mathbf{VV}^T$ Gut (2009), which is positive definite since $\mathbf{V}$ has full row rank. These are sufficient conditions (Ash) for the pdf to be given in the form as stated in the Proposition. Since distinct columns of $\mathbf{VX}$ are statistically independent (as they are generated by separate columns of $\mathbf{X}$), the last statement of the proposition follows.

■

The following Lemma demonstrates that the the only vectors in the row span of $f(\mathbf{V}^*\mathbf{X})$ with the same sign and sparsity pattern as $f(\mathbf{V}^*\mathbf{X})_{i,*}$, for any given row $i$, are positive scalings of $f(\mathbf{V}^*\mathbf{X})_{i,*}$. Recall that a sparsity pattern $S \subseteq [n]$ of a vector $y \in \mathbb{R}^n$ is just set of coordinates $i \in S$ such that $y_i > 0$.

**Lemma 33** *Let $\mathbf{X} \in \mathbb{R}^{d\times\ell}$ be an i.i.d. Gaussian matrix for any $\ell > t\text{poly}(k, \kappa(\mathbf{V}^*))$. Let $S$ be the sparsity pattern of a fixed row $f(\mathbf{V}^*\mathbf{X})_{i,*}$, and let $\emptyset \subsetneq S' \subseteq S$. Then w.h.p. (in $t$), the only vectors in the row span of $f(\mathbf{V}^*\mathbf{X})$ with sparsity pattern $S'$, if any exist, are non-zero scalar multiples of $f(\mathbf{V}^*\mathbf{X})_{i,*}$.*

**Proof** Suppose $\mathbf{Z} = wf(\mathbf{V}^*\mathbf{X})$ had sparsity pattern $S'$ and was not a scaling of $f(\mathbf{V}^*\mathbf{X})_{i,*}$. Then $w$ is not 1-sparse, since otherwise it would be a scaling of a another row of $f(\mathbf{V}^*\mathbf{X})$, and by Proposition 31 no row's sparsity pattern is contained within any other row's sparsity pattern. Let $\mathbf{W}$ be $f(\mathbf{V}^*\mathbf{X})$ restricted to the rows corresponding to the non-zero coordinates in $w$, and write

$\mathbf{Z} = w\mathbf{W}$ (where now $w$ has also been restricted to the appropriate coordinates). Since $\mathbf{W}$ has at least 2 rows, and since the sparsity pattern of $w\mathbf{W}$ is contained within the sparsity pattern of $f(\mathbf{V}^*\mathbf{X})_{i,*}$, by Proposition 31, taking $t = 10k^2$, we know that there are at least $10k^2$ non-zero columns of $\mathbf{W}$ for which $w\mathbf{W}$ is 0, so let $\mathbf{W}'$ be the submatrix of all such columns.

Now for each row $\mathbf{W}'_i$ of $\mathbf{W}'$ with less than $k$ non-zero entries, remove this row $\mathbf{W}'_i$ and also remove all columns of $\mathbf{W}'$ where $\mathbf{W}'_i$ was non-zero. Continue to do this removal iteratively until we obtain a new matrix $\mathbf{W}''$ where now every row has at least $k$ non-zero entries. Observe that the resulting matrix $\mathbf{W}''$ has at least $9k^2$ columns. If there are no rows left, then since we only removed $k$ columns for every row removed, this means there were at least $9k^2$ columns of $\mathbf{W}'$ which contained only zeros, which is a contradiction since by construction the columns of $\mathbf{W}'$ were non-zero to begin with. So, let $k' \leq k$ be the number of rows remaining in $\mathbf{W}''$. Note that since the rows we removed were zero on the columns remaining in $\mathbf{W}''$, there must still be a vector $w'$, which in particular is $w$ restricted to the rows of $\mathbf{W}''$, which has no zero-valued entries and such that $w'\mathbf{W}'' = 0$.

Now observe once we obtain this matrix $\mathbf{W}''$, note that we have only conditioned on the sparsity pattern of the entries of $\mathbf{W}''$ (over the randomness of the Gaussians $\mathbf{X}$), but we have not conditioned on the values of the non-zero entries of $\mathbf{W}''$. Note that this conditioning does not change the continuity of the joint distributions of the columns of $\mathbf{W}''$, since this conditioning is simply restricting the columns to the non-zero intersection of half spaces which define this sign pattern. Since the joint density function of the columns of $\mathbf{VX}$ is non-zero on all of $\mathbb{R}^k$ by Lemma 32, it follows that, after conditioning, any open set in this intersection of half spaces which defines the sparsity pattern of $\mathbf{W}''$ has non-zero probability measure with respects to the joint density function.

Given this, the argument now proceeds as in Lemma 16. Since each row of $\mathbf{W}''$ has at least $k$ non-zero entries, we can find a square matrix $\mathbf{W}^\dagger \in \mathbb{R}^{k' \times k'}$ obtained by a taking a subset of $k' < 9k^2$ columns of $\mathbf{W}''$ and permuting them such that the diagonal of $\mathbf{W}^\dagger$ has a non-zero sign pattern. After conditioning on the sign pattern so that the diagonal is non-zero, the determinant $\det(\mathbf{W}^\dagger)$ of $\mathbf{W}^\dagger$ is a non-zero polynomial in $s$ random variables with $k' \leq s \leq (k')^2$. By Lemma 32, the joint density function of these $s$ variables is absolutely continuous and everywhere non-zero on the domain. Here the domain $\Omega$ is the intersection of half spaces given by the sign pattern conditioning.

Since $\Omega$ is non-empty, it has unbounded Lebesgue measure in $\mathbb{R}^s$. Since $\det(\mathbf{W}^\dagger)$ is a non-zero polynomial in $s$ real variables, it is well known that $\det(\mathbf{W}^\dagger)$ cannot vanish on any non-empty open set in $\mathbb{R}^s$ (see e.g. Theorem 2.6 of Conrad, and note the subsequent remark on replacing $\mathbb{C}^s$ with $\mathbb{R}^s$). It follows that the set of zeros of $\det(\mathbf{W}^\dagger)$ contain no open set of $\mathbb{R}^s$, and thus has Lesbegue measure 0 in $\Omega$. Integrating the joint pdf of the $s$ random variables over this subset of measure 0, we conclude that the probability that the realization of the random variables is in this set is 0. So the matrix $\mathbf{W}''$ has rank $k'$, and so $w'\mathbf{W}'' = 0$ is impossible, a contradiction. It follows that $\mathbf{Z}$ is a scaling of a row of $f(\mathbf{V}^*\mathbf{X})$ as needed. ■

We will now need the following perturbation bounds for the pseudo-inverse of matrices.

**Proposition 34 (Theorem 1.1 Meng and Zheng (2010))** *Let $\mathbf{B}^\dagger$ denote the MoorePenrose Pseudo-inverse of $\mathbf{B}$, and let $\|\mathbf{B}\|_2$ denote the operator norm of $\mathbf{B}$. Then for any $\mathbf{E}$ we have*

$$\|(\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger\|_F \leq \sqrt{2}\max\left\{\|\mathbf{B}^\dagger\|_2^2, \|(\mathbf{B}+\mathbf{E})^\dagger\|_2^2\right\}\|\mathbf{E}\|_F$$

We prove the following corollary which will be useful to us.

**Corollary 35** *For any* $\mathbf{B}, \mathbf{E}$ *and* $\frac{1}{4} \geq \epsilon > 0$ *with* $\|\mathbf{B}\|_2 \geq 1$, $\|\mathbf{E}\|_F \leq \frac{\epsilon}{\kappa^2}$ *and where* $\kappa = \kappa(\mathbf{B})$ *is the condition number of* $\mathbf{B}$. *Then we have*

$$\|(\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger\|_F \leq O(\epsilon)$$

*and moreover, if* $\mathbf{B}$ *has full column rank, then*

$$\|(\mathbf{B}+\mathbf{E})^\dagger\mathbf{B} - \mathbb{I}\|_F \leq O(\|\mathbf{B}\|_2\epsilon)$$

**Proof** We have $\|(\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger\|_F \leq \max\left\{\|\mathbf{B}^\dagger\|_2^2, \|(\mathbf{B}+\mathbf{E})^\dagger\|_2^2\right\}\frac{O(\epsilon)}{\kappa^2}$ by applying Proposition 34. In the first case, this is at most $\frac{1}{\sigma_{\min}^2(\mathbf{B})}\frac{O(\epsilon)}{\kappa^2} = O(\epsilon)$ as stated. Here we used the fact that $\|\mathbf{B}\|_2 = \sigma_{\max}(\mathbf{B}) \geq 1$, so $1/\sigma_{\min}(\mathbf{B}) \leq \kappa$. In the second case of the max, we have $\|(\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger\|_F \leq \|(\mathbf{B}+\mathbf{E})^\dagger\|_2^2\frac{O(\epsilon)}{\kappa^2} = \sigma_{\min}^{-2}(\mathbf{B}+\mathbf{E})\frac{O(\epsilon)}{\kappa^2}$. By the Courant-Fisher theorem [2], using that $\|\mathbf{E}\|_2 \leq \|\mathbf{E}\|_F \leq 1/(4\kappa)$, we have

$$\sigma_{\min}(\mathbf{B}+\mathbf{E}) \geq \inf_{x:\|x\|_2=1}\|x(\mathbf{B}+\mathbf{E})\|_2 \geq \inf_{x:\|x\|_2=1}\big|\,\|x\mathbf{B}\|_2 - \|x\mathbf{E}\|_2\,\big|$$

$$\geq \sigma_{\min}(\mathbf{B}) - \frac{1}{4\kappa} \geq \sigma_{\min}(\mathbf{B})/2 \geq 1/(2\kappa)$$

where the minimum is taken over vectors $x$ with the appropriate dimensions. Thus in both cases, we have $\|(\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger\|_F \leq O(\epsilon)$, so

$$\|(\mathbf{B}+\mathbf{E})^\dagger\mathbf{B} - \mathbb{I}\|_F = \|((\mathbf{B}+\mathbf{E})^\dagger - \mathbf{B}^\dagger)\mathbf{B}\|_F \leq \|\mathbf{B}\|_2 O(\epsilon)$$

■

We now are ready to complete the proof of the correctness of Algorithm 4

**Theorem 36** *(Exact Recovery for Orthonormal* $\mathbf{V}^*$*.) Given* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$*, for rank* $k$*-matrices* $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ *where* $\mathbf{V}^*$ *is orthonormal and* $\mathbf{X} \in \mathbb{R}^{d\times n}$ *which is i.i.d. Gaussian with* $n = \mathit{poly}(d, k, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))$*, there is a* $\mathit{poly}(n)$*-time algorithm which recovers* $\mathbf{U}^*, \mathbf{V}^*$ *exactly with probability* $1 - \frac{1}{\mathit{poly}(d,m,k)}$.

**Proof** By Corollary 29, after sketching $\mathbf{A}$ by a Gaussian matrix $\mathbf{T} \in \mathbb{R}^{k\times m}$ and running ICA on $\mathbf{TA}$ in $\mathrm{poly}(d, m, k, \kappa(\mathbf{U}^*))$ time, we recover $\widehat{\mathbf{TU}^*}$ such that $\|\widehat{\mathbf{TU}} - \mathbf{TU}^*\mathbf{\Pi}\mathbf{D}\|_F \leq \frac{1}{\mathrm{poly}(d,k,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))}$ for a sufficiently high constant-degree polynomial, such that $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{D}$ is strictly positive diagonal matrix. We can disregard $\mathbf{\Pi}$ by assuming the rows of $\mathbf{V}^*$ have also been permuted by $\mathbf{\Pi}$, and we can disregard $\mathbf{D}$ by pulling this scaling into $\mathbf{V}^*$ (which can be done since it is a positive scaling). Thus $\|\widehat{\mathbf{TU}} - \mathbf{TU}^*\|_F \leq \frac{1}{\mathrm{poly}(d,k,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))}$

Observe now that we can assume that $1 \leq \|\mathbf{TU}^*\|_2 \leq 2$ by guessing a scaling factor $c$ to apply to $\mathbf{A}$ before running ICA. To guess this scaling factor, we can find the largest column (in $L_2$) $y$ of $\mathbf{TA}$, and note that $y = (\mathbf{TU}^*)f(\mathbf{V}^*\mathbf{X}_{*,j})$ for some $j$. Since $\|f(\mathbf{V}^*\mathbf{X}_{*,j})\|_2 \leq O(\sqrt{\log(n)})d$

2. See https://en.wikipedia.org/wiki/Min-max_theorem

with high probability for all $j \in [n]$ (using the Gaussian tails of $\mathbf{X}$), it follows that $\|y\|_2 \leq \sigma_{\max}(\mathbf{T}\mathbf{U}^*)O(\sqrt{\log(n)})d$. Since with w.h.p there is at least one column of $f(\mathbf{V}^*\mathbf{X})$ with norm at least $1/\text{poly}(n)$, it follows that $\|y\|_2 \geq \sigma_{\min}(\mathbf{T}\mathbf{U}^*)/\text{poly}(n) \geq \frac{\sigma_{\max}(\mathbf{T}\mathbf{U}^*)}{\text{poly}(n,\kappa)}$. Thus one can make $\log\big(\text{poly}(n,\kappa,d)\big) = O(\log(n))$ guesses in geometrically increasing powers of 2 between $\|y\|_2/O(\sqrt{\log(n)})d$ and $\|y\|_2\text{poly}(n,\kappa)$ to find a guess such that $\|c\mathbf{T}\mathbf{U}^*\|_2 \in (1,2)$ as desired. This will allow us to use Corollary 35 in the following paragraph.

Now let $\widehat{\mathbf{T}\mathbf{U}}^\dagger$ be the pseduo-inverse of $\widehat{\mathbf{T}\mathbf{U}}$, and let $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}})$ where $\overline{\mathbf{X}}$ is the first $\text{poly}(d,k,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))$ columns of $\mathbf{X}$. We now claim that the sign pattern of $(\widehat{\mathbf{T}\mathbf{U}})^\dagger\mathbf{T}\overline{\mathbf{A}} = \widehat{\mathbf{T}\mathbf{U}}^\dagger\mathbf{T}\mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}})$ is exactly equal to that of $f(\mathbf{V}^*\overline{\mathbf{X}})$ after rounding all entries of with value less than $1/\text{poly}(\ell)$ to 0. Note that since $\mathbf{T}\mathbf{U}^*$ is full rank, it has an inverse (which is given by the pseudoinverse $(\mathbf{T}\mathbf{U}^*)^\dagger$. Let $\mathbf{Z}$ be the resulting matrix after rounding performing this rounding to $\widehat{\mathbf{T}\mathbf{U}}^\dagger\mathbf{T}\mathbf{A}'$. We now apply Corollary 35, with $\mathbf{T}\mathbf{U}^* = \mathbf{B}$ and $\widehat{\mathbf{T}\mathbf{U}} = \mathbf{B} + \mathbf{E}$. Since we guesses $\sigma_{\max}(\mathbf{T}\mathbf{U}^*)$ up to a factor of 2 and normalized $\widehat{\mathbf{T}\mathbf{U}}$ by it, it follows that the entries of the diagonal matrix $\mathbf{D}$ are all at most 2 and at least $1/(2\kappa(\mathbf{T}\mathbf{U}^*))$, and then using the fact that $\|f(\mathbf{V}^*\overline{\mathbf{X}})\|_F < \|\mathbf{V}^*\overline{\mathbf{X}}\|_F \leq \sqrt{\ell}\|V^*\|_F \leq \sqrt{\ell k}$ w.h.p. in $\ell$ (using well-known upper bounds on the spectral norm of a rectangular Gaussian matrix, see e.g. Corollary 5.35 if Vershynin (2010)) we obtain

$$\begin{aligned}\|\mathbf{Z} - \mathbf{D}f(\mathbf{V}^*\overline{\mathbf{X}})\|_F &= \|\big(\widehat{\mathbf{T}\mathbf{U}}^\dagger(\mathbf{T}\mathbf{U}^*) - \mathbb{I}\big)\mathbf{D}f(\mathbf{V}^*\mathbf{X}')\|_F \\ &\leq \frac{1}{\text{poly}\left(d,k,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*)\right)}\end{aligned}$$

Note that algorithmically, instead of computing the inverse $\widehat{\mathbf{T}\mathbf{U}}^\dagger$, we can first randomly perturb $\widehat{\mathbf{T}\mathbf{U}}$ by an entry-wise additive $1/\text{poly}(n)$ to ensure it is full rank, and then compute the true inverse, which can be done via Gaussian elimination in polynomially many arithmetic operations. By the same perturbational bounds, our results do not change when using the $1/\text{poly}(n)$ perturbed inverse, as opposed to the original pseudo-inverse.

Now since the positive entries of $\mathbf{D}f(\mathbf{V}^*\overline{\mathbf{X}})$ have normal Gaussian marginals, and $\mathbf{D}$ is a diagonal matrix which is entry-wise at most 2 and at least $1/(2\kappa(\mathbf{T}\mathbf{U}^*))$, the probability that any non-zero entry of $f(\mathbf{V}^*\overline{\mathbf{X}})$ is less than $1/\text{poly}(\ell)$ is at most $2\kappa(\mathbf{T}\mathbf{U}^*)/\text{poly}(\ell)$, and we can then union bound over $\text{poly}(d,k,m,\kappa)$ such entries in $\overline{\mathbf{X}}$. Note that by Lemma 28, $\kappa(\mathbf{T}\mathbf{U}^*) < \text{poly}(k,d,m)\kappa(\mathbf{U}^*)$ w.h.p. in $k,d,m$, so $\text{poly}(\ell) >> \kappa(\mathbf{T}\mathbf{U}^*)$. Conditioned on this, with probability $1 - 1/\text{poly}(d,m,k,\kappa)$ for sufficiently large $\ell = \text{poly}(d,k,m,\kappa)$, every strictly positive entry of $\mathbf{D}f(\mathbf{V}^*\overline{\mathbf{X}})$, and therefore of $f(\mathbf{V}^*\overline{\mathbf{X}})$, is non-zero in $\mathbf{Z}$, and moreover, and every other entry will be 0 in $\mathbf{Z}$, which completes the claim that the sign and sparsity patterns of the two matrices are equal.

Given this, for each $i \in [k]$ we can then solve a linear system to find a vector $w_j$ such that $(w_j\overline{\mathbf{A}})_p = 0$ for all $p$ not in the sparsity pattern of $\mathbf{Z}_{i,*}$. In other words, the sparsity pattern of $(w_j\overline{\mathbf{A}})$ must be contained in the sparsity pattern of $\mathbf{Z}_{i,*}$, which is the sparsity pattern of $f(\mathbf{V}^*\overline{\mathbf{X}})_{i,*}$ be the prior argument. By Lemma 33, the only vector in the row span of $\overline{\mathbf{A}}$ (which is the same as the row span of $f(\mathbf{V}^*\overline{\mathbf{X}})$ since $\mathbf{U}^*$ is full rank) which has a non-zero sparsity pattern contained in that of $f(\mathbf{V}^*\overline{\mathbf{X}})_{i,*}$ must be a non-zero scaling of $f(\mathbf{V}^*\overline{\mathbf{X}})_{i,*}$. It follows that there is a unique $w_j$, up to a scaling, such that $w_j\overline{\mathbf{A}}$ is zero outside of the sparsity pattern of $f(\mathbf{V}^*\overline{\mathbf{X}})_{i,*}$. Since at least one of the entries $r$ of $w_j$ is non-zero, there exists some scaling such that $w_j\mathbf{A}$ is zero outside of the sparsity pattern of $f(\mathbf{V}^*\overline{\mathbf{X}})_{i,*}$ *and* $(w_j)_r = 1$ (where $(w_j)_r$) is the $r$-th coordinate of $w_j$). Since the first constraint is satisfied uniquely up to a scaling, it follows that there will be a unique solution

$w_j^r$ to at least one of the $r \in [k]$ linear systems in Step 5 of Algorithm 4, which will therefore be optained by the linear system. This vector $w_j$ we obtain from Steps 5 and 6 of Algorithm 4 will therefore be such that $w_j \overline{\mathbf{A}}$ is a non-zero scaling of $f(\mathbf{V}^* \overline{\mathbf{X}})_{i,*}$.

Then in Step 7 of Algorithm 4, we construct the matrix $\mathbf{W}$, and flip the signs appropriately so that each row of $\mathbf{W}$ is a strictly positive scaling of a row of $f(\mathbf{V}^* \overline{\mathbf{X}})$. We then solve the linear system $(\mathbf{W}_{i,*})_{S_i} = \mathbf{V}_{i,*} \overline{\mathbf{X}}_{S_i}$ for the unknowns $\mathbf{V}$, which can be done with a polynomial number of arithmetic operations via Gaussian elimination. Recall here that $S_i$ is the set of coordinates where $\mathbf{W}_{i,*}$, and therefore $f(\mathbf{V}^*_{i,*} \overline{\mathbf{X}})$, is non-zero. Since at least $1/3$ of the signs in a given row $i$ will be positive with probability $1 - 2^{-\Omega(\ell)}$ by Chernoff bounds, restricting to this subset $S_i$ of columns gives the equation $\mathbf{W}_{i,*} = \mathbf{V}^*_{i,*} \overline{\mathbf{X}}_{S_i}$. Conditioned on $S_i$ having at least $d$ columns, we have that $\overline{\mathbf{X}}_{S_i}$ is full rank almost surely, since it is a matrix of Gaussians conditioned on the fact that every column lies in a fixed halfspace. To see this, apply induction on the columns of $\overline{\mathbf{X}}_{S_i'}$, and note at every step $i < d$, the Lesbegue measure of the span of the first $i$ columns is 0 in this halfspace, and thus the $i+1$ column will not be contained in it almost surely. It follows that there is a unique solution $\mathbf{V}_{i,*}$ for each row $i$, which must therefore be the corresponding row of $\mathbf{V}^*$ (we normalize the rows of $\mathbf{V}_{i,*}$ to have unit norm so that they are precisely the same). So we recover $\mathbf{V}^*$ exactly via these linear systems. Finally, we can solve the linear system $\mathbf{A} = \mathbf{U} f(\mathbf{V}^* \mathbf{X})$ for the variables $\mathbf{U}$ to recover $\mathbf{U}^*$ exactly in strongly polynomial time. Note that this linear system has a unique solution, since $f(\mathbf{V}^* \mathbf{X})$ is full rank w.h.p. by Lemma 16, which completes the proof. ■

### C.2. General Algorithm

We now show how to generalize the algorithm from the previous sub-section to handle non-orthonormal $\mathbf{V}^*$. Observe that when $\mathbf{V}^*$ is no longer orthonormal, the entries within a column of $\mathbf{V}^* \mathbf{X}$ are no longer independent. Moreover, due to the presence of the non-linear function $f(\cdot)$, no linear transformation will exist which can make the samples (i.e. columns of $f(\mathbf{V}^* \mathbf{X})$) independent entry-wise. While the entries do still have Gaussian marginals, they will have the non-trivial covariance matrix $\mathbf{V}^* (\mathbf{V}^*)^T \neq \mathbb{I}_k$. Thus it is no longer possible to utilize previously developed techniques from independent component analysis to recover good approximations to $\mathbf{U}^*$. This necessitates a new approach.

Our starting point is the generative model considered by Janzamin et. al. Janzamin et al. (2015), which matches our setting, i.e. $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$. The main idea behind this algorithm is to construct a tensor that is a function of both $\mathbf{A}, \mathbf{X}$ and then run a tensor decomposition algorithm to recover the low-rank components of the resulting tensor. While computing a tensor decomposition is NP-hard in general Hillar and Lim (2013), there is a plethora of work on special cases, where computing such decompositions is tractable Bhaskara et al. (2014); Song et al. (2016); Wang and Anandkumar (2016); Goyal et al. (2014); Ge and Ma (2015); Barak and Moitra (2016). Tensor decomposition algorithms have recently become an invaluable algorithmic primitive and found a tremendous number of applications in statistical and machine learning tasks Janzamin et al. (2015, 2014); Ge et al. (2017); Anandkumar et al. (2014); Barak et al. (2015).

A key step is to construct a non-linear transform of the input by utilizing knowledge about the underlying pdf for the distribution of $\mathbf{X}$, which we denote by $p(x)$. The non-linear function considered is the so called Score Function, defined in Janzamin et al. (2014), which is the normalized $m$-th order derivative of the input probability distribution function $p(x)$.

**Definition 37** *(Score Function.) Given a random vector $x \in \mathbb{R}^d$ such that $p(x)$ describes the corresponding probability density function, the $m$-th order score function $\mathcal{S}_m(x) \in \otimes^m \mathbb{R}^d$ is defined as*

$$\mathcal{S}_m(x) = (-1)^m \frac{\nabla_x^{(m)} p(x)}{p(x)}$$

The tensor that Janzamin et. al. Janzamin et al. (2014) considers is the cross moment tensor between $\mathbf{A}$ and $\mathcal{S}_3(\mathbf{X})$. This encodes the correlation between the output and the third order score function. Intuitively, working with higher order tensors is necessary since matrix decompositions are only identifiable up to orthogonal components, whereas tensor have identifiable non-orthogonal components, and we are specifically interested in recovering approximations for non-orthonormal $\mathbf{V}^*$. Computing the score function for an arbitrary distribution can be computationally challenging. However, as mentioned in Janzamin et. al. Janzamin et al. (2014), we can use orthogonal polynomials that help us compute the closed form for the score function $\mathcal{S}_{(m)}(x)$, in the special case when $x \sim \mathcal{N}(0, \mathbb{I})$.

**Definition 38** *(Hermite Polynomials.) If the input is drawn from the multi-variate Gaussian distribution, i.e. $x \sim \mathcal{N}(0, \mathbb{I})$, then $\mathcal{S}_{(m)}(x) = \mathcal{H}_m(x)$, where $H_m(x) = \frac{(-1)^m \nabla_x^{(m)} p(x)}{p(x)}$ and $p(x) = \frac{1}{(\sqrt{2\pi})^d} e^{-\frac{\|x\|_2^2}{2}}$.*

Since we know a closed form for the $m$-th order Hermite polynomial, the tensor $\mathcal{S}_{(m)}$ can be computed efficiently. The critical structural result in the algorithm of Janzamin et al. (2015) is to show that in expectation, the cross moment of the output and the score function actually forms a rank-$k$ tensor, where the rank-1 components capture the rows of $\mathbf{V}^*$. Formally,

**Lemma 39** *(Generalized Stein's Lemma Janzamin et al. (2015).) Let $\mathbf{A}, \mathbf{X}$ be input matrices such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$, where $f$ is a non-linear, thrice differentiable activation function. Let $\mathcal{S}_3(x)$ be the 3-rd order score function from Definition 37. Then,*

$$\widetilde{\mathbf{T}} = \mathbb{E}\left[\sum_{i=1}^{n} \mathbf{A}_{*,i} \otimes \mathcal{S}_3(\mathbf{X}_{*,i})\right] = \sum_{j=1}^{k} \mathbb{E}_x\left[f'''(\mathbf{V}^* x)\right] \mathbf{U}^*_{*,j} \otimes \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*}$$

*where $f'''$ is the third derivative of the activation function and $x \sim p(x)$.*

Note, $\widetilde{\mathbf{T}}$ is a 4-th order tensor and can be constructed from the input $\mathbf{A}$ and $\mathbf{X}$. The first mode of $\widetilde{\mathbf{T}}$ can be contracted by multiplying it with a random vector $\theta$, therefore,

$$\mathbb{E}\left[\sum_{i=1}^{n} \mathbf{A}_{*,i} \otimes \mathcal{S}_3(\mathbf{X}_{*,i})\right] = \sum_{j=1}^{k} \lambda_j \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*}$$

where $\lambda_j = \mathbb{E}_x\left[f'''(\mathbf{V}^* x)\right] \langle \mathbf{U}^*_{*,j}, \theta \rangle$. Therefore, if we could recover the low-rank components of $\widetilde{\mathbf{T}}$ we would be obtain a approximate solution to $\mathbf{V}^*$. The main theorem in Janzamin et al. (2015) states that under a set of conditions listed below, there exists a polynomial time algorithm that recovers an additive error approximation to $\mathbf{V}^*$. Formally,

**Theorem 40** *(Approximate recovery Janzamin et al. (2015)) Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{X} \in \mathbb{R}^{d\times n}$ be inputs such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \eta$, where $f$ is a non-linear thrice differentiable activation function, $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ has full column rank, $\mathbf{V}^* \in \mathbb{R}^{k\times d}$ has full row rank, for all $i \in [n]$, $\mathbf{X}_{*,i} \sim \mathcal{N}(0, \mathbb{I})$ and $\eta$ is mean zero sub-Gaussian noise with variance $\sigma_{noise}$. Then, there exists an algorithm that recovers $\widehat{\mathbf{V}}$ such that $\|\widehat{\mathbf{V}} - \mathbf{D}\mathbf{\Pi}\mathbf{V}^*\|_F \leq \epsilon$, where $\mathbf{D}$ is a diagonal $\pm 1$ matrix and $\mathbf{\Pi}$ is a permutation matrix. Further, the algorithm runs in time*

$$poly\left(m, d, k, \frac{1}{\epsilon}, \mathbb{E}\left[\|\mathbf{M}_3(x)\mathbf{M}_3(x)^T\|_2\right], \mathbb{E}\left[\|\mathcal{S}_2(x)\mathcal{S}_2(x)^T\|_2\right], \frac{1}{\lambda_{\min}}, \lambda_{\max}, \frac{\widetilde{\lambda}_{\max}}{\widetilde{\lambda}_{\min}}, \kappa(\mathbf{V}^*), \sigma_{noise}\right)$$

*where $\mathcal{S}_3$ is the 3-rd order score function, $\mathbf{M}_3(x)\mathbb{R}^{d\times d^2}$ is the matricization of $\mathcal{S}_3$, $\lambda_j = \mathbb{E}_x\left[f'''(\mathbf{V}^*x)\right]\langle \mathbf{U}^*_{*,j}, \theta\rangle$, $\widetilde{\lambda}_j = \mathbb{E}_x\left[f''(\mathbf{V}^*x)\right]\langle \mathbf{U}^*_{*,j}, \theta\rangle$ ,$\kappa(\mathbf{V}^*)$ is the condition number, $\sigma_{noise}$ is the variance of $\eta$ and. Note, in the case where $\mathbf{X}_{*,i} \sim \mathcal{N}(0, \mathbb{I})$, $\mathbb{E}\left[\|\mathbf{M}_3(x)\mathbf{M}_3(x)^T\|_2\right] = O(d^3)$ and $\mathbb{E}\left[\|\mathcal{S}_2(x)\mathcal{S}_2(x)^T\|_2\right] = O(d^2)$.*

**Remark 41** *We only use the Whitening, Tensor Decomposition and Unwhitening steps from Janzamin et. al. Janzamin et al. (2015), and therefore the sample complexity and running time only depends on Lemma 9 and Lemma 10 in Janzamin et al. (2015).*

However, there are many technical challenges in extending the aforementioned result to our setting. We begin with using the estimator from Theorem 40 in the setting where the noise, $\eta$, is 0. The first technical challenge is the above theorem requires the activation function $f$ to be thrice diffrentiable, however ReLU is not. To get around this, we use a result from approximation theory to show that ReLU can be well approximated every where with a low-degree polynomial.

**Lemma 42** *(Approximating ReLU Goel and Klivans (2017).) Let $f(x) = \max(0, x)$ be the ReLU function. Then, there exists a polynomial $p(x)$ such that*

$$\sup_{x\in[-1,1]} |f(x) - p(x)| \leq \eta$$

*and $deg(p) = O(\frac{1}{\eta})$ and $p([-1, 1]) \subseteq [0, 1]$.*

This polynomial is at least thrice differentiable and can be easily extended to the domain we care about using simple transformations. We assume that the samples we observe are of the form $\mathbf{U}^*p(\mathbf{V}^*\mathbf{X})$ corrupted by small adversarial error. Formally, the label matrix $\mathbf{A}$ can be viewed as being generated via $\mathbf{A} = \mathbf{U}^*p(\mathbf{V}^*\mathbf{X}) + \mathbf{Z}$, where $\mathbf{Z} = \mathbf{U}^*\big(f(\mathbf{V}^*\mathbf{X}) - p(\mathbf{V}^*\mathbf{X})\big)$. We note that we only use the approximation as an analysis technique and show that we can get an approximate solution to $\mathbf{V}^*$. First, we make a brief remark regarding the normalization of the entries in $\mathbf{A}$.

**Remark 43** *Observe in both the noiseless and noisy cases, the latter being where $\mathbf{A} = \mathbf{U}^*f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ where $\mathbf{E}$ is i.i.d. mean 0 with variance $\sigma^2$, that by scaling $\mathbf{A}$ by $1/\|\mathbf{A}_{*,\max}\|_2$, where $\|\mathbf{A}_{*,\max}\|_2$ is the largest column norm of $\mathbf{A}$, we can ensure that the resulting $\mathbf{U}^*$ has $\|\mathbf{U}^*\|_2 < m\max\{1, \sigma\}\kappa(\mathbf{U}^*)$, where $\sigma^2$ is the variance of the noise $\mathbf{E}$ (in the noisy case). To see why this is true, suppose this were not the case. Observe that w.h.p. at least half of the columns $\mathbf{U}^*f(\mathbf{V}^*\mathbf{X})$ which will have norm at least $\omega(1)\sigma^{-1}_{\min}(\mathbf{U}^*)$ (since w.h.p. half the columns of $f(\mathbf{V}^*\mathbf{X})$ have norm $\omega(1)$), thus*

*if* $\|\mathbf{U}^*\|_2 > m\max\{1,\sigma\}\kappa(\mathbf{U}^*)$ *after normalization, then then at least half of the normalized columns of* $\mathbf{U}f(\mathbf{V}^*\mathbf{X})$ *will have norm* $\omega(m\max\{1,\sigma\})$*. By Markov inequality and a Chernoff bound, strictly less than* $1/4$ *of the columns of the original* $\mathbf{E}$ *can have norm* $\omega(m\sigma)$ *w.h.p., and since the normalized* $\mathbf{E}$ *is strictly smaller, by triangle inequality there will be a column of* $\mathbf{A} = \mathbf{U}^*f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ *after normalization with larger than unit norm, a contradiction. Thus we can assume this normalization, giving* $\eta << \frac{1}{\|\mathbf{U}^*\|_2}$ *for sufficiently small* $\eta = O(\frac{1}{poly(n,d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*),\sigma)})$.

We now set $\eta$ in Lemma 42 to be $\frac{1}{\text{poly}(n,d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*),\sigma)}$. By the operator norm bound of Lemma 47, we know that $\|\mathbf{V}^*\mathbf{X}\|_F = O(\sqrt{nk})$, w.h.p., so $\|\mathbf{Z}\|_F = O(\|\mathbf{U}^*\|_2\sqrt{nk}\eta) = O(\frac{1}{\text{poly}(n)})$ as needed. We again construct the same tensor, $\widetilde{\mathbf{T}} = \mathbb{E}\left[\sum_{i=1}^n \mathbf{A}_{*,i} \otimes \mathcal{S}_3(\mathbf{X}_{*,i})\right]$. Our analysis technique is now as follows. We add a light $\mathcal{N}(0,1)$ random matrix to our input $\mathbf{A}$, and argue that the variation distance between the distribution over inputs $\mathbf{A}$ (for a fixed $\mathbf{X}$), between the case of $\mathbf{A}$ using $f$ and $\mathbf{A}$ using the polynomial $p$ as a non-linear activation, is at most $1/\text{poly}(n)$. As a result, the input using ReLUs is statistically indistinguishable in variation distance from samples generated using the polynomial approximation to the ReLU function. Thus, any algorithm that succeeds on such a polynomial approximation must also succeed on the ReLU. Therefore, the algorithm from Theorem 40 still holds for approximate recovery using ReLUs. Formally,

**Lemma 44** *The variational distance between* $n$ *samples of the form* $\mathbf{A} = \mathbf{U}^*f(\mathbf{V}^*\mathbf{X}) + \mathbf{G}$*, where the columns of* $\mathbf{G}$ *are* $\mathcal{N}(0,\mathbb{I}_d)$ *and* $\mathbf{X}$ *is fixed, and* $\mathbf{A}' = \mathbf{U}^*p(\mathbf{V}^*\mathbf{X}) + \mathbf{G} + \mathbf{Z}$ *where* $\|\mathbf{Z}\|_F = \frac{1}{poly(n)}$ *is at most* $\frac{1}{poly(n)}$.

**Proof** Given two independent Gaussian $\mathcal{N}(\mu_1,\mathbb{I}), \mathcal{N}(\mu_2,\mathbb{I})$, a standard result in probability theory is that their variations distance is $\Theta(\|\mu_1 - \mu_2\|_2)$ DasGupta (2008). Thus the variation distance between the $i$-th column of $\mathbf{A}$ and $\mathbf{A}'$ is $O(\|\mathbf{Z}_{*,i}\|_2)$. Since the columns of the input are independent, the overall distribution is a product distribution so the variation distance adds. Thus the total variation distance is at most $O(\|\mathbf{Z}_{i,*}\|_F^2) = \frac{1}{\text{poly}(n)}$ as needed. ■

It follows from the above lemma that the algorithm corresponding to Theorem 40 cannot distinguish between receiving samples from the ReLU distribution with artificially added Gaussian noise or the samples from the polynomial approximation with small adversarial noise. Therefore, the algorithm recovers an approximation to the underlying weight matrix $\mathbf{V}^*$ in polynomial time. Formally, if we have an algorithm which can solve a class of problems coming from a distribution $\mathcal{D}$ with failure probability at most $\delta$, then it can solve problems coming a distribution $\mathcal{D}'$ with failure probability at most $O(\delta + \delta')$, where $\delta'$ is the variational distance between $\mathcal{D}$ and $\mathcal{D}'$. Since $\delta'$ in our case is $\frac{1}{\text{poly}(n)}$, we can safely ignore this additional failure probability going forward. This is summarized in the following lemma, which follows directly from the definition of variation distance. Namely, that the probability of any event in one distribution can change by at most the variation distance in another distribution, in particular the event that an algorithm succeeds on that distribution.

**Lemma 45** *Suppose we have an algorithm* $\mathcal{A}$ *that solves a problem* $\mathcal{P}$ *taken from a distribution* $\mathcal{D}$ *over* $\mathbb{R}^n$ *with probability* $1-\delta$*. Let* $\mathcal{D}'$ *be a distribution over* $\mathbb{R}^n$ *with variation distance at most* $\delta' \geq 0$ *from* $\mathcal{D}$*. Then if* $\mathcal{P}'$ *is drawn from* $\mathcal{D}'$*, algorithm* $\mathcal{A}$ *will solve* $\mathcal{P}'$ *with probability* $1-O(\delta+\delta')$.

**Corollary 46** *(Approximate ReLU Recovery.) Let* $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{X} \in \mathbb{R}^{d\times n}$ *be inputs such that* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$*, where* $f$ *is the ReLU activation function,* $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ *has full column rank,* $\mathbf{V}^* \in \mathbb{R}^{k\times d}$ *has full row rank, for all* $i \in [n]$, $\mathbf{X}_{*,i} \sim \mathcal{N}(0, \mathbb{I})$*. Then, there exists an algorithm that recovers* $\widehat{\mathbf{V}}$ *such that* $\|\widehat{\mathbf{V}} - \mathbf{D}\mathbf{\Pi}\mathbf{V}^*\|_F \leq \frac{1}{poly(n,m,d,\kappa(\mathbf{U}^*))}$*, where* $\mathbf{D}$ *is a diagonal* $\pm 1$ *matrix and* $\mathbf{\Pi}$ *is a permutation matrix. Further, the running time of this algorithm is* $poly(n, m, d, \kappa(\mathbf{U}^*))$.

First observe that we can assume WLOG that $\Pi = \mathbb{I}$, in other words that we recover an approximate $\mathbf{V}^*$ only up to its signs and not a permutation. We do this by simply (implicitly) permuting the rows of $\mathbf{V}^*$ to agree with our permutation, and permuting the columns of $\mathbf{U}^*$ by the same permutation. The resulting $\mathbf{A}$ is identical, and so we can assume that we know the permutation already.

**Algorithm 5 : ExactNeuralNet**$(\mathbf{A}, \mathbf{X})$

`Input :` Matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{X} \in \mathbb{R}^{d\times n}$ such that each entry in $\mathbf{X} \sim \mathcal{N}(0, 1)$.

1. Let $\mathcal{S}_3(x) = H_3(x)$, where $H_3(x) = \frac{-\nabla_x^{(3)} p(x)}{p(x)}$ is the 3-rd order Hermite polynomial and and $p(x) = \frac{1}{(\sqrt{2\pi})^d} e^{-\frac{\|x\|_2^2}{2}}$.
2. Let $\mathbf{A}' = \mathbf{A} + \mathbf{G}$ where $\mathbf{G} \in \mathbb{R}^{m\times n}$ and $\mathbf{G}_{i,j} \sim \mathcal{N}(0, 1)$.
3. Compute the 4-th order tensor $\widetilde{\mathbf{T}} = \frac{1}{n}\sum_{i=1}^n \mathbf{A}'_{*,i} \otimes \mathcal{S}_3(\mathbf{X}_{*,i})$. Collapse the first mode using a random vector $\theta$. By Lemma 39, $\widetilde{T}(\theta, \mathbb{I}, \mathbb{I}, \mathbb{I}) = \sum_{j=1}^k \lambda_j \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*} \otimes \mathbf{V}^*_{j,*}$, where $\lambda_j = \mathbb{E}_x\left[f'''(\mathbf{V}^* x)\right] \langle \mathbf{U}^*_{*,j}, \theta\rangle$.
4. Compute a CP-decomposition of $\widetilde{T}(\theta, \mathbb{I}, \mathbb{I}, \mathbb{I})$ using Tensor Power Method corresponding to Theorem 40, Janzamin et al. (2015), with accuracy parameter $\epsilon = \frac{1}{\text{poly}(d,m,\kappa(\mathbf{V}),\kappa(\mathbf{U}))}$ to obtain $\widehat{\mathbf{V}}$ such that $\|\widehat{\mathbf{V}} - \mathbf{D}\mathbf{\Pi}\mathbf{V}^*\|_F \leq \frac{1}{\text{poly}(d,m,\kappa(\mathbf{V}),\kappa(\mathbf{U}))}$, where $\mathbf{D}$ is a diagonal $\pm 1$ matrix and $\mathbf{\Pi}$ is a permutation matrix.
5. Run the Recovering Signs Algorithm (6) on $\widehat{\mathbf{V}}$, $\mathbf{A}$ and $\mathbf{X}$ to obtain $\mathbf{V}^*$.
6. Using the matrix $\mathbf{V}^*$ obtained above, set up and solve the following linear system for the matrix $\mathbf{U}$:
$$\mathbf{A} = \mathbf{U} f(\mathbf{V}^*\mathbf{X}) \tag{1}$$
7. Let $\mathbf{U}^*$ be the solution to the above linear system.

`Output :` $\mathbf{U}^*, \mathbf{V}^*$.

Unfortunately, the ambiguity in signs resulting from the algorithm of Theorem 40 is a non-trivial difficulty, and must be resolved algorithmically. This is due to the fact that the ReLU is sensitive to negative scalings, as $f(\cdot)$ only commutes with positive scalings. Suppose the diagonal of $\mathbf{D}$ of Corollary 46 is given by the coefficents $\xi_i \in \{1, -1\}$. Then in order to recover the weights, we must recover the terms $\xi_i$, Naively trying each sign results in a running time of $2^k$, which is no longer

polynomial[3]. Thus, a considerably technical challenge will be to show how to determine the correct scaling for each row even in the presence of noise. We begin with the case where there is no noise.

**Algorithm 6: Exact Recovery of $\mathbf{V}^*$**

`Input`: Matrices $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$, and $v_i^T \in \mathbb{R}^d$ s.t. $\|v_i - \xi_i \mathbf{V}^*_{i,*}\|_2 \leq \epsilon$ for some $\epsilon = O\left(\frac{1}{\text{poly}(d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))}\right)$ for some unknown $\xi_i \in \{1,-1\}$ and each $i = 1, 2, \ldots, k$.

1. Let $\overline{\mathbf{X}} \in \mathbb{R}^{d \times \ell}$ be the first $\ell = \text{poly}\left(k, d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*)\right)$ columns of $\mathbf{X}$, and let $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^* \overline{\mathbf{X}})$.

2. Let $\tau = \Theta(1/\text{poly}(\ell))$ be a thresholding value. Define the row vectors $v_i^+, v_i^- \in \mathbb{R}^\ell$ via

$$(v_i^+)_j = \begin{cases} f(v_i \overline{\mathbf{X}})_j & \text{if } f(v_i \overline{\mathbf{X}})_j > \tau \\ 0 & \text{otherwise} \end{cases} \qquad v_i^- = \begin{cases} f(-v_i \overline{\mathbf{X}})_j & \text{if } f(-v_i \overline{\mathbf{X}})_j > \tau \\ 0 & \text{otherwise} \end{cases}$$

   for $j = 1, 2, \ldots, \ell$.

3. Let $S_i^+$ be the sign pattern of $v_i^+$, and $S_i^-$ be the sign pattern of $v_i^-$. For $q \in \{+,-\}$, solve define the $r$ linear systems of equations in the variable $w_i^q \in \mathbb{R}^k$, where the $r$-th system is given by

$$(w_i^q \overline{\mathbf{A}})_j = 0 \quad \text{for } j \notin S_i^q$$
$$(w_i^q)_r = 1$$

   Where $(w_i^q)_r$ is the $r$-th coordinate of $(w_i^q)$. Then let $(w_i^q)$ be the vector returned from the first linear system which had a solution.

4. Let $q'$ be such that the above linear system returns a solution $w_i^{q'}$ with the constraints given by $S_i^{q'}$ (and at least one of the constraints of the form $(w_i^{q'})_r = 1$). We output **FAIL** if this occurs for both $q \in \{+,-\}$.

5. Output $\mathbf{V}_{i,*} = z_i / \|z_i\|_2$ where $z_i$ is the solution to the following linear system.

$$\text{for all } j \in S_i^{q'} \quad (z_i \overline{\mathbf{X}})_j = (w_i^q \overline{\mathbf{A}})_j$$

`Output`: $\mathbf{V}$ such that $\mathbf{V} = \mathbf{V}^*$.

**Recovering $\mathbf{V}^*$ from the Tensor Decomposition in the Noiseless Case.** Recall that the tensor power method provides us with row vectors $v_i$ such that $\|v_i - \xi_i \mathbf{V}^*_{i,*}\|_2 \leq \epsilon$ where $\epsilon =$

3. We remark that some prior results Janzamin et al. (2015) were able to handle this ambiguity by considering only a restricted class of smooth activation functions $f(\cdot)$ with the property that $f(x) = 1 - f(-x)$ for all $x \in \mathbb{R}$. Using affine transformations after application of the ReLU, this sign ambiguity for such activation functions can be accounted for. Since firstly the ReLU does not satisfy this condition and is non-trivially sensitive to the signs of its input,and secondly we are restricting to optimization over networks without affine terms, a more involved approach to dealing with sign ambiguity is required (especially for the noisy case).

$O\left(\frac{1}{\text{poly}(d,m,k,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))}\right)$ for $\xi_i \in \{\mathbf{V}^*_{i,*}, -\mathbf{V}^*_{i,*}\}$. Thus, the tensor power method gives us a *noisy* version of *either* $\mathbf{V}^*_{i,*}$ or $-\mathbf{V}^*_{i,*}$, however we do not know which. A priori, it would require $2^k$ time to guess the correct signs of the $k$ vectors $v_i$. In this section, we show that using the combinatorial sparsity patterns in the row span of $\mathbf{A}$, we can not only recover the signs, but recover the matrix $\mathbf{V}^*$ *exactly*. Our procedure is detailed in Algorithm 6 below, which takes the outputs $v_i$ from the tensor power method and returns the true matrix $\mathbf{V}^*$ up to a permutation of the rows.

Before we proceed, we recall a standard fact about the singular values of random Gaussian matrices.

**Lemma 47 (Corollary 5.35 Vershynin (2010))** *Let $\mathbf{S} \in \mathbb{R}^{k\times n}$ be a matrix of i.i.d. normal $\mathcal{N}(0,1)$ random variables, with $k < 10n$. Then with probability $1 - 2e^{-n/8}$, for all row vectors $w \in \mathbb{R}^\ell$ we have*

$$\sqrt{n}/3\|w\|_2 \leq \|w\mathbf{S}\|_2 \leq 2\sqrt{n}\|w\|_2$$

*In other words, we have $\sqrt{n}/3 \leq \sigma_{\min}(\mathbf{S}) \leq \sigma_{\max}(\mathbf{S}) \leq 2\sqrt{n}$.*

**Theorem 48** *With high probability in $d, m$, Algorithm 6 does not fail, and finds $\mathbf{V}$ such that $\mathbf{V} = \mathbf{V}^*$.*

**Proof** Fix a $i \in [k]$, and WLOG suppose the input row $v_i$ is such that $\|v_i - \mathbf{V}^*_{i,*}\|_2 \leq \epsilon$ (i.e. WLOG suppose $\xi_i = 1$). Then $\|v_i\overline{\mathbf{X}} - \mathbf{V}^*_{i,*}\overline{\mathbf{X}}\|_2 \leq O(1)\sqrt{\ell}\epsilon$ by the operator norm bound of Lemma 47, and since $f$ can only decrease the distance between matrices, it follows that $\|f(v_i\overline{\mathbf{X}}) - f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 \leq O(1)\sqrt{\ell}\epsilon$ . Similarly, we have $\|f(-v_i\overline{\mathbf{X}}) - f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 \leq O(1)\sqrt{\ell}\epsilon$ .

We now condition on the event that none of the non-zero entries of $f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$, and $f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$ are less than $\tau = \Theta(1/\text{poly}(\ell))$ (where $\tau$ is as in Algorithm 6), which holds by a union bound with probability $1 - 1/\text{poly}(\ell)$ (high prob in $d, m$) and the fact that the non-zero entries of these matrices have folded Gaussian marginals (distributed as the absolute value of a Gaussian). Given this, it follows that the sign patterns $S_i^+$ and $S_i^-$ of $v_i^+$ and $v_i^-$ are precisely the sign patterns of $f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$ and $f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$ respectively. Since $f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$ is in the row space of $\mathbf{A}$, at least one of the the linear systems run on $S_i^+$ will have a unique solution given by taking $w_i^+ = ce_i^T\mathbf{U}^{-1}$ for an appropriate constant $c \neq 0$ such that one of the constraints of the form $(w_i^+)_r = 1$ is satisfied, and where $\mathbf{U}^{-1}$ is the left inverse of $\mathbf{U}$.

Now consider the matrix $\mathbf{W}$ such that $\mathbf{W}$ is $\mathbf{V}^*$ with the row $-\mathbf{V}^*_{i,*}$ appended at the end. Then Applying the same argument as in Lemma 31, we see that the sign patterns of every pair of rows of $f(\mathbf{W}\overline{\mathbf{X}})$ disagrees on at least $\text{poly}(k)$ signs w.h.p.. This is easily seen for all pairs which contain one of $\{\mathbf{V}^*_{i,*}, -\mathbf{V}^*_{i,*}\}$ by applying the exact argument of the lemma and noting that the condition number of the matrix $\mathbf{V}^*$ does not change after negating the $i$-th row. The pair $\{f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\}$ itself disagrees on all sign patterns, which completes the proof of the claim. Note here that disagree means that, for any two rows $y^i, y^j$ in question there are at least $\text{poly}(k)$ coordinates such that *both* $y^i_p > 0$ and $y^j_p < 0$ *and* vice-versa. Thus no sparsity pattern is contained within any other. Then by Lemma 33, it follows that with high probability the only vector in the row span of $f(\mathbf{W}\overline{\mathbf{X}})$ which has a sparsity pattern contained within $S_i^-$ is a scalar multiple of $f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})$. Since no vector with such a sparsity pattern exists in the row span of $f(\mathbf{V}^*\overline{\mathbf{X}})$, the linear system with constraints given by $S_i^-$ will be infeasible with high probability.

We conclude from the above $q' = +$ in the fourth step of Algorithm 6, and that $w_i^{q'} = w_i^+$ is such that the sign pattern of $w_i^+ \overline{\mathbf{A}}$ is $S_i^+$, which is also the sign pattern of $f(\mathbf{V}_{i,*}^* \overline{\mathbf{X}})$. Since $w_i^+ \overline{\mathbf{A}}$ is in the row span of $f(\mathbf{V}^* \overline{\mathbf{X}})$, again by Lemma 33, we conclude that $w_i^+ \overline{\mathbf{A}} = cf(\mathbf{V}_{i,*}^* \overline{\mathbf{X}})$ for some constant $c > 0$ (we can enfoce $c > 0$ by flipping the sign of $w_i^+$ so that $w_i^+ \overline{\mathbf{A}}$ has no strictly negative entries). The linear system in step 5 solves the equation $z_i \overline{\mathbf{X}}_{S_i^+} = w_i^+ \overline{\mathbf{A}}_{S_i^+}$, where $\overline{\mathbf{X}}_{S_i^+}$ is $\overline{\mathbf{X}}$ restricted to the columns corresponding to indices in $S_i^+$, and similarly with $\overline{\mathbf{A}}_{S_i^+}$. This will have a unique solution if $\overline{\mathbf{X}}_{S_i^+}$ has full row rank. Since an index is included in $S_i^+$ with probability $1/2$ independently, it follows that $|S_i^+| > \ell/3 > \text{poly}(d)$ with probability $1 - 2^{-\Omega(\ell)}$. A column of $\overline{\mathbf{X}}_{S_i^+}$ is just an i.i.d. Gaussian vector conditioned on being in a fixed half-space. Then if the first $i < d$ columns of $\overline{\mathbf{X}}_{S_i^+}$ are independent, they span a $i-1$ dimensional subspace. The Lebesgue measure of this subspace intersected with the halfspace has has measure 0, since the half-space is $d$-dimensional and the subspace is $i$-dimensional. It follows that the probability that the $i+1$ column of $\overline{\mathbf{X}}_{S_i^+}$ is in this subspace is 0, from which we conclude by induction that $\overline{\mathbf{X}}_{S_i^+}$ has rank $d$. Thus the solution $z_i$ is unique, and must therefore be equal to $\frac{1}{c}\mathbf{V}_{i,*}^*$ as we also have $\mathbf{V}_{i,*}^* \mathbf{X}_{S_i^+} = cw_i^+ \overline{\mathbf{A}}$ for $c > 0$. After normalizing $z_i$ to have unit norm, we conclude $\mathbf{V}_{i,*} = z_i / \|z\|_2 = \mathbf{V}_{i,*}^*$ as needed. ■

Given the results developed thus far, the correctness of our algorithm for the exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in the realizable (noiseless) case follows immediately. Recall we can always assume WLOG that $\|\mathbf{V}_{i,*}^*\|_2 = 1$ for all rows $i \in [k]$.

**Theorem 49** *(Exact Recovery for Gaussian Input.) Suppose* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ *where* $\mathbf{U}^* \in \mathbb{R}^{m \times k}, \mathbf{V}^* \in \mathbb{R}^{k \times d}$ *are both rank-k, and such that* $\mathbf{X} \in \mathbb{R}^{d \times n}$ *is i.i.d. Gaussian. Assume WLOG that* $\|\mathbf{V}_{i,*}^*\|_2 = 1$ *for all rows* $i \in [k]$*. If* $n = \Omega(\text{poly}(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*)))$*, then Algorithm 5 runs in* $\text{poly}(n)$*-time and recovers* $(\mathbf{U}^*)^T, \mathbf{V}^*$ *exactly up to a permutation of the rows w.h.p. (in* $d, m$*).*

**Proof** By Theorem 40 and Corollary 46, we can recover $\mathbf{D}\mathbf{V}^*$ up to $\epsilon = \frac{1}{\text{poly}(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))}$ error in polynomial time, and then by Theorem 48 we can not only recover the signs $\xi_i$ that constitute the diagonal of $\mathbf{D}$, but also recover $\mathbf{V}^*$ *exactly* (all in a polynomial number of arithmetic operations). Given the fact that $f(\mathbf{V}^* \mathbf{X})$ is full rank by Lemma 16, the solution $\mathbf{U}$ to the linear system $\mathbf{U} f(\mathbf{V}^* \mathbf{X}) = \mathbf{A}$ is unique, and therefore equal to $\mathbf{U}^*$. This linear system can be solved in polynomial time by Gaussian elimination, and thus the runtime does not depend on the bit-complexity of $\mathbf{U}^*, \mathbf{V}^*$ in the real RAM model. So the entire procedure runs in time polynomial in the sample complexity $n$, which is polynomial in all relevant parameters as stated in the Theorem. ■

### C.3. Extension to Symmetric Input Distributions.

The independent and concurrent work of Ge et al. Ge et al. (2018) demonstrates the existence of an algorithm that approximately recovers $\mathbf{U}^*, \mathbf{V}^*$ in polynomial time, given that the input $\mathbf{X}$ is drawn from a mixture of a symmetric probability distribution and a Gaussian. In this section, we observe how our techniques can be combined with the those of Ge et al. (2018) to achieve exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ for this broader class of distributions. Namely, that we can replace running the tensor decomposition algorithm from Janzamin et al. (2015) with the algorithm of Ge et al. (2018) instead to obtain good approximations to $\mathbf{U}^*, \mathbf{V}^*$, and then use our results on the uniqueness of sparsity

patterns in the row-span of $f(\mathbf{V}^*\mathbf{X})$ to obtain exactly recovery. Only minor changes are needed in the proofs of our sparsity pattern uniqueness results (Lemmas 31 and 33) to extend them to mixtures of symmetric distributions and Gaussians.

.

**Definition 50** *(Symmetric Distribution.) Let $x \in \mathbb{R}^d$ be a vector random variable and $\mathcal{D}$ be a probability distribution function such that $x \sim \mathcal{D}$. Then, $\mathcal{D}$ is a symmetric distribution if for all $x$, the probability of $x$ and $-x$ is equal, i.e. $\mathcal{D}(x) = \mathcal{D}(-x)$.*

Ge et. al. Ge et al. (2018) define an object called the distinguishing matrix, denoted by $\mathbf{M}$, and require that the minimum singular value of $\mathbf{M}$ is bounded away from 0.

**Definition 51** *(Distinguishing Matrix Ge et al. (2018).) Given an input distribution $\mathcal{D}$ the distinguishing matrix is defined as $\mathbf{N}^{\mathcal{D}} \in \mathbb{R}^{d^2 \times \binom{k}{2}}$, whose columns are indexed by $i, j$ such that $1 \leq i < j \leq k$ and*

$$\mathbf{N}^{\mathcal{D}}_{i,j} = \frac{1}{n} \sum_{k \in [n]} (\mathbf{V}^*_{i,*}\mathbf{X}_{*,k})(\mathbf{V}^*_{j,*}\mathbf{X}_{*,k})(\mathbf{X}_{*,k} \otimes \mathbf{X}_{*,k}) \mathbf{1}\left\{(\mathbf{V}^*_{i,*}\mathbf{X}_{*,k})(\mathbf{V}^*_{j,*}\mathbf{X}_{*,k}) \leq 0\right\}$$

*Similarly an augmented distinguishing matrix $\mathbf{M}^{\mathcal{D}} \in \mathbb{R}^{d^2 \times \left(\binom{k}{2}+1\right)}$ has all the same columns as $\mathbf{N}^{\mathcal{D}}$ with the last column being $\frac{1}{n}\sum_{k \in [n]} \mathbf{X}_{*,k} \otimes \mathbf{X}_{*,k}$.*

In order to bound the singular values of the distinguishing matrix, Ge et. al. consider input distributions that are perturbations of symmetric distributions. In essence, given a desired target distribution $\mathcal{D}$, the algorithm of Ge et. al. can handle a similar distribution $\mathcal{D}_\gamma$, which is obtained by mixing $\mathcal{D}$ with a Gaussian with random covariance.

More formally, the perturbation is paramaterized by $\gamma \in (0, 1)$, which will define the mixing rate. It is required that $\gamma > \frac{1}{\text{poly}(N)}$ in order to achieve polynomial running time (where poly$(N)$ is the desired running time of the algorithm). First, let $\mathbf{G}$ be an i.i.d. entry-wise $\mathcal{N}(0, 1)$ random Gaussian matrix, which will be used to give the random the covariance. To generate $\mathcal{D}_\gamma$, first define a new distribution $\mathcal{N}'_{\mathbf{G}}$ as follows. To sample a point from $\mathcal{N}'_{\mathbf{G}}$, first sample a Gaussian $g \sim \mathcal{N}(0, \mathbb{I}_d)$ and then output $\mathbf{G}g$. Then the perturbation $\mathcal{D}_\gamma$ of the input distribution $\mathcal{D}$ is a mixture between $\mathcal{D}$ and $\mathcal{N}'_{\mathbf{G}}$. To sample $\mathbf{X}_{*,i}$ from $\mathcal{D}_\gamma$, pick $z$ as a Bernoulli random variable where $\Pr[z = 1] = \gamma$ and $\Pr[z = 0] = 1 - \gamma$, then for $i \in [n]$

$$\mathbf{X}_{*,i} \sim \begin{cases} \mathcal{D} & \text{if } z = 0 \\ \mathbf{G}g & \text{otherwise} \end{cases}$$

If the input is drawn from a mixture distribution $\mathcal{D}_\gamma$, $\sigma_{\min}(\mathbf{M})$ is bounded away from 0. We refer the reader to Section 2.3 in Ge et al. (2018) for further details. We observe that we can extend the main algorithmic result therein with our results on exact recover to recover $\mathbf{U}^*$, $\mathbf{V}^*$ with zero-error in polynomial time.

**Theorem 52** *(Informal Theorem 7 in Ge et al. (2018).) Let $\mathbf{U}^* \in \mathbb{R}^{m \times k}$, $\mathbf{V}^* \in \mathbb{R}^{k \times d}$ be full rank $k$ such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, $f$ is ReLU, and for all $i \in [n]$ $\mathbf{X}_{*,i} \sim \mathcal{D}_\gamma$ as defined above. Let $\mathbf{M}$ be the distinguishing matrix as defined in Ge et al. (2018). For all $i \in [n]$, let $\Gamma$ be such that $\|\mathbf{X}_{*,i}\|_2 \leq$*

$\Gamma$. *Then, there exists an algorithm than runs in time poly* $\left(\Gamma, 1/\epsilon, 1/\delta, \|\mathbf{U}^*\|_2, \frac{1}{\sigma_{\min}(\mathbb{E}[\mathbf{X}_{*,i}\mathbf{X}_{*,i}^T])}, \frac{1}{\sigma_{\min}(\mathbf{U}^*)}, \frac{1}{\sigma_{\min}(\mathbf{M})}\right)$ *and with probability* $1-\delta$ *outputs a matrix* $\widehat{\mathbf{U}}$ *such that* $\|\widehat{\mathbf{U}} - \mathbf{U}^*\mathbf{\Pi}\mathbf{D}\|_F \leq \epsilon$.

We use the algorithm corresponding to the aforementioned theorem to obtain an approximation $\widehat{\mathbf{U}}$ to $\mathbf{U}^*$, and then obtain an approximation to $f(\mathbf{V}^*\mathbf{X})$ by multiplying $\mathbf{A}$ on the left by $\widehat{\mathbf{U}}^{-1}$. The error in our approximation of $\mathbf{V}^*$ obtained via $\widehat{\mathbf{U}}^{-1}\mathbf{A}$ is analyzed in Section C.1. Given this approximation of $\mathbf{V}^*$, we observe that running steps 4-8 of our Algorithm 4 recovers $\mathbf{V}^*, \mathbf{U}^*$ exactly (see Remark 54 below). Note that the only part of Algorithm 4 that required $\mathbf{V}^*$ to be orthonormal is step 3 which runs ICA, which we are replacing here with the algorithm of Theorem 52.

Here we remove the random matrix $\mathbf{T}$ from Algorithm 4, as it is not needed if we are already given an approximation $\widehat{\mathbf{U}}$ of $\mathbf{U}^*$. Thus we proceed exactly as in Algorithm 4 by restricting $\mathbf{X}, \mathbf{A}$ to $\ell = \text{poly}(d, m, k, \frac{1}{\gamma}, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*))$ columns $\overline{\mathbf{X}}, \overline{\mathbf{A}}$, and then rounding the entries of $\widehat{f(\mathbf{V}\overline{\mathbf{X}})} = \widehat{\mathbf{U}}^{-1}\overline{\mathbf{A}}$ below $\tau$ to 0. Finally, we solve the same linear system as in Algorithm 4 to recover the rows of $f(\mathbf{V}^*\mathbf{X})$ exactly, from which $\mathbf{V}^*$ and then $\mathbf{U}^*$ can be exactly recovered via solving the final two linear systems in Algorithm 4. We summarized this formally as follows.

**Corollary 53** *(Exact Recovery for Symmetric Input.) Suppose* $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ *where* $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ *are both rank-k, for all* $i \in [n]$, $\mathbf{X}_{*,i} \sim \mathcal{D}_\gamma$, *and* $\|\mathbf{X}_{*,i}\|_2 \leq \Gamma$. *Assume WLOG that* $\|\mathbf{V}^*_{i,*}\|_2 = 1$ *for all rows* $i \in [k]$. *If*

$$n \geq \text{poly}\left(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \Gamma, \frac{1}{\gamma}, \|\mathbf{U}^*\|_2, \frac{1}{\sigma_{\min}(\mathbb{E}[\mathbf{X}_{*,i}\mathbf{X}_{*,i}^T])}, \frac{1}{\sigma_{\min}(\mathbf{U}^*)}, \frac{1}{\sigma_{\min}(\mathbf{M})}\right)$$

*then there exists an algorithm that runs in poly*$(n)$*-time and recovers* $(\mathbf{U}^*)^T, \mathbf{V}^*$ *exactly up to a permutation of the rows w.h.p. (in* $d, m$*).*

**Remark 54** *To prove the correctness of Algorithm 6 on* $\mathcal{D}_\gamma$*, we need only to generalize Lemmas 31 and 33 which together give the uniqueness of sparsity patterns of the rows of* $f(\mathbf{V}^*\mathbf{X})$ *in the rowspan of* $\mathbf{A}$. *We note that Lemma 31 can be easily generalized by first conditioning on the input being Gaussian, which in* $\mathcal{D}_\gamma$ *occurs with* $\gamma$ *probability, and then going applying the same argument, replacing* $\frac{1}{\kappa}$ *with* $\frac{\gamma}{\kappa}$ *everywhere. The only change in the statement of Lemma 31 is that we now require* $\ell = tpoly(k, \kappa, \frac{1}{\gamma})$ *to handle* $\mathbf{X} \sim \mathcal{D}_\gamma$.

*Next, the proof of Lemma 33 immediately goes through as the argument in the proof which demonstrates the determinant in question is non-zero only requires that the distribution* $\mathcal{D}_\gamma$ *is non-zero everywhere in the domain. Namely, the proof requires that the support of* $\mathcal{D}_\gamma$ *is all of* $\mathbb{R}^d$. *Note, this condition is always the case for the mixture* $\mathcal{D}_\gamma$ *since Gaussians are non-zero everywhere in the domain.*

### C.4. Necessity of poly$(\kappa(\mathbf{V}^*))$ Sample Complexity

So far, our algorithms for the exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ have had polynomial dependency on the condition numbers of $\mathbf{U}^*$ and $\mathbf{V}^*$. In this section, we make a step towards justifying the necessity of these dependencies. As always, we work without loss of generality under the assumption that $\|\mathbf{V}_{i,*}\|_2 = 1$ for all rows $i \in [k]$. Specifically, demonstrate the following.

**Lemma 55** *Any algorithm which, when run on $\mathbf{A}, \mathbf{X}$, where that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$, and $\mathbf{X}$ has i.i.d. Gaussian $\mathcal{N}(0,1)$ entries, recovers $(\mathbf{U}^*)^T, \mathbf{V}^*$ exactly (up to a permutation of the rows) with probability at least $1 - c$ for some sufficiently small constant $c > 0$, requires $n = \Omega(\kappa(\mathbf{V}^*))$ samples.*

**Proof** We construct two instances of $\mathbf{A}^1 = \mathbf{U}^1 f(\mathbf{V}^1 \mathbf{X})$ and $\mathbf{A}^2 = \mathbf{U}^1 f(\mathbf{V}^2 \mathbf{X})$ . Let

$$\mathbf{U}^1 = \begin{bmatrix} \sqrt{1+a^2}/2 & \sqrt{1+a^2}/2 \end{bmatrix} \qquad \mathbf{V}^1 = \begin{bmatrix} \frac{1}{\sqrt{1+a^2}} & \frac{a}{\sqrt{1+a^2}} \\ \frac{1}{\sqrt{1+a^2}} & -\frac{a}{\sqrt{1+a^2}} \end{bmatrix}$$

$$\mathbf{U}^2 = \begin{bmatrix} \sqrt{1+(2a)^2}/2 & \sqrt{1+(2a)^2}/2 \end{bmatrix} \qquad \mathbf{V}^2 = \begin{bmatrix} \frac{1}{\sqrt{1+(2a)^2}} & \frac{a}{\sqrt{1+(2a)^2}} \\ \frac{1}{\sqrt{1+(2a)^2}} & -\frac{(2a)}{\sqrt{1+(2a)^2}} \end{bmatrix}$$

Now note that for $a \in [0,1]$, the rows of $\mathbf{V}^1$ have unit norm, and $\kappa(\mathbf{V}^1) = \frac{1}{a}$. Now let $a^i = ia$, and note, however, that for the $j$-th sample $\mathbf{X}_{*,j} = [x_1^j, x_2^j]$ i.i.d. Gaussian, we have for $i \in \{1,2\}$

$$\mathbf{U}^i f(\mathbf{V}^i \mathbf{X}_{*,j}) = \frac{f(x_1^j + a^i x_2^j) + f(x_1^j - a^i x_2^j)}{2}$$

Now note that when $|x_1^j| > (2a)|x_2^j|$, for *both* $i \in \{1,2\}$ we have either

$$\mathbf{A}^i_{*,j} = \mathbf{U}^i f(\mathbf{V}^i \mathbf{X}_{*,j}) = 0$$

or

$$\mathbf{A}^i_{*,j} = \mathbf{U}^i f(\mathbf{V}^i \mathbf{X}_{*,j}) = x_1^j$$

And in either case we do not get any information about $a$. In such a case, the $j$-th column of $\mathbf{A}^1$ and $\mathbf{A}^2$ are the same. In particular, conditioned on a given $\mathbf{X}$ such that $|x_1^j| > (2a)|x_2^j|$ for all columns $j$, we have $\mathbf{A}^1 = \mathbf{A}^2$. Now note that the probability that one Gaussian is $\frac{1}{2a}$ times larger than another is $\Theta(\frac{1}{a})$, thus any algorithm that takes less than $c\frac{1}{a}$ samples, for some absolute constant $c > 0$, cannot distinguish between $\mathbf{A}^1$ and $\mathbf{A}^2$, since we will have $\mathbf{A}^1 = \mathbf{A}^2$ with $\Omega(1)$ probability in this case, which completes the proof. ■

## Appendix D. A Polynomial Time Algorithm for Gaussian input and Sub-Gaussian Noise

In the last section, we gave two algorithms for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in the noiseless (exact) case. Namely, where the algorithm is given as input $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X})$ and $\mathbf{X}$. Our general algorithm for this problem first utilized a tensor decomposition algorithm which allowed for approximate recovery of $\mathbf{V}^*$, up to the signs of its rows. Observe that this procedure, given by Theorem 40, can handle mean zero subgaussian noise $\mathbf{E}$, such that $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^* \mathbf{X}) + \mathbf{E}$. In this section, we will show how to utilize this fact as a sub-procedure to recover approximately recover $\mathbf{U}^*, \mathbf{V}^*$ in this noisy case.

We begin with using the algorithm corresponding to Theorem 40 to get an approximate solution to $\mathbf{V}^*$, up to permutations and $\pm 1$ scaling. We note that the guarantees of Theorem 40 still hold when the noise $\mathbf{E}$ is sub-Gaussian. Therefore, we obtain a matrix $\widetilde{\mathbf{V}}$ such that $\|\widehat{\mathbf{V}} - \mathbf{D}\mathbf{\Pi}\mathbf{V}^*\|_F \leq \epsilon$, where $\mathbf{D}$ is a diagonal $\pm 1$ matrix and $\mathbf{\Pi}$ is a permutation matrix.

**Algorithm 7: Recovering Signs**$(v_i, \mathbf{A}, \mathbf{X})$

`Input:` Matrices $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$, and $v_i^T \in \mathbb{R}^d$ s.t. $\|v_i - \xi_i \mathbf{V}^*_{i,*}\|_2 \leq \epsilon$ for some unknown $\xi_i \in \{1, -1\}$ and $i = 1, 2, \ldots, k$, where $\epsilon = O(\frac{1}{\text{poly}(d,m,k,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*)}})$.

1. Let $\overline{\mathbf{X}} \in \mathbb{R}^{d \times \ell}$ be the first $\ell = \text{poly}(k, d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$ columns of $\mathbf{X}$, and similarly define $\overline{\mathbf{E}}$, and let $\overline{A} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}}) + \overline{\mathbf{E}}$.

2. For $i \in [k]$, let
$$S_{i,+} = \{f(v_j\overline{\mathbf{X}}), f(-v_j\overline{\mathbf{X}})\}_{j \neq i} \cup \{f(v_i\overline{\mathbf{X}})\}$$
and
$$S_{i,-} = \{f(v_j\overline{\mathbf{X}}, f(-v_j\overline{\mathbf{X}})\}_{j \neq i} \cup \{f(-v_i\overline{\mathbf{X}})\}$$

3. Let $\mathbf{P}_{S_{i,+}}$ be the orthogonal projection matrix onto the row span of vectors in $S_{i,+}$. Compute
$$a^+_{i,j} = \|\overline{\mathbf{A}}_{j,*}(\mathbb{I} - \mathbf{P}_{S_{i,+}})\|_2^2$$
$$a^-_{i,j} = \|\overline{\mathbf{A}}_{j,*}(\mathbb{I} - \mathbf{P}_{S_{i,-}})\|_2^2$$
For each $j \in [m]$.

4. Let $a^+_i = \sum_j a^+_{i,j}$, and $a^-_i = \sum_j a^-_{i,j}$. If $a^+_i < a^-_i$, set $\mathbf{V}_{i,*} = v_i$, otherwise set $\mathbf{V}_{i,*} = -v_i$.

`Output:` $\mathbf{V}$ such that $\|\mathbf{V} - \mathbf{V}^*\|_2 \leq \epsilon$, thus recovering $\xi_i$ for $i \in [k]$.

Recall that in the noiseless case, we needed to show that given a approximate version of $\mathbf{V}^*$ up to the signs of the rows, we can recover both the signs and $\mathbf{V}^*$ exactly in polynomial time. Formally, we were given rows $v_i$ such that $\|v_i - \xi_i \mathbf{V}^*_{i,*}\|_2$ was small for some $\xi_i \in \{1, -1\}$, however we did not know $\xi_i$. This issue is a non-trivial one, as we cannot simply guess the $\xi_i$'s (there are $2^k$ possibilities), and moreover we cannot assume WLOG that the $\xi_i$'s are 1 by pulling the scaling through the ReLU, which is only commutes with *positive* scalings. Our algorithm for recovery of the true signs $\xi_i$ in the exact case relied on combinatorial results about the sparsity patterns of $f(\mathbf{V}^*\mathbf{X})$. Unfortunately, these combinatorial results can no longer be used as a black-box in the noisy case, as the sparsity patterns can be arbitrarily corrupted by the noise. Thus, we must develop a refined, more general algorithm for the recovery of the signs $\xi_i$ in the noise case. Thus we begin by doing precisely this.

### D.1. Recovering the Signs $\xi_i$ with Subgaussian Noise

**Lemma 56** *Let $g \in \mathbb{R}^n$ be a row vector of i.i.d. mean zero variables with variance $\sigma$, and let $\mathcal{S}$ be any fixed $k$ dimensional subspace of $\mathbb{R}^n$. Let $\mathbf{P}_{\mathcal{S}} \in \mathbb{R}^{n \times n}$ be the projection matrix onto $\mathcal{S}$. Then for any $\delta > 0$ with probability $1 - \delta$, we have*

$$\|g\mathbf{P}_{\mathcal{S}}\|_2 = \sigma\sqrt{k/\delta}$$

**Proof** We can write $P_{\mathcal{S}} = \mathbf{W}^T\mathbf{W}$ for matrices $\mathbf{W} \in \mathbb{R}^{k\times n}$ with orthonormal rows. Then $\mathbb{E}[\|g\mathbf{W}^T\|_2^2] = \sigma^2 k$, and by Markov bounds with probability $1-\delta$ we have $\|g\mathbf{W}^T\|_2^2 = \|g\mathbf{W}^T\mathbf{W}\|_2^2 < \sigma^2 k/\delta$ as needed.

■

**Lemma 57** *Let $\mathbf{Q} \in \mathbb{R}^{k\times \ell}$ be a matrix of row vectors for $\ell > poly(k)$ (for some sufficiently large polynomial) with $1 \le \|\mathbf{Q}\|_2$ and let $\mathbf{P_Q} = \mathbf{Q}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}$ be the projection onto them. Let $\mathbf{E}$ be such that $\|\mathbf{E}\|_F \le \frac{\epsilon}{\left(\kappa(\mathbf{Q})\|\mathbf{Q}\|_2\right)^4}$, and let $\mathbf{P_{Q+E}}$ be the projection onto the rows of $\mathbf{Q}+\mathbf{E}$. Then for any vector $x^T \in \mathbb{R}^\ell$, we have*

$$\|x\mathbf{P_{Q+E}}\|_2 = \|x\mathbf{P_Q}\|_2 \pm O(\epsilon\|x\|_2)$$

**Proof** We have $\mathbf{P_{Q+E}} = (\mathbf{Q}+\mathbf{E})^T((\mathbf{Q}+\mathbf{E})^T(\mathbf{Q}+\mathbf{E}))^{-1}(\mathbf{Q}+\mathbf{E})$. Now

$$(\mathbf{Q}+\mathbf{E})^T(\mathbf{Q}+\mathbf{E}) = \mathbf{Q}^T\mathbf{Q} + \mathbf{E}^T\mathbf{Q} + \mathbf{Q}^T\mathbf{E} + \mathbf{E}^T\mathbf{E}$$

Further, $\|\mathbf{E}^T\mathbf{Q} + \mathbf{Q}^T\mathbf{E} + \mathbf{E}^T\mathbf{E}\|_F \le \|\mathbf{E}\|_F\|\mathbf{Q}\|_2 + \|\mathbf{E}\|_F^2 \le 2\frac{\epsilon}{\kappa^4(\mathbf{Q})\|\mathbf{Q}\|_2^2}$. Thus we can write $(\mathbf{Q}+\mathbf{E})^T(\mathbf{Q}+\mathbf{E}) = \mathbf{Q}^T\mathbf{Q} + \mathbf{Z}$ where $\|\mathbf{Z}\|_F \le 2\frac{\epsilon}{\kappa^4(\mathbf{Q})\|\mathbf{Q}\|_2^2}$. Applying Corollary 35 with $\mathbf{B} = \mathbf{Q}^T\mathbf{Q}$, and $\mathbf{E} = \mathbf{Z}$, we can write $(\mathbf{Q}+\mathbf{E})^T(\mathbf{Q}+\mathbf{E}))^{-1} = (\mathbf{Q}^T\mathbf{Q})^{-1} + \mathbf{Z}'$, where $\|\mathbf{Z}'\|_F \le O(\frac{\epsilon}{\kappa^2(\mathbf{Q})\|\mathbf{Q}\|_2^2})$. Thus

$$\begin{aligned}\mathbf{P_{Q+E}} &= (\mathbf{Q}+\mathbf{E})^T((\mathbf{Q}^T\mathbf{Q})^{-1} + \mathbf{Z}')(\mathbf{Q}+\mathbf{E}) \\ &= \mathbf{P_Q} + \mathbf{Q}^T\mathbf{Z}'(\mathbf{Q}+\mathbf{E}) + \mathbf{Q}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{E} + \mathbf{E}^T((\mathbf{Q}^T\mathbf{Q})^{-1} + \mathbf{Z}')(\mathbf{Q}+\mathbf{E})\end{aligned}$$

Therefore,

$$\begin{aligned}\|\mathbf{Q}^T\mathbf{Z}'(\mathbf{Q}+\mathbf{E})\|_F &\le \|\mathbf{Q}^T\|_2\|\mathbf{Z}'(\mathbf{Q}+\mathbf{E})\|_F \\ &\le \|\mathbf{Q}^T\|_2\|\mathbf{Z}'\|_F\|(\mathbf{Q}+\mathbf{E})\|_2 \\ &\le \|\mathbf{Q}^T\|_2\|\mathbf{Z}'\|_F(\|\mathbf{Q}\|_2 + \|\mathbf{E}\|_F) \\ &= O\left(\frac{\epsilon}{\kappa^2(\mathbf{Q})}\right) = O(\epsilon)\end{aligned}$$

Next, we have

$$\begin{aligned}\|\mathbf{Q}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{E}\|_F &\le \|\mathbf{Q}^T\|_2\|(\mathbf{Q}^T\mathbf{Q})^{-1}\|_2\|\mathbf{E}\|_F \\ &\le \frac{\epsilon}{\|\mathbf{Q}\|_2^3} \\ &< \epsilon\end{aligned}$$

where in the second to last inequality we used the fact that $\|\mathbf{Q}\|_2 > 1$ so $\sigma_{\min}^{-2}(\mathbf{Q}) = \|(\mathbf{Q}^T\mathbf{Q})^{-1}\|_2 < 1/\kappa^2(\mathbf{Q})$. Applying the above bounds similarly, we have $\|\mathbf{E}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}\|_F \le O(\epsilon)$, $\|\mathbf{E}^T(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{E}\|_F \le O(\epsilon)$, and $\|\mathbf{E}^T\mathbf{Z}'(\mathbf{Q}+\mathbf{E})\|_F \le O(\epsilon)$. We conclude $\mathbf{P_{Q+E}} = \mathbf{P_Q} + \mathbf{Z}''$, where $\|\mathbf{Z}''\|_F \le O(\epsilon)$. It follows that for any $x \in \mathbb{R}^\ell$, we have

$$\begin{aligned}\|x\mathbf{P_{Q+E}}\|_2 &= \|x\mathbf{P_Q}\|_2 \pm \|x\mathbf{Z}''\|_2 \\ &= \|x\mathbf{P_Q}\|_2 \pm O(\epsilon\|x\|_2)\end{aligned}$$

■

**Lemma 58** *Let $\mathbf{Q} \in \mathbb{R}^{r \times \ell}$ for $1 \le r \le 2k$ be any matrix whose rows are formed by taking $r$ distinct rows from the set $\{f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), f(-\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\}_{i\in[k]}$, where $\overline{\mathbf{X}}$ is $\mathbf{X}$ restricted to the first $\ell = poly(k, d, m, \kappa(\mathbf{V}^*))$ columns. Then w.h.p. (in $\ell$), both $\|\mathbf{Q}\|_F^2 \le 10r\ell$ and $\sigma_{\min}(\mathbf{Q}) = \Omega(\frac{\sqrt{\ell}}{(\kappa(\mathbf{V}^*))^2}))$.*

**Proof** The first bound $\|\mathbf{Q}\|_F^2 \le 10r\ell$ follows from the fact that the $\|\cdot\|_2^2$ norm of each row is distributed as a $\chi^2$ random variable, so the claim follows from standard tail bounds for such variables Laurent and Massart (2000). For the second claim, write $\mathbf{Q} = f(\mathbf{W}\overline{\mathbf{X}})$, where the rows of $\mathbf{W}$ are the $r$ distinct rows from the set $\{\mathbf{V}^*_{i,*}, -\mathbf{V}^*_{i,*}\}_{i\in[k]}$ corresponding to the rows of $\mathbf{Q}$. Let $\mathbf{W}^+$ be the subset of rows of the form $\mathbf{V}^*_{i,*}$, and $\mathbf{W}^-$ its complement. There now there is a rotation matrix $\mathbf{R}$ that rotates $\mathbf{W}^+$ to be lower triangular, so that the $j$-th row of $\mathbf{W}^+\mathbb{R}$ is supported on the first $j$ columns. Let $\mathbf{W}$ be such that the pairs of rows $\{\mathbf{V}^*_{i,*}, -\mathbf{V}^*_{i,*}\}$ with the same index $i$ are placed together. Then $\mathbf{W}\mathbb{R}$ is block-upper triangular, where the $j$-th pair of rows of the form $\{\mathbf{V}^*_{i,*}, -\mathbf{V}^*_{i,*}\}$ are supported on the first $j$ columns. Since Gaussians are rotationally invariant, $\mathbf{W}\mathbf{R}\overline{\mathbf{X}}$ has the same distribution as $\mathbf{W}\overline{\mathbf{X}}$, thus we can assume that $\mathbf{W}$ is in this block lower triangular form, and $\mathbf{V}^*$ is in lower triangular form.

WLOG assume the rank of $\mathbf{W}$ is $k$ (the following arguments will hold when the rank is $k' < k$). We now claim that we can write $\mathbf{W}_{r,*} = \alpha + \varphi e_k$, where $\alpha$ is in the span of $e_1, \ldots, e_{k-1}$ and $\varphi = \Omega(\frac{1}{\kappa})$ where $\kappa = \kappa(\mathbf{V}^*)$. To see this, note that if this were not the case, the projection of $\mathbf{W}_{r,*}$ onto the all prior rows with the same sign (i.e. all either of the form $\mathbf{V}^*_{i,*}$ or $-\mathbf{V}^*_{i,*}$) would be less than $\frac{1}{\kappa}$, since the prior span all of $\mathbb{R}^{k-1}$ on the first $k-1$ columns, and the only part of $\mathbf{W}_{r,*}$ outside of this span has weight $\varphi$. Let $w$ be such that $w\mathbf{W}'$ is this projection, where $\mathbf{W}'$ excludes the last row of $\mathbf{W}$ WLOG this row is of the form $\mathbf{V}^*_{i,*}$, and WLOG $i = k$. Then can write $w\mathbf{W}' = v(\mathbf{V}^*)'$ where $(\mathbf{V}^*)'$ excludes the last row of $\mathbf{V}^*$. Then $\|[v, -1]\mathbf{V}^*\|_2 < \frac{1}{\kappa}$, and since $\|\mathbf{V}^*\|_2 \ge 1$ it follows that $\kappa(\mathbf{V}^*) > \kappa$, a contradiction since $\kappa$ is defined as the condition number of $\mathbf{V}^*$.

Now let $x^i$ be the $i$-th column of $\overline{\mathbf{X}}$, and let $\mathcal{E}_i$ be the event that $\varphi x^i_k > \lambda|\langle\alpha, (x^i_1, \ldots, x^i_{k-1})\rangle|$, where $\lambda$ will later be set to be a sufficiently large constant. Since $\varphi x^i_k$ and $\langle\alpha, (x^i_1, \ldots, x^i_{k-1})\rangle$ are each distributed as a Gaussian with variance at most 1, by anti-concentration of Gaussians $\Pr[\mathcal{E}_i] = \Omega(\frac{1}{\lambda\kappa})$. So let $\mathcal{S} \subset [\ell]$ be the subset of $i$ such that $\mathcal{E}_i$ holds, and let $\mathbf{X}_\mathcal{S}$ be $\overline{\mathbf{X}}$ restricted to this subset. Let $\mathbf{V}^*_{i,*}$ be the last column of $\mathbf{W}$ (WLOG we assume it is $\mathbf{V}^*_{i,*}$ and not $-\mathbf{V}^*_{i,*}$). We now upper bound the norm of the projection of $f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S})$ onto the row span of $f(\mathbf{W}\mathbf{X}_\mathcal{S})$. Now $f(-\mathbf{V}^*_{i,*}\mathbf{X})$, if it exists as a row of $f(\mathbf{W}\mathbf{X})$, will be identically 0 on the coordinates in $\mathcal{S}$ (because $\mathbf{V}^*_{i,*}\mathbf{X}$ is positive on these coordinates by construction). So we can disregard it in the following analysis. By construction of $\mathcal{S}$ we can write

$$f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S}) = f(\varphi(\mathbf{X}_\mathcal{S})_{k,*}) + b$$

where $\|b\|_2 \le \|\frac{1}{\lambda}f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S})\|_2$, where $(\mathbf{X}_\mathcal{S})_{k,*}$ is the $k$-th row of $\mathbf{X}_\mathcal{S}$. By the triangle inequality, the projection of $f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S})$ onto the rowspan of $f(\mathbf{W}'\mathbf{X}_\mathcal{S})$ (where $\mathbf{W}'$ is $\mathbf{W}$ excluding $\mathbf{V}^*_{i,*}$), is

$$\|f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S})\mathbf{P}_{\mathbf{W}'}\|_2 \le \frac{1}{\lambda}\|f(\mathbf{V}^*_{i,*}\mathbf{X}_\mathcal{S})\|_2 + \|f(\varphi(\mathbf{X}_\mathcal{S})_{k,*})\mathbf{P}_{\mathbf{W}'}\|_2$$

where $\mathbf{P}_{\mathbf{W}'}$ is the projection onto the rowspan of $f(\mathbf{W}'\mathbf{X}_\mathcal{S})$. Crucially, observe that $f(\mathbf{W}'\mathbf{X}_\mathcal{S})$, and thus $\mathbf{P}_{\mathbf{W}'}$, does not depend on the $k$-th row $(\mathbf{X}_\mathcal{S})_{k,*}$ of $\mathbf{X}_\mathcal{S}$. Now

$$\|f(\varphi(\mathbf{X}_\mathcal{S})_{k,*})\mathbf{P}_{\mathbf{W}'}\|_2 \le \|f(\varphi(\mathbf{X}_\mathcal{S})_{k,*})\mathbf{P}_{\mathbf{W}'+1}\|_2$$

where $\mathbf{P}_{\mathbf{W}'+1}$ is the projection onto the row span of $\{f(\mathbf{W}'\mathbf{X}_{\mathcal{S}})_{j,*}\}_{\text{rows } j \text{ of } \mathbf{W}'} \bigcup \{\mathbf{1}\}$ where $\mathbf{1}$ is the all $1's$ vector. This holds since adding a vector to the span of the subspace being projected onto can only increase the length of the projection. Let $\mathbf{P}_1$ be the projection just onto the row $\mathbf{1}$.

Now observe that $\varphi f((\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)$ is a mean 0 i.i.d. shifted rectified-Gaussian vector with variance strictly less than $\varphi^2$ (here rectified means 0 with prob $1/2$ and positive Gaussian otherwise). Moreover, the mean of the entries of $f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})$ is $\Theta(\varphi)$. The $L_2$ of these vectors are thus sums of sub-exponential random variables, so by standard sub-exponential concentration (see e.g. Wainwright (2019)) we have $\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\|_2 = \Theta(\varphi)\sqrt{|S|}$, and moreover

$$\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2 - \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\|_2 = \Omega(\varphi)\sqrt{|\mathcal{S}|} \tag{2}$$

w.h.p in $\log(|\mathcal{S}|)$ where $|S| \geq \Theta(1)\frac{\ell}{\kappa\lambda} = \text{poly}(d, m, k, \kappa)$. Now by Lemma 56, we have

$$\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\mathbf{P}_{\mathbf{W}'+1}\|_2 \leq \varphi\sqrt{(2k+1)/\delta}$$

with probability $1 - \delta$, for some $\delta = 1/\text{poly}(k, d, m)$. So

$$\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\mathbf{P}_{\mathbf{W}'+1}\|_2 \leq O(\frac{\text{poly}(k, d, m)}{\sqrt{|\mathcal{S}|}})\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\|_2$$

Write $f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*}) = f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\mathbf{P}_{+1} + f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)$. Then by triangle inequality, we can upper bound $\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\mathbf{P}_{\mathbf{W}'+1}\|_2$ by

$$\begin{aligned}
&\leq \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\mathbf{P}_1\mathbf{P}_{\mathbf{W}'+1}\|_2 + \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\mathbf{P}_{\mathbf{W}'+1}\|_2 \\
&\leq \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\mathbf{P}_1\|_2 + O(\frac{\text{poly}(k, d, m)}{\sqrt{|S|}})\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\|_2 \\
&= \left(\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2^2 - \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})(\mathbb{I} - \mathbf{P}_1)\|_2^2\right)^{1/2} + O(\frac{\text{poly}(k, d, m)}{\sqrt{|S|}})\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2
\end{aligned}$$

Using the bound from Equation 2, for some constants $c, c' < 1$ bounded away from 1, we have

$$\begin{aligned}
&= \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2(1 - c) + O(\frac{\text{poly}(k, d, m)}{\sqrt{|S|}})\|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2 \\
&\leq \|f(\varphi(\mathbf{X}_{\mathcal{S}})_{k,*})\|_2(1 - c')
\end{aligned}$$

Thus $\|f(\mathbf{V}_{i,*}^*\mathbf{X}_{\mathcal{S}})\mathbf{P}_{\mathbf{W}'}\|_2 \leq \|f(\mathbf{V}_{i,*}^*\mathbf{X}_{\mathcal{S}})\mathbf{P}_{\mathbf{W}'+1}\|_2 \leq (1 - \Theta(1))\|f(\mathbf{V}_{i,*}^*\mathbf{X}_{\mathcal{S}})\|_2$. Now by setting $\lambda > 2c'$ a sufficently large constant, the $b$ term becomes negligible, and the above bound holds replacing $c'$ with $c'/2$. Since we have $\|f(\mathbf{V}_{i,*}^*\mathbf{X}_{\mathcal{S}})\|_2 = \Theta(\varphi\sqrt{|\mathcal{S}|})$, and $\|f(\mathbf{V}_{i,*}^*\overline{\mathbf{X}})\|_2 = \Theta(\sqrt{\ell})$, if $\overline{\mathbf{P}_{\mathbf{W}'}}$ is projection of onto the rows of $f(\mathbf{W}'\overline{X})$, we have

$$\begin{aligned}
\|f(\mathbf{V}_{i,*}^*\overline{\mathbf{X}})\overline{\mathbf{P}_{\mathbf{W}'}}\|_2 &\leq \|f(\mathbf{V}_{i,*}^*\overline{\mathbf{X}})\|_2\left(1 - \Theta\left(\frac{\varphi}{\kappa\lambda}\right)\right) \\
&< \|f(\mathbf{V}_{i,*}^*\overline{\mathbf{X}})\|_2(1 - \Theta(\frac{1}{\kappa^2}))
\end{aligned}$$

Where here we recall $\lambda = \Theta(1)$. Since this argument used no facts about the row $i$ we were choosing, it follows that the norm of the projection of any row onto the subspace spanned by the

others others in $f(\mathbf{W}\overline{\mathbf{X}})$ is at most a $(1-\Theta(\frac{1}{\kappa^2}))$ factor less than the norm was before the projection. In particular, this implies that $f(\mathbf{W}\overline{\mathbf{X}})$ is full rank. Note by sub-exponential concentration, each row norm is $\Theta(\sqrt{\ell})$ w.h.p. We are now ready to complete the argument. Write $f(\mathbf{W}\overline{\mathbf{X}}) = \mathbf{B}\Sigma\mathbf{Q}^T$ in its singular value decomposition. Since the projection of one row onto another does not change by a row rotation,, we can rotate $\mathbf{Q}^T$ to be the identity, and consider $\mathbf{B}\Sigma$. Let $u_i$ be a unit vector in the direction of the $i$-th row projected onto the orthogonal space to the prior rows. Now for any unit vector $u$, write it as $u = \sum_i u_i a_i$ (which we can do because $f(\mathbf{W}\overline{\mathbf{X}})$ is full rank). Noting that $\frac{\|f(\mathbf{W}\overline{\mathbf{X}})_{i,*}\|_2}{\|f(\mathbf{W}\mathbf{X}_{\mathcal{S}})_{i,*}\|_2} = O(\text{poly}(k)\kappa^2)$ for any row $i$, we have

$$\begin{aligned}\|f(\mathbf{W}\overline{\mathbf{X}})u\|_2^2 &\geq \sum_i \langle u_i, u\rangle^2 \Omega(\frac{\ell}{\kappa^4}) \\ &\geq \sum_i a_i^2 \Omega(\frac{\ell}{\kappa^4}) \\ &= \Omega(\frac{\ell}{\kappa^4})\end{aligned}$$

Thus $\sigma_{\min}(\mathbf{Q}) = \sigma_{\min}(f(\mathbf{W}\overline{\mathbf{X}})) = \Omega(\frac{\sqrt{\ell}}{\kappa^2})$ as needed, where recall we have been writing $\kappa = \kappa(\mathbf{V}^*)$. ■

Using the bounds developed within the proof of the prior lemma gives the following corollary.

**Corollary 59** *Let $\mathbf{P}_{S_{i,+}}, \mathbf{P}_{S_{i,-}}$ be as in Algorithm 7. Then*

$$\|f(\mathbf{V}_{i,*}^*\mathbf{X})\mathbf{P}_{S_{i,-}}\|_2 = \|f(\mathbf{V}_{i,*}^*\mathbf{X})\|_2\Big(1 - \Omega(\frac{1}{\kappa(\mathbf{V}^*)^2 poly(k)})\Big)$$

*and*

$$\|f(-\mathbf{V}_{i,*}^*\mathbf{X})\mathbf{P}_{S_{i,+}}\|_2 = \|f(-\mathbf{V}_{i,*}^*\mathbf{X})\|_2\Big(1 - \Omega(\frac{1}{\kappa(\mathbf{V}^*)^2 poly(k)})\Big)$$

**Theorem 60**

*Let $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$, where $\mathbf{E}$ is i.i.d. mean zero with variance $\sigma^2$. Then given $v_i^T \in \mathbb{R}^d$ such that. $\|v_i - \xi_i \mathbf{V}_{i,*}^*\|_2 \leq \epsilon$ for some unknown $\xi_i \in \{1, -1\}$ and $i = 1, 2, \ldots, k$, where $\epsilon = O(\frac{1}{poly(d,m,k,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))})$ is sufficiently small, with high probability, Algorithm 7 returns $\mathbf{V}$ such that $\|\mathbf{V} - \mathbf{V}^*\|_2 \leq \epsilon$ in poly$(d, m, k, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$ time.*

**Proof** Consider a fixed $i \in [k]$, and WLOG assume $\xi_i = 1$, so we have $\|v_i - \mathbf{V}_{i,*}^*\|_2 \leq \epsilon = O(\frac{1}{\text{poly}(d,m,k,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*))})$. We show $a_i^+ < a_i^-$ with high probability. Now fix a row $j$ of $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}}) + \overline{\mathbf{E}}$ as in Algorithm 7, where $\overline{\mathbf{Q}}$ refers to restricting to the first $\ell = \text{poly}(k, d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$ columns of a matrix $\mathbf{Q}$. Note that we choose $\epsilon$ so that $\epsilon < 1/\text{poly}(\ell)$ (which is achieved by taking $n$ sufficiently large). This row is given by $\overline{\mathbf{A}}_{j,*} = \mathbf{U}_{j,*}^* f(\mathbf{V}^*\overline{\mathbf{X}}) + \overline{\mathbf{E}}_{j,*}$. As in the proof of Theorem 48, using Lemma 47 to bound $\|\overline{\mathbf{X}}\|_2$, we have $\|f(v_i\overline{\mathbf{X}}) - f(\xi\mathbf{V}_{i,*}^*\overline{\mathbf{X}})\|_2 \leq O(\epsilon\sqrt{\ell})$. We now using Lemma 57 to bound the projection difference between using approximate projection matrix $\mathbf{P}_{S_{i,+}}$ formed by our approximate vectors $f(v_i\overline{\mathbf{X}})$, and the true projection matrix $\mathbf{P}_{S_{i,+}}^*$ formed by the vectors $f(\mathbf{V}_{i,*}^*\overline{\mathbf{X}})$. By Lemma 58, the condition number and the spectral

norm of the matrix formed by the rows that span $\mathbf{P}^*_{S_{i,+}}$ are at most $O(r\text{poly}(k)\kappa^2)$ and $O(r\ell) = \text{poly}(k,d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*),\sigma)$ respectively, w.h.p. (in $k,d,m$). Setting $\epsilon = \epsilon'/\text{poly}(k,d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*),\sigma)$ sufficiently small, Lemma 57 gives $\|x\mathbf{P}_{S_{i,+}}\|_2 = \|x\mathbf{P}^*_{S_{i,+}}\|_2 \pm O(\epsilon'\|x\|_2)$ for $\epsilon' = \frac{1}{\text{poly}(k,d,m,\kappa(\mathbf{V}^*),\kappa(\mathbf{U}^*),\sigma}$ and any vector $x$.

Now we have $a^+_{i,j} = (\|\overline{\mathbf{E}_{j,*}}(\mathbb{I} - \mathbf{P}_{S_{i,+}})\|_2 \pm O(\epsilon'\sigma\sqrt{\ell}))^2 \leq \|\overline{\mathbf{E}_{j,*}}\|_2^2 + \pm O(\sigma^2\epsilon'\ell\text{poly}(k,d,m))$. Here we used that $\|\mathbf{E}\|_2^2 \leq \sigma^2\ell\text{poly}(k,d,m)$, w.h.p. in $k,d,m$ (by Chebyshev's inequality), and the fact that w.h.p. we have $\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\|_2 = O(\sigma\sqrt{\ell})$. Then, setting $\epsilon' = \epsilon''/\text{poly}(\ell)$, we have

$$\begin{aligned}
(a^-_{i,j})^2 &= \|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})(\mathbb{I} - \mathbf{P}_{S_{i,-}})\|_2^2 \pm O(\epsilon'') \\
&= \|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\|_2^2 - \|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\mathbf{P}_{S_{i,-}}\|_2^2 \pm O(\epsilon'') \\
&\geq \|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\|_2^2 - \left(\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\mathbf{P}_{S_{i,-}}\|_2 + \|\|\overline{\mathbf{E}_{j,*}}\mathbf{P}_{S_{i,-}}\|_2\right)^2 \pm O(\epsilon'')
\end{aligned}$$

where the second equality follows by the Pythagorean Theorem. Applying Lemma 56 and Corollary 59, writing $\kappa = \kappa(\mathbf{V}^*)$, with probability $1-\delta$ we have

$$\begin{aligned}
(a^-_{i,j})^2 \geq &\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\|_2^2 \\
&- \left(\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2(1 - \Omega(\frac{1}{\kappa^2\text{poly}(k)})) + 2\sigma\sqrt{k/\delta}\right)^2 \pm O(\epsilon'')
\end{aligned}$$

Setting $\delta < 1/\text{poly}(k,d,m)$ to get high probability gives

$$\begin{aligned}
(a^-_{i,j})^2 \geq &\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{j,*}})\|_2^2 - \|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2(1 - \Omega(\frac{1}{\kappa^2\text{poly}(k)})) \\
&- 4\sigma\sqrt{\frac{k}{\delta}}\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 \pm O(\epsilon'\sqrt{\ell}\sigma + \frac{\sigma^2 k}{\delta}) \\
&= \Omega(\frac{1}{\kappa^2\text{poly}(k)})\|\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 + \|\overline{\mathbf{E}_{j,*}}\|_2^2 + 2\langle\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle \\
&- 4\sigma\sqrt{k/\delta}\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 \pm O(\epsilon'\sqrt{\ell}\sigma + \frac{\sigma^2 k}{\delta})
\end{aligned}$$

Thus,

$$\begin{aligned}
a^-_i - a^+_i > &\sum_{j\in[m]} \Omega(\frac{1}{\kappa^2\text{poly}(k)})\|\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 + 2\langle\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle \\
&- 4\sigma\sqrt{k/\delta}\|(\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 \pm O(\epsilon'\sqrt{\ell}\sigma + \frac{\sigma^2 k}{\delta})
\end{aligned}$$

By Chebyshev's inequality, we have, $|2\langle\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle| < \text{poly}(dkm)\sigma\|\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2$ w.h.p. in $d,k,m$. Thus $\sum_j |2\langle\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle| < \text{poly}(dkm)\sigma\|\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2$. Now $\|\mathbf{U}^*_{*,i}\|_2 > 1/\kappa(\mathbf{U}^*)$, otherwise we would have $\|\mathbf{U}^*e_i\|_2 < 1/\kappa(\mathbf{U}^*)$, which is impossible by definition. Thus there is at least one entry of $\mathbf{U}^*_{*,i}$ with magnitude at least $1/(m\kappa(\mathbf{U}^*))$. So

$$\begin{aligned}
\sum_j \|\mathbf{U}^*_{j,i}f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 &\geq \frac{1}{m\kappa(\mathbf{U}^*)}\|f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 \\
&= \Omega(\ell\frac{1}{m\kappa(\mathbf{U}^*)})
\end{aligned}$$

where the last bound follows via bounds on $\chi^2$ variables Laurent and Massart (2000).

The above paragraph also demonstrates that $\frac{|2\langle \mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle|}{\|\mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2} \leq \frac{\text{poly}(dkm)\sigma}{\sqrt{\ell}}$, so taking $\ell$ sufficiently large this is less than $1/2$. Thus

$$\sum_{j\in[m]} \Omega(\frac{1}{\kappa^2 \text{poly}(k)})\|\mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 + 2\langle \mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}}), \overline{\mathbf{E}_{j,*}}\rangle > \frac{1}{2}\sum_{j\in[m]} \Omega(\frac{1}{\kappa^2 \text{poly}(k)})\|\mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2$$

and we are left with

$$\begin{aligned} a_i^- - a_i^+ &> \sum_{j\in[m]} \Omega(\frac{1}{\kappa^2 \text{poly}(k)})\|\mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2^2 - 4\sigma\sqrt{k/\delta}\|(\mathbf{U}^*_{j,i} f(\mathbf{V}^*_{i,*}\overline{\mathbf{X}})\|_2 - O(\epsilon'\sqrt{\ell}\sigma + \frac{\sigma^2 k}{\delta}) \\ &\geq \Omega(\ell \frac{1}{m\kappa^2\kappa(\mathbf{U}^*)\text{poly}(k)}) - O(\sigma\sqrt{k\ell/\delta} + \epsilon'\sqrt{\ell}\sigma + \frac{\sigma^2 k}{\delta}) \end{aligned}$$

Taking $\delta = 1/\text{poly}(d,k,m)$ as before and $\ell$ sufficiently larger than $1/\delta^2$, the above becomes $a_i^- - a_i^+ = \Omega(\ell \frac{1}{m\kappa^2\kappa(\mathbf{U}^*)\text{poly}(k)}) = \omega(1)$ w.h.p. in $d, k, m$. Thus the algorithm correctly determines $\xi_i = 1$ after seeing $a_i^- > a_i^+$, and the analysis is symmetric in the case that $xi_i = -1$.

■

### D.2. Recovering the Weights $\mathbf{U}^*, \mathbf{V}^*$

We have now shown in the prior section that given approximate $\mathbf{V}^*_{i,*}$'s where the signs $\xi_i$ are unknown, we can recover the signs exactly in polynomial time in the noisy setting. Thus we recover $\mathbf{V}$ such that $\|\mathbf{V} - \mathbf{V}^*_{i,*}\|_2 \leq \epsilon$ for some polynomially small $\epsilon$. To complete our algorithm, we simply find $\mathbf{U}^*$ by solving the linear regression problem

$$\min_{\mathbf{U}} \|\mathbf{U} f(\mathbf{V}\mathbf{X}) - \mathbf{A}\|_F$$

It is well know that the solution to the above regression problem can be solved by computing the pseduoinverse of the matrix $f(\mathbf{V}\mathbf{X})$, thus the entire procedure can be carried out in polynomial time. The following lemma states that, even in the presence of noise, the solution $\mathbf{U}$ from the regression problem must in fact be very close to the true solution $\mathbf{U}^*$.

**Lemma 61** *Let $\overline{\mathbf{X}}$ be the first $\ell = \text{poly}(k, d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$ columns of $\mathbf{X}$, and similarly define $\overline{\mathbf{E}}$. Set $\overline{\mathbf{A}} = \mathbf{U}^* f(\mathbf{V}^*\overline{\mathbf{X}}) + \overline{\mathbf{E}}$. Then given $\mathbf{V}$ such that $\|\mathbf{V} - \mathbf{V}^*\|_2 \leq \epsilon$ where $\epsilon = \frac{\epsilon'}{\text{poly}(\ell)}$ for some $\epsilon' > 0$, if $\mathbf{U}$ is the regression solution to $\min_{\mathbf{U}} \|\mathbf{U} f(\mathbf{V}\overline{\mathbf{X}}) - \overline{\mathbf{A}}\|_2$, then $\|\mathbf{U} - \mathbf{U}^*\|_2 < O(\epsilon')$ with high probability in $m, d$.*

**Proof** Note $\|f(\mathbf{V}\overline{\mathbf{X}}) - f(\mathbf{V}^*\overline{\mathbf{X}})\|_2 \leq O(1)\sqrt{\ell}\epsilon$ by standard operator norm bounds on Gaussian matrices (see Lemma 47), and the fact that $|a-b| > |f(a) - f(b)|$ for any $a, b \in \mathbb{R}$. The regression problem is solved row by row, so fix a row $i \in [m]$ and consider $\min_u \|uf(\mathbf{V}\overline{\mathbf{X}}) - \overline{\mathbf{A}}_{i,*}\|_2 = \min_u \|uf(\mathbf{V}\overline{\mathbf{X}}) - (\mathbf{U}^*_{i,*} f(\mathbf{V}^*\overline{\mathbf{X}}) + \overline{\mathbf{E}}_{i,*})\|_2 = \|uf(\mathbf{V}\overline{\mathbf{X}}) - (\mathbf{U}^*_{i,*} f(\mathbf{V}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{i,*}} + \mathbf{U}^*_{i,*}\mathbf{Z})\|_2$, where

$\mathbf{Z}$ is a matrix such that $\|\mathbf{Z}\|_F \leq O(1)\sqrt{\ell}\epsilon$. Now by the normal equations[4], if $u^*$ is the above optimizer, we have

$$\begin{aligned} u^* &= \Big(\mathbf{U}^*_{i,*} f(\mathbf{V}\overline{\mathbf{X}}) + \overline{\mathbf{E}_{i,*}} + \mathbf{U}^*_{i,*}\mathbf{Z}\Big) f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1} \\ &= \mathbf{U}^*_{i,*} + \Big(\overline{\mathbf{E}_{i,*}} + \mathbf{U}^*_{i,*}\mathbf{Z}\Big) f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1} \\ &= \mathbf{U}^*_{i,*} + \overline{\mathbf{E}_{i,*}} f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1} + \mathbf{U}^*_{i,*}\mathbf{Z}' \end{aligned}$$

Where $\mathbf{Z}'$ is a matrix such that $\|\mathbf{Z}'\|_F = O\big(\frac{\sqrt{\ell}\epsilon\kappa(f(\mathbf{V}\overline{\mathbf{X}})}{\sigma_{\min}(f(\mathbf{V}\overline{\mathbf{X}}))}\big)$. Note that we can scale $\overline{\mathbf{A}}$ at the beginning so that no entry is larger than $\ell^2$, which implies w.h.p. that each row of $\mathbf{U}^*$ has norm at most $\ell^2$. Thus

$$\|\mathbf{U}_{i,*}\mathbf{Z}'\|_F = O\big(\frac{\ell^{5/2}\epsilon\kappa(f(\mathbf{V}\overline{\mathbf{X}})}{\sigma_{\min}(f(\mathbf{V}\overline{\mathbf{X}}))}\big)$$

Now note $\mathbb{E}[\|\overline{\mathbf{E}_{i,*}} f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1}\|_2^2] = O(\sqrt{\ell k}\frac{\sigma^2}{\sigma^2_{\min}(f(\mathbf{V}\overline{\mathbf{X}}))})$ using $\|f(\mathbf{V}\overline{\mathbf{X}}\|_2 = O(\sqrt{\ell k})$ by the same operator norm bounds as before. By Markov bounds, w.h.p. in $m, d$, we have

$$\|\overline{\mathbf{E}_{i,*}} f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1}\|_2^2 = O(\sqrt{\ell}\text{poly}(d,m)\frac{\sigma^2}{\sigma^2_{\min}(f(\mathbf{V}\overline{\mathbf{X}}))})$$

Now by Courant-Fischer theorem and application of the triangle inequality, we have $\sigma_{\min}(f(\mathbf{V}\overline{\mathbf{X}})) > \sigma_{\min}(f(\mathbf{V}^*\overline{\mathbf{X}})) - O(\sqrt{\ell}\epsilon)$, and by Lemma 58 (see Section D.1), we have $\sigma_{\min}(f(\mathbf{V}^*\overline{\mathbf{X}})) = \Omega(\frac{\sqrt{\ell}}{\kappa(\mathbf{V}^*)\text{poly}(k)})$, thus for $\ell$ sufficiently large we obtain $\sigma_{\min}(f(\mathbf{V}\overline{\mathbf{X}})) = \Omega(\frac{\sqrt{\ell}}{\kappa(\mathbf{V}^*)\text{poly}(k)})$, in which case we have

$$\|\overline{\mathbf{E}_{i,*}} f(\mathbf{V}\overline{\mathbf{X}})^T \big[f(\mathbf{V}\overline{\mathbf{X}}) f(\mathbf{V}\overline{\mathbf{X}})^T\big]^{-1}\|_2^2 = O\Big(\text{poly}(d,m)\frac{\sigma^2}{\sqrt{\ell}}\Big)$$

Setting $\ell, 1/\epsilon$ to be sufficiently large polynomials in $(d, m, k, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$, we obtain

$$\|u^* - \mathbf{U}^*_{i,*}\|_2 \leq O(\epsilon'/\sqrt{m})$$

from which the Lemma follows. ■

We now state our main theorem for recovery of the weight matrices in the noisey case.

**Theorem 62** *Let $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ be given, where $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ are rank-$k$ and $\mathbf{E}$ is a matrix of i.i.d. mean zero subgaussian random variables with variance $\sigma^2$. Then given $n = \Omega\Big(\text{poly}\big(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma, \frac{1}{\epsilon}\big)\Big)$, there is an algorithm that runs in poly$(n)$ time and w.h.p. outputs $\mathbf{V}, \mathbf{U}$ such that*

$$\|\mathbf{U} - \mathbf{U}^*\|_F \leq \epsilon \qquad \|\mathbf{V} - \mathbf{V}^*\|_F \leq \epsilon$$

**Proof** The proof of correctness of the Tensor Decomposition based approximate recovery of $\mathbf{V}^*$ up to the signs is the same as in the exact case, via Theorem 40. By Theorem 60, we can recover the signs $\xi_i$, and thus recover $\mathbf{V}$ so that $\|\mathbf{V} - \mathbf{V}^*\|_F \leq \epsilon$. Observe that while the results in Section D.1

4. See https://en.wikipedia.org/wiki/Linear_least_squares

were stated for $\epsilon = \Theta\left(\frac{1}{\text{poly}(d,m,\kappa(\mathbf{U}^*),\kappa(\mathbf{V}^*)\sigma)}\right)$, they can naturally be generalized to any $\epsilon$ which is at least this small by increasing $n$ by a $\text{poly}(1/\epsilon)$ factor before running the tensor decomposition algorithm. Then by Lemma 61, we can recover $\mathbf{U}$ in polynomial time such that $\|\mathbf{U} - \mathbf{U}^*\|_F \leq \epsilon$ as desired, which completes the proof. ■

**Remark 63** *As in Remark 43, we have implicitly normalized the entire matrix $\mathbf{A}$ so that the columns of $\mathbf{A}$ have at most unit norm. If one seeks bounds for the recovery of the unnormalized $\mathbf{U}^*$, the error becomes $\|\mathbf{U} - \mathbf{U}^*\|_F \leq \epsilon\|\mathbf{U}^*\|_2$. To see why this holds, note that the normalization factor of Remark 43 is at least $\Omega(\frac{1}{\|\mathbf{U}^*\|_2+\sqrt{m\log(\ell)}})$, where $\ell = poly(d, m, \kappa(\mathbf{U}^*), \kappa(\mathbf{V}^*), \sigma)$ is as in Section D.1, and $O(\sqrt{m\log(\ell)})$ is a bound on the max column norm of $\mathbf{E}$ by subgaussian concentration. Thus multiplying by the inverse of this normalization factor blows up the error to $\|\mathbf{U}^*\|_2\epsilon$ after scaling $\epsilon$ down by a polynomial factor.*

**Algorithm 8 : FPTExactNeuralNet$(\mathbf{V}', \mathbf{X}, \mathcal{S})$.**

`Input:` Matrices $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) \in \mathbb{R}^{d\times n}$ and $\mathbf{X} \in \mathbb{R}^{r\times n}$ such that each entry in $\mathbf{X} \sim \mathcal{N}(0, 1)$.

1. Find a subset $S$ of columns of non-zero columns of $\mathbf{A}$ such that each for each $i \in S$ there is a $j \in S$, $j \neq i$, with $\mathbf{A}_{*,i} = c\mathbf{A}_{*,j}$ for some $c > 0$.
2. Partition $S$ into $S_r$ for $r \in [k]$ such that for each pair $i, j \in S_r$, $i \neq j$, we have $\mathbf{A}_{*,i} = c\mathbf{A}_{*,j}$ for some $c \in \mathbb{R}^{\neq 0}$.
3. For each $i \in [k]$, choose a representative $j_i \in S_i$, and let $\mathbf{U}_{*,i} = \mathbf{A}_{*,j_i}$. For each $j \in S_i$, let $c_{i,j}$ be such that $c_{i,j}\mathbf{U}_{*,i} = \mathbf{A}_{*,j}$.
4. let $\mathbf{W}$ be the matrix where the $i$-th row is given by the solution $w_i$ to the following linear system:
$$\forall i \in [k]: \quad w_i\mathbf{X}_{*,j} = c_{i,j} \quad \text{if } j \in S_i$$
5. Set $\mathbf{V}_{i,*} = \mathbf{W}_{i,*}/\|\mathbf{W}_{i,*}\|_2$, and let $\mathbf{U}$ be the solution to the following linear system:
$$\mathbf{U}f(\mathbf{V}\mathbf{X}) = \mathbf{A}$$

`Output:` $(\mathbf{U}, \mathbf{V})$.

## Appendix E. A Fixed-Parameter Tractable Exact Algorithm for Arbitrary $\mathbf{U}^*$

In the prior sections, we required that $\mathbf{U}^* \in \mathbb{R}^{m\times k}$ have rank $k$ in order to recover it properly. Of course, this is a natural assumption, making $\mathbf{U}^*$ identifiable. In this section, however, we show that

even when $m < k$ and $\mathbf{U}^*$ does not have full column rank, we can still recover $\mathbf{U}^*\mathbf{V}^*$ exactly in the noiseless case where we are given $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and $\mathbf{X}$, as long as the no two columns of $\mathbf{U}^*$ are non-negative scalar multiples of each other. Observe that this excludes columns from being entirely zero, but allows for columns of the form $[u, -u]$ for for $u \in \mathbb{R}^m$, as long as $u$ is non-zero. Our algorithm requires $n = \text{poly}(d, k)\kappa^{\Omega(k)}$, samples, and runs in time $O(n\text{poly}(d, m, k))$. Here $\kappa = \kappa(\mathbf{V}^*)$ is the condition number of $\mathbf{V}^*$. Our algorithm does not have any dependency on the condition number of $\mathbf{U}^*$.

The runtime of our algorithm is polynomial in the sample complexity $n$ and the size of the networks $d, k$, but simply requires $\text{poly}(d, k)\kappa^{\Omega(k)}$ samples in order to obtain columns of $f(\mathbf{V}^*\mathbf{X})$ which are 1-sparse, in which case the corresponding column of $\mathbf{A}$ will be precisely a positive scaling of a column of $\mathbf{U}^*$. In this way, we are able to progressively recover each column of $\mathbf{U}^*$ simply by finding columns of $\mathbf{A}$ which are scalar multiples of each other. The full algorithm, Algorithm 8, is given formally below.

**Lemma 64** *For each $i \in [k]$, with probability $1 - \delta$, at least $d$ columns of $f(\mathbf{V}^*\overline{\mathbf{X}})$ are positive scalings of $e_i^T$, where $\overline{\mathbf{X}}$ is the first $\ell$ columns of $\mathbf{X}$ for $n = \Omega(d \log(k/\delta)\kappa^{O(k)})$. In other words, $|S_i| \geq d$.*

**Proof** Let $\kappa = \kappa(\mathbf{V}^*)$. As in the proof of Lemma 58, we can assume that $\mathbf{V}^*$ is lower triangular by rotating the rows by a matrix $\mathbf{R}$, and noting that $\mathbf{RX}$ has the same distribution as $\mathbf{X}$ by the rotational invariance of Gaussians. We now claim that $\Pr[\|\mathbf{V}^* g\|_2 < \frac{1}{k\kappa}] = \Omega((\frac{1}{k\kappa})^k)$, where $g \sim \mathcal{N}(0, \mathbb{I}_d)$ is a Gaussian vector. To see this, since $\mathbf{V}^*$ is rank $k$ and in lower triangular form, $\mathbf{V}^*$ is supported on its the first $k$ columns. Thus it suffices to compute the value $\|\mathbf{V}^* g\|_2$ were $g \in \mathbb{R}^k$ is a $k$-dimensional Gaussian. By the anti-concentration of Gaussian, each $g_i < \frac{1}{k\kappa}$ with probability at least $\Omega(1/(k\kappa))$. Since the entries are independent, it follows that $\Pr[\|g\|_2 \leq \frac{1}{\sqrt{k}\kappa}] = \Omega(1/(k\kappa)^k)$. Let $\mathcal{E}_1$ be the event that this occurs. Since $\mathbf{V}^*$ has unit norm rows, it follows by Cauchy-Schwartz that conditioned on $\mathcal{E}_1$, we have $\tilde{g} = \mathbf{V}^* g$ satisfies $\|\tilde{g}\|_2 = O(\frac{1}{\kappa})$

Now consider the pdf of the $k$-dimensional multivariate Gaussian $\tilde{g}$ that has covariance $\Sigma = \mathbf{V}^*(\mathbf{V}^*)^T$, which is given by

$$p(x) = \frac{\exp\big(-\frac{1}{2}x\Sigma^{-1}x\big)}{\sqrt{(2\pi)^k \det(\Sigma)}}$$

for $x \in \mathbb{R}^k$. Now condition on the event $\mathcal{E}_2$ that $\tilde{g}$ is contained within the ball $\mathcal{B}$ of radius $O(\frac{1}{\kappa})$ centered at 0. Since $\mathcal{E}_1$ implies $\mathcal{E}_2$, we have $\Pr[\mathcal{E}_2] = \Omega(1/(k\kappa)^k)$ Now the eigenvalues of $\Sigma$ are the squares of the singular values of $\mathbf{V}^*$, which are all between $1/\kappa$ and $\sqrt{k}$. So all eigenvalues of $\Sigma^{-1}$ are between $1/k$ and $\kappa^2$. Thus for all $x \in \mathcal{B}$, we have

$$\frac{1}{2} \leq \frac{1}{e^{1/2}} \leq \exp\big(-\frac{1}{2}x\Sigma^{-1}x\big) \leq 1$$

It follows that

$$\sup_{x,y \in \mathcal{B}} \frac{p(x)}{p(y)} \leq 2$$

Now let $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_{2^k}$ be the intersection of all $2^k$ orthants in $\mathbb{R}^k$ with $\mathcal{B}$. The above bound implies that

$$\max_{i,j \in [2^k]} \frac{\int_{\mathcal{O}_i} p(x)dx}{\int_{\mathcal{O}_j} p(y)dy} \leq 2$$

Thus conditioned on $\mathcal{E}_2$ for the i.i.d. gaussian vector $\tilde{g} \sim \mathcal{N}(0, \Sigma) \in \mathbb{R}^k$, the probability that $\tilde{g}$ is in a given $\mathcal{O}_i$ is at most twice the probability that $g$ is in $\mathcal{O}_j$ for any other $j$. Thus $\min_{i \in [2^k]} \Pr[\tilde{g} \in \mathcal{O}_i] > \frac{1}{2^{k+1}}$. Thus for any sign pattern $\mathcal{S}$ on $k$-dimensional vectors, and in particular for the sign partner $\mathcal{S}_i$ of $e_i$, the probability that $\tilde{g}$ has this sign pattern conditioned on $\mathcal{E}_2$ is at least $\frac{1}{2^{k+1}}$. Since $\Pr[\mathcal{E}_2] = \Omega(1/(k\kappa)^k)$, it follows that in $n = \Omega(d \log(k/\delta)(k\kappa)^{2k})$ repetitions, a scaling of $e_i$ will appear at least $d$ times in the columns of $f(\mathbf{V}^*\mathbf{X})$ with probability $1 - \delta/k$, and the Lemma follows by a union bound over $e_i$ for $i \in [k]$.

■

**Theorem 65** *Suppose $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ for $\mathbf{U}^* \in \mathbb{R}^{m \times k}$ for any $m \geq 1$ such that no two columns of $\mathbf{U}^*$ are non-negative scalar multiples of each other, and $\mathbf{V}^* \in \mathbb{R}^{k \times n}$ has rank$(\mathbf{V}^*) = k$, and $n > \kappa^{O(k)}$poly$(dkm)$. Then Algorithm 8 recovers $\mathbf{U}^*, \mathbf{V}^*$ exactly with high probability in time $\kappa^{O(k)}$poly$(d, k, m)$.*

**Proof** By Lemma 64, at least $d$ columns of $f(\mathbf{V}^*\mathbf{X})$ will be scalar multiples of $e_i$ for each $i$. Thus the set $S$ of indices, as defined in Step 2 of Algorithm 8, will contain each column of $\mathbf{U}^*$ as a column. It suffices to show that no two columns of $\overline{\mathbf{A}}$ can be scalar multiples of each other if they are not a scalar multiple of $\mathbf{U}^*$. To see this, if two columns of $f(\mathbf{V}^*\mathbf{X})$ were not 1-sparse, then the distribution of $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ on these columns is supported on a $t$-dimensional manifold living inside $\mathbb{R}^m$, for some $t \geq 2$. In particular, this manifold is the conic hull of at least $t' \geq t \geq 2$ columns of $\mathbf{U}^*$ (where $t'$ is the sparsity of the columns of $f(\mathbf{V}^*\mathbf{X})$. This follows from the fact that the conic hull of any subset of 2 columns of $\mathbf{U}^*$ is 2-dimensional, since no columns two of $\mathbf{U}^*$ are non-negative scalings of each other. Thus the probability that two draws from such a distribution lie within the same 1-dimensional subspace, which has measure 0 inside of any $t \geq 2$-dimensional conic hull, is therefore 0, which completes the claim.

To complete the proof of the theorem, by pulling a diagonal matix $\mathbf{D}$ through $f$, we can assume $\mathbf{U} = \mathbf{U}^*$. By construction then, $c_{i,j}$ is such that $(\mathbf{D}\mathbf{V}_{i,*}\mathbf{X}_{*,j}) = c_{i,j}$, as it is the scaling which takes $\mathbf{U}_{*,i}$ to $\mathbf{A}_{*,j}$. Thus $w_i$, as defined in step 4 of Algorithm 8, is the solution to a linear equation $w_i \mathbf{X}_{S_i} = c$ for some fixed vector $c$, where $\mathbf{X}_{S_i}$ is $\mathbf{X}$ restricted to the columns in $S_i$. Since $|S_i| \geq d$ by Lemma 64, to show that $w_i$ is unique it suffices for $\mathbf{X}_{S_i}$ to be full rank. But as argued in the proof of Theorem 36, any subset of $d$ columns of $\mathbf{X}$ will be rank $d$ and invertible with probability 1. Thus $w_i$ is unique, and must therefore be a scaling of $\mathbf{V}^*_{i,*}$, which we find by normalizing $w_i$ to have unit norm. After this normalization, we can renormalize $\mathbf{U}$, or simply solve a linear system for $\mathbf{U}$ as in Step 5 of Algorithm 8. By Lemma 16, $f(\mathbf{V}^*\mathbf{X})$ will have full rank w.h.p., so the resulting $\mathbf{U}$ will be unique and therefore equal to $\mathbf{U}^*$ as desired.

■

## Appendix F. A Fixed-Parameter Tractable Algorithm for Arbitrary Non-Adversarial Noise

In the noisy model, the observed matrix $\mathbf{A}$ is generated by a perturbation $\mathbf{E}$ of some neural network $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ with rank $k$ matrices $\mathbf{U}^* \in \mathbb{R}^{m \times k}, \mathbf{V} \in \mathbb{R}^{k \times d}$, and i.i.d. Gaussian $\mathcal{N}(0, 1)$ input $\mathbf{X} \in \mathbb{R}^{d \times n}$. Formally, we are given as input $\mathbf{X}$ and $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}\mathbf{X}) + \mathbf{E}$, which is a noisy observation of the underlying network $\mathbf{U}^* f(\mathbf{V}\mathbf{X})$, and tasked with recovering approximations to

this network. In Section D, we showed that approximate recovery of the weight matrices $\mathbf{U}^*, \mathbf{V}^*$ is possible in polynomial time when the matrix $\mathbf{E}$ was i.i.d. mean 0 and sub-Gaussian. In this section, we generalize our noise model substantially to include all error matrices $\mathbf{E}$ which *do not depend* on the input matrix $\mathbf{X}$. Our goal is then to obtain $\mathbf{U}, \mathbf{V}$ such that

$$\|\mathbf{U}f(\mathbf{V}\mathbf{X}) - \mathbf{A}\|_F \leq (1+\epsilon)\|\mathbf{E}\|_F$$

Thus we would like to be able to recover a good approximation to the observed input, where we are competing against the cost $\text{OPT} = \|\mathbf{A} - \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})\|_2 = \|\mathbf{E}\|_2$. Observe that this is a slightly different objective than before, where our goal was to recover the actual weights $\mathbf{U}^*, \mathbf{V}^*$ approximately. This is a product of the more general noise model we consider in this Section. The loss function here can be thought of as recovering $\mathbf{U}, \mathbf{V}$ which approximate the observed classification nearly as well as the optimal generative $\mathbf{U}^*, \mathbf{V}^*$ do. This is more similar to the empirical loss considered in other words Arora et al. (2016). The main result of this section is the development of a fixed parameter tractable algorithm which returns $\mathbf{U}, \mathbf{V}$ such that

$$\|\mathbf{A} - \mathbf{U}f(\mathbf{V}\mathbf{X})\|_F \leq \|\mathbf{E}\|_F + O\Big(\Big[\sigma_{\min}\epsilon\sqrt{nm}\|\mathbf{E}\|_2\Big]^{1/2}\Big) \tag{3}$$

Where $\sigma_{\max} = \sigma_{\max}(\mathbf{U}^*)$, and $\|\mathbf{E}\|_2$ is the spectral norm of $\mathbf{E}$. In this section, to avoid clustering, we will write $\sigma_{\max}, \sigma_{\min}$, and $\kappa$ to denote the singular values and condition number of $\mathbf{U}^*$. Our algorithm has no dependency on the condition number of $\mathbf{V}^*$. The runtime of our algorithm is $(\frac{\kappa}{\epsilon})^{O(k^2)}\text{poly}(n, r, d)$, which is fixed-parameter tractable in $k, \kappa, \frac{1}{\epsilon}$. Here the sample complexity $n$ satisfies $n = \Omega(\text{poly}(r, d, \kappa, \frac{1}{\epsilon}))$.

We remark that the above bound in Equation 3 may at first seem difficult to parse. Intuitively, this bound will be a $(1+\epsilon)$ multiplicative approximation whenever the Frobenius norm of $\mathbf{E}$ is roughly an $\sqrt{m}$ factor larger than the spectral norm–in other words, when the error $\mathbf{E}$ is relatively flat. Note that these bounds will hold when $\mathbf{E}$ is drawn from a very wide class of random matrices, including matrices with heavier tails (see Vershynin (2010) and discussion below). When this is not the case, and $\|\mathbf{E}\|_2 \approx \|\mathbf{E}\|_F$, then we lose an additive $\sqrt{m}$ factor in the error guarantee. Note that this can be compensated by scaling $\epsilon$ by a $\frac{1}{\sqrt{m}}$ factor, in which case we will get a $(1+\epsilon)$ multiplicative approximation for any $\mathbf{E}$ which is not too much smaller than $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ (meaning $\|\mathbf{E}\|_F = \Omega(\epsilon\|\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})\|_F)$). The runtime in this case will be $(m\kappa/\epsilon)^{O(k^2)}$, which is still $(\kappa/\epsilon)^{O(k^3)}$ whenever $m = O(2^k)$. Note that if the noise $\mathbf{E}$ becomes arbitrarily smaller than the signal $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, then the multiplicative approximation of Equation 3 degrades, and instead becomes an additive guarantee.

To see why this is a reasonable bound, we must first examine the normalizations implicit in our problem. As always we assume that $\mathbf{V}^*$ has unit norm rows. Using the 2-stability of Gaussians, we know that $\mathbb{E}[(\mathbf{V}^*\mathbf{X})_{i,j}^2] = 1$ for any $i, j \in [k] \times [n]$, and by symmetry of Gaussians we have that $\mathbb{E}[f(\mathbf{V}^*\mathbf{X})_{i,j}^2] = 1/2$. By linearity of expectation we have $\mathbb{E}[\|f(\mathbf{V}^*\mathbf{X})\|_F^2] = kn/2$. Since $\sigma_{\min}^2\|f(\mathbf{V}^*\mathbf{X})\|_F^2 \leq \|\mathbf{U}f(\mathbf{V}^*\mathbf{X})\|_F^2 \leq \sigma_{\max}^2\|f(\mathbf{V}^*\mathbf{X})\|_F^2$, it follows that

$$\frac{\sigma_{\min}^2(\mathbf{U})kn}{2} \leq \mathbb{E}[\|\mathbf{U}f(\mathbf{V}^*\mathbf{X})\|_F^2] \leq \frac{\sigma_{\max}^2(\mathbf{U})kn}{2}$$

Thus for the scale of the noise $\mathbf{E}$ to be within a $\Omega(1)$ factor of the average squared entry of $\mathbf{U}f(\mathbf{V}^*\mathbf{X})$ on average, we expect $\|\mathbf{E}\|_F = O(\sigma_{\max}\sqrt{nk})$ and $\|\mathbf{E}\|_F = \Omega(\sigma_{\min}\sqrt{nk})$

Now consider the case where $\mathbf{E}$ is a random matrix, coming from a very broad class of distributions. Since $\mathbf{E} \in \mathbb{R}^{m \times n}$ with $n >> m$, one of the main results of random matrix theory is that many such matrices are *approximately* isometries Vershynin (2010). Thus, for a such a random matrix $\mathbf{E}$ normalized to be within a constant of the signal, we will have $\|\mathbf{E}\|_2 = O(\sigma_{\max}\sqrt{\frac{nk}{m}})$. This gives

$$\|\mathbf{A} - \mathbf{U}f(\mathbf{V}^*\mathbf{X})\|_F \leq \|\mathbf{E}\|_F(1 + O(\epsilon))$$

after scaling $\epsilon$ by a quadratic factor. In general, we get multiplicative approximations whenever either the spectrum of $\mathbf{E}$ is relatively flat, or when we allow $(m\kappa/\epsilon)^{O(k^2)}$ runtime. Note that in both cases, for the above bound to be a $\|\mathbf{A} - \mathbf{U}f(\mathbf{V}^*\mathbf{X})\|_F \leq (1 + \epsilon)\|\mathbf{E}\|_F$ approximation, we must have $\|\mathbf{E}\|_F = \Omega(\epsilon\|\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})\|_F)$ as noted above. Otherwise, the error we are trying to compete against is too small when compared to the matrices in question to obtain a multiplicative approximation.

### F.1. Main Algorithm

Our algorithm is then formally given in Figure 9. Before presenting it, we first recall some fundamental tools of numerical linear algebra. First, we recall the notion of a subspace-embedding.

**Definition 66 (Subspace Embedding)** *Let $\mathbf{U} \in \mathbb{R}^{m \times k}$ be a rank-$k$ matrix and, let $\mathcal{F}$ be family of random matrices with $m$ columns, and let $\mathbf{S}$ be a random matrix sampled from $\mathcal{F}$. Then we say that $\mathbf{S}$ is a $(1 \pm \delta)$-$\ell_2$-subspace embedding for the column space of $\mathbf{U}$ if for all $x \in \mathbb{R}^k$,*

$$\|\mathbf{S}\mathbf{U}x\|_2 = (1 \pm \delta)\|\mathbf{U}x\|_2$$

Note in the above definition, $\mathbf{S}$ is a subspace embedding for the column span of $\mathbf{U}$, meaning for any other basis $\mathbf{U}'$ spanning the same columns as $\mathbf{U}$, we have that $\mathbf{S}$ is also a $(1 \pm \delta)$-$\ell_2$-subspace embedding for $\mathbf{U}'$. For brevity, we will generally say that $\mathbf{S}$ is a subspace embedding for a matrix $\mathbf{U}$, with the understanding that it is in fact a subspace embedding for *all* matrices with the same column span as $\mathbf{U}$. Note that if $\mathbf{S}$ is a subspace embedding for a rank-$k$ matrix $\mathbf{U}$ with largest and smallest singular values $\sigma_{\max}$ and $\sigma_{\min}$ respectively, then $\mathbf{S}\mathbf{U}$ is rank-$k$ with largest and smallest singular values each in the range $(1 \pm \delta)\sigma_{\max}$ and $(1 \pm \delta)\sigma_{\min}$ respectively. The former fact can be seen by the necessity that $\|\mathbf{S}\mathbf{U}x\|_2$ be non-zero for all non-zero $x \in \mathbb{R}^k$, and the latter by the fact that $\max_{x \in \mathbb{R}^k,\, \|x\|_2 = 1} \|\mathbf{S}\mathbf{U}x\|_2 = (1 \pm \delta) \max_{x \in \mathbb{R}^k,\, \|x\|_2 = 1} \|\mathbf{U}x\|_2 = \sigma_{\max}$, and the same bound holds replacing $\max$ with $\min$. Our algorithm will utilize the following common family of random matrices.

**Proposition 67 (Gaussian Subspace Embedding Sarlos (2006))** *Fix any rank $k$-matrix matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$, and let $\mathbf{S} \in \mathbb{R}^{c_1 k/\epsilon^2 \times m}$ be a random matrix where every entry is generated i.i.d. Gaussian $\mathcal{N}(0, 1/k)$, for some sufficiently large constant $c_1$. Then with probability $99/100$, $\mathbf{S}$ is a subspace embedding for $\mathbf{U}$.*

Our algorithm is then as follows. We first sketch the input matrix $\mathbf{A}$ by a $O(k) \times m$ Gaussian matrix $\mathbf{S}$, and condition on it being a subspace embedding for $\mathbf{U}$. We then left multiply by $\mathbf{S}$ to obtain $\mathbf{S}\mathbf{A} = \mathbf{S}\mathbf{U}f(\mathbf{V}\mathbf{X}) + \mathbf{S}\mathbf{E}$. Now we would like to ideally recover $f(\mathbf{V}\mathbf{X})$, and since $\mathbf{S}$ is a subspace embedding for $\mathbf{U}$, we know that $\mathbf{S}\mathbf{U}$ has full column rank and thus has an left inverse. Since we do not know $\mathbf{U}$, we must guess the left inverse $(\mathbf{S}\mathbf{U})^{-1} \in \mathbb{R}^{k \times O(k)}$ of $\mathbf{S}\mathbf{U}$. We generate

guesses $\mathbf{M}^i$ of $(\mathbf{SU})^{-1}$, and try each of them. For the right guess, we know that after left multiplying by $\mathbf{M}^i$ we will have $\mathbf{M}^i\mathbf{SA} = f(\mathbf{VX}) + \mathbf{M}^i\mathbf{SE} + \mathbf{Z}$, where $\mathbf{Z}$ is some error matrix which arises from our error in guessing $(\mathbf{SU})^{-1}$.

**Algorithm 9 : Learning Neural Nets with Gaussian Inputs and Arbitrary Non-Adversarial Noise$(\mathbf{A}, \mathbf{X})$.**

1. Generate a random matrix $\mathbf{S} \in \mathbb{R}^{c_1 k/\delta^2 \times m}$ of i.i.d. Gaussian $\mathcal{N}(0, 1/k)$ variables for some sufficiently large constant $c_1$ and $\delta = 1/10$.
2. Enumerate all $k \times c_1 k/\delta^2$ matrices $\mathbf{M}^1, \mathbf{M}^2, \ldots, \mathbf{M}^\nu$ with entries of the form $\frac{1}{\sigma_{\min}(\mathbf{U})}(1+\frac{\epsilon^4}{ck^4})^{-i}$ for integers $0 \leq i \leq c'k^8(1/\epsilon^8)\kappa^8$ for sufficiently large constants $c, c'$ and any $\frac{1}{\epsilon} > k$. Note that $\nu = 2^{O(k^2 \log(\frac{1}{\epsilon}k\kappa))}$.
3. For $i = 1, 2, \ldots, \nu$
   (a) Generate a matrix $\mathbf{G} \in \mathbb{R}^{k \times n}$ s.t. $\mathbf{G}$ consists of i.i.d. $\mathcal{N}\left(0, \Theta\left(\left(\frac{\epsilon^{-2}\kappa^2 k\|\mathbf{MSE}\|_F}{\sqrt{n}}\right)^2\right)\right)$ random variables. Note, we can guess the value $\|\mathbf{MSE}\|_F$ in $O(\log(n))$ powers of 2 around $\|\mathbf{MSA}\|_F$.
   (b) For each row $p \in [k]$ and $q \in [n]$, let $y_q = \mathtt{sign}(\mathbf{M}^i\mathbf{SA} + \mathbf{G})_{p,q}$.
   (c) For each $p = 1, 2, \ldots, [k]$, let $w_i^p$ be the solution to the following convex program:
   $$\max_{w,} \sum_{i=1}^{n} y_i \langle w, X_{*,i} \rangle$$
   $$\text{subject to } \|w\|_2^2 \leq 1$$
   (d) Let $\mathbf{V}^i \in \mathbb{R}^{k \times d}$ be the matrix with $p$-th row equal to $w_i^p$.
4. Let $\mathbf{U}$ and $\mathbf{V}^{i^*}$ be the matrices that achieve the minimum value of the linear regression problem
   $$\arg\min_{\mathbf{U}, \mathbf{V}^i} \|\mathbf{A} - \mathbf{U}f(\mathbf{V}^i\mathbf{X})\|_F^2$$

**Output:** $(\mathbf{U}, \mathbf{W}^{i^*})$.

We then observe that the signs of each row of this matrix can be thought of as labels to a noisy halfspace classification problem, where the sign of $(\mathbf{M}^i\mathbf{SA})_{p,q}$ is a noisy observation of the sign of $\langle \mathbf{V}_{p,*}, \mathbf{X}_{*,q} \rangle$. Using this fact, we then run a convex program to recover each row $\mathbf{V}_{p,*}$. In order for recovery to be possible, there must be some non-trivial correlation between the labeling of these signs, meaning the sign of $(\mathbf{M}^i\mathbf{SA})_{p,q}$, and the true sign of $\langle \mathbf{V}_{p,*}, \mathbf{X}_{*,q} \rangle$. In order to accomplish this, we must *spread out* the error $\mathbf{E}$ to allow the value of $\langle \mathbf{V}_{p,*}, \mathbf{X}_{*,q} \rangle$ to have an effect on the observed sign a non-trivial fraction of the time. We do this by adding a matrix $\mathbf{G}$ such that the $i$-th row $\mathbf{G}_{i,*}$ consists of i.i.d. $\mathcal{N}\big(0, \Theta\big(\big(\frac{\epsilon^{-2}\kappa^2 k\|\mathbf{MSE}\|_F}{\sqrt{n}}\big)^2\big)\big)$ random variables to $\mathbf{M}^i\mathbf{SA}$. We will simply guess the value $\|\mathbf{MSE}\|_F$ here in $O(\log(n))$ powers of 2 around $\|\mathbf{MSA}\|_F$. We prove a general theorem (Theorem 69) about the recovery of hyperplanes $v \in \mathbb{R}^d$ when given noisy labels from a

combination of ReLU observations, adversarial, and non-adversarial noise components. Finally, we solve for $\mathbf{U}$ by regression. The full procedure is described formally given in Algorithm 9.

### F.2. Analysis

First note that by our earlier bounds on the singular values of $\mathbf{X}$ (Proposition 47), we have $\|f(\mathbf{V}^*\mathbf{X})\|_F \leq O(\sqrt{nk})$, thus if $\|\mathbf{E}\|_F > \sigma_{\max}(\mathbf{U}^*)\frac{\sqrt{nk}}{\epsilon}$, we can simply return $\mathbf{U}^* = 0, \mathbf{V}^* = 0$, and obtain our desired competitive approximation with the cost $OPT = \|\mathbf{E}\|_F$. Thus, where can now assume that $\|\mathbf{E}\|_F < \sigma_{\max}(\mathbf{U}^*)\frac{\sqrt{nk}}{\epsilon}$.

By Proposition 67, with probability $99/100$ we have both that $\mathbf{SU}^*$ is rank-$k$ and that the largest and smallest singular values of $\mathbf{SU}^*$ are perturbed by at most a $(1 \pm \delta)$ factor, meaning $\sigma_{\max}(\mathbf{U}^*) = (1 \pm \delta)\sigma_{\max}(\mathbf{SU}^*)$ and $\sigma_{\min}(\mathbf{U}^*) = (1 \pm \delta)\sigma_{\in}(\mathbf{SU}^*)$, from which it follows that $\kappa(\mathbf{U}^*) = (1 \pm O(\delta))\kappa(\mathbf{SU}^*)$. Note that we can repeat the algorithm $O(n)$ times to obtain this result with probability $1 - \exp(-n)$ at least once by Hoeffding bounds. So we can now condition on this and assume the prior bounds on the singular values and rank of $\mathbf{SU}^*$. Thus we will now write $\sigma_{\max} = \sigma_{\max}(\mathbf{SU}^*)$, $\sigma_{\min} = \sigma_{\min}(\mathbf{SU}^*)$, and $\kappa = \kappa(\mathbf{SU}^*)$, with the understanding that these values have been perturbed by a $(1 \pm 3\delta) < (1 \pm 1/2)$ factor.

We can assume that we know $\kappa$ and $\sigma_{\min}(\mathbf{U}^*)$ up to a factor of 2 by guessing them in geometrically increasing intervals. Note that we can assume $\sigma_{\max}$ is within a $\text{poly}(n)$ factor of the largest column norm of $\mathbf{A}$, since otherwise $\|\mathbf{E}\|_F$ would necessarily be larger than $\sigma_{\max}(\mathbf{U}^*)\frac{\sqrt{nk}}{\epsilon}$. Given this column norm, we obtain an interval $[a, b] \subset \mathbb{R}$ $\frac{a}{b} = \text{poly}(n, \kappa)$, such that both $\kappa$ and $\sigma_{\min}(\mathbf{U}^*)$ must live inside $[a, b]$. Then we can make $O(\log^2(\frac{a}{b})) = O(\log^2(n\kappa))$ guesses to find $\kappa$ and $\sigma_{\min}(\mathbf{U}^*)$ up to a factor of 2. Thus guessing the correct approximations to $\kappa, \sigma_{\min}(\mathbf{U}^*)$ will not effect our run time bounds, since our overall complexity is already polynomial in $n$ and $\kappa$. Similarly, we can also guess the value of $\|\mathbf{MSE}\|_F$ up to a factor of 2 using $O(\log(n\kappa))$ guesses, as is needed in step $3a$ of Algorithm 9.

The following Proposition gives the error bound needed for the right guess of the inverse $(\mathbf{SU})^{-1}$

**Proposition 68** *On the correct guess of $\sigma_{\max}(\mathbf{SU}^*)$ (up to a constant factor of 2 error), there is an $i \in [\nu]$ such that $\mathbf{M}^i = (\mathbf{SU}^*)^{-1} + \mathbf{\Lambda}$ where $\|\mathbf{\Lambda}\|_\infty \leq \frac{\epsilon^4}{\sigma_{\min}\kappa^4 k^4}$.*

**Proof** First note that no entry in $(\mathbf{SU}^*)^{-1}$ can be greater than $\frac{1}{\sigma_{\min}}$ (since $\sigma_{\min}$ is the smallest singular value of $\mathbf{SU}^*$, and therefore $\frac{1}{\sigma_{\min}}$ is the largest singular value of $(\mathbf{SU}^*)^{-1}$. Thus there is a guess of $\mathbf{M}^i$ such that for each entry $(p, q)$ of $(\mathbf{SU}^*)^{-1}$ in the range $(\frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4k^4}, \frac{1}{\sigma_{\min}})$, we have $\mathbf{M}^i_{p,q} = (\mathbf{SU}^*)^{-1}_{p,q}(1 \pm \frac{1}{(1/\epsilon)^4\kappa^4k^4}) = (\mathbf{SU}^*)^{-1} \pm \frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4k^4}$. For all other entries less than $\frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4k^4}$, we get $\mathbf{M}^i_{p,q} = (\mathbf{SU}^*)^{-1}_{p,q} \pm \frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4k^4}$ by setting $\mathbf{M}^i_{p,q} = \frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4k^4}$ (which is the lowest guess of value which we make for the coordinates of $\mathbf{M}^i$), from which the proposition follows. ■

### F.3. Learning Noisy Halfspaces:

By Proposition 68, we know that for the correct guess of $\mathbf{M}^i$ we can write $\mathbf{M}^i\mathbf{SA} = f(\mathbf{V}^*\mathbf{X}) + (\mathbf{M}^i\mathbf{SE}) + \mathbf{Z}$ where $\mathbf{Z} = \mathbf{\Lambda}\mathbf{SU}^*f(\mathbf{V}^*\mathbf{X})$. Thus $\mathbf{M}^i\mathbf{SA}$ can be thought of as a noisy version

of $f(\mathbf{V}^*\mathbf{X})$. We observe now that our problem can be viewed as the problem of learning a half-space in $\mathbb{R}^d$ with noisy labels. Specifically, we are obtain examples of the form $\mathbf{X}_{*,q}$ with the label $y_q = \text{Sign}\big(f(\mathbf{V}^*_{p,*}\mathbf{X}_{*:q}) + (\mathbf{M}^i\mathbf{S}\mathbf{E})_{p,q} + \mathbf{Z}_{p,q}\big) \in \{1,-1\}$, and our goal is to recover $\mathbf{V}^*_{p,*}$ from these labeled examples $\{y_q\}$. Note that if the labeled examples were of the form $\mathbf{X}_{*,q}$ and $\text{Sign}(\langle \mathbf{V}^*_{p,*}, \mathbf{X}_{*,q}\rangle)$, then this would correspond to the noiseless learning problem for half-spaces. Unfortunately, our problem is not noiseless, as it will often be the case that $\text{Sign}\big(f(\mathbf{V}_{p,*}\mathbf{X}_{*:q}) + (\mathbf{M}^i\mathbf{S}\mathbf{E})_{p,q} + \mathbf{Z}_{p,q}\big) \neq \text{Sign}(\langle \mathbf{V}_{p,*}, \mathbf{X}_{*,q}\rangle)$ (in fact, this will happen very close to half of the time). We will demonstrate, however, that recovery of $\mathbf{V}_{p,*}$ is still possible by showing that there is a non-trivial correlation between the labels $y_q$ and the true sign. To do this, we show the following more general result.

**Theorem 69** *Given $n$ i.i.d. Gaussian examples $\mathbf{X} \in \mathbb{R}^{d\times n}$ with labels $y_q = \text{Sign}\big((f(\mathbf{V}\mathbf{X}) + \mathbf{G} + \mathbf{B})_{p,q}\big) \in \{1,-1\}$ where $\mathbf{G}$ is an arbitrary fixed matrix independent of $\mathbf{X}$, and $\mathbf{B}$ is any matrix such that $\|\mathbf{B}\|_F \leq \frac{\sqrt{n}}{\omega}$ for any $\omega = o(\sqrt{n})$. Then if $v_{p,*}$ is the solution to the convex program in step 3c of Figure 9 run on the inputs examples $\mathbf{X}$ and $\{y_q\}$, then with probability $1 - e^{-n^{1/2}/10}$ we have*

$$\|v_{p,*} - \mathbf{V}_{p,*}\|_2^2 = O\left(\sqrt{\omega}\frac{\|\mathbf{G}\|_F}{\sqrt{n}}\left(\frac{\sqrt{d}}{\sqrt{n}} + \frac{1}{n^{1/4}} + \frac{\log(\omega)}{\omega}\right)\right)$$

Before we prove the theorem, we first show that our setting fits into this model. Observe that in our setting, $\mathbf{G} = \mathbf{M}^i\mathbf{S}\mathbf{E}$, and $\mathbf{B} = \mathbf{Z}$. Note that the Gaussian matrix added in Step 3a of Algorithm 9 is a component of proof of Theorem 69, and different than the $\mathbf{G}$ here. Namely, for Theorem 69 to work, one must first add Gaussian matrix to *smear out* the fixed noise matrix $\mathbf{M}^i\mathbf{S}\mathbf{E}$. See the proof of Theorem 69 for further details. The following Proposition formally relates our setting to that of Theorem 69.

**Proposition 70** *We have $\|(\mathbf{M}^i\mathbf{S}\mathbf{E})\|_F = O(\frac{1}{\sigma_{\min}}\sqrt{m}\|\mathbf{E}\|_2)$, and $\|\mathbf{Z}\|_F = \|\mathbf{\Lambda}\mathbf{S}\mathbf{U}^* f(\mathbf{V}\mathbf{X})\|_2 \leq \sqrt{n}\frac{2}{(1/\epsilon)^4\kappa^4 k^2}$*

**Proof** Since $\mathbf{S}\mathbf{U}^*$ is $\kappa = \sigma_{\max}/\sigma_{\min}$ conditioned (as conditioned on by the success of $\mathbf{S}$ as a subspace embedding for $\mathbf{U}^*$), it follows that for any row $p$, we have $\|((\mathbf{S}\mathbf{U}^*)^{-1})_{p,*}\|_2 \leq \frac{1}{\sigma_{\min}}$. Thus by Proposition 68 we have $\|\mathbf{M}^i_{p,*}\|_2 \leq \frac{1}{\sigma_{\min}} + \frac{1}{\sigma_{\min}(1/\epsilon)^4\kappa^4 k^3} \leq \frac{2}{\sigma_{\min}}$, and by Proposition 47, noting that $\mathbf{S}$ can be written as a i.i.d. matrix of $\mathcal{N}(0,1)$ variables scaled by $\frac{1}{\sqrt{k}}$, we have $\|\mathbf{M}^i_{p,*}\mathbf{S}\|_2 \leq \frac{2}{\sigma_{\min}}\sqrt{\frac{2m}{k}}$. Applying this over all $O(k)$ rows, it follows that $\|\mathbf{M}^i\mathbf{S}\mathbf{E}\|_F = O(\frac{1}{\sigma_{\min}}\sqrt{m}\|\mathbf{E}\|_2)$, where $\|\mathbf{E}\|_2$ is the spectral norm of $\mathbf{E}$.

For the second, note the bound $\|\mathbf{\Lambda}\|_\infty \leq 1/(\sigma_{\min}(1/\epsilon)^4\kappa^4 k^4)$ from Proposition 68 implies that $\|\mathbf{\Lambda}_{p,*}\|_2 \leq 1/(\sigma_{\min}(1/\epsilon)^4\kappa^4 k^3)$ (using that $k > c_1/\delta$ where $\delta$ is as in Figure 9), so $\|\mathbf{\Lambda}_{p,*}\mathbf{S}\mathbf{U}^*\|_2 \leq \frac{\sigma_{\max}}{\sigma_{\min}(1/\epsilon)^4\kappa^4 k^3} \leq \frac{1}{(1/\epsilon)^4\kappa^4 k^3}$. Now by Proposition 47, we have that the largest singular value of $\mathbf{X}$ is at most $2\sqrt{n}$ with probability at least $1 - 2e^{-n/8}$, which we now condition on. Thus $\|\mathbf{V}\mathbf{X}\|_F \leq 2\sqrt{nk}$, from which it follows $\|f(\mathbf{V}\mathbf{X})\|_F \leq 2\sqrt{nk}$, giving $\|\mathbf{Z}_{p,*}\|_2 \leq 2\sqrt{n}\frac{1}{(1/\epsilon)^4\kappa^4 k^{5/2}}$ for every $p \in [k]$, so $\|\mathbf{Z}\|_F \leq \sqrt{n}\frac{2}{(1/\epsilon)^4\kappa^4 k^2}$ as needed. ■

By Theorem 69 and Proposition 70, we obtain the following result.

**Corollary 71** *Let $i$ be such that $\mathbf{M}^i = (\mathbf{S}\mathbf{U}^*)^{-1} + \mathbf{\Lambda}$, where $\|\mathbf{\Lambda}\|_\infty \leq 1/(\sigma_{\min}(1/\epsilon)^4\kappa^4 k^4)$ as in Proposition 68, and let $\mathbf{W}^i$ be the solution to the convex program as defined in Step 3d of the algorithm in Figure 9. Then with probability $1 - \exp(-\sqrt{n}/20)$, for every row $p \in [k]$ we have*

$$\|\mathbf{V}_{p,*} - \mathbf{W}^i_{p,*}\|_2^2 \leq \frac{\epsilon\sqrt{m}\|\mathbf{E}\|_2}{\sigma_{\min}\sqrt{n}}$$

**Proof** By Proposition 70 we can apply Theorem 69 with $\omega = \epsilon^{-4}\kappa^4 k^2$ and $\|\mathbf{G}\|_F = O(\frac{1}{\sigma_{\min}}\sqrt{m}\|\mathbf{E}\|_2)$, we obtain the stated result for a single row $p$ with probability at least $1 - e^{-\sqrt{n}/10}$ after taking $n = \text{poly}(\kappa, d)$ sufficiently large. Union bounding over all $k$ rows gives the desired result. ■

**Proof of Theorem 69** To prove the theorem, we will use techniques from Plan and Vershynin (2013). Let $v \in \mathbb{R}^d$ be fixed with $\|v\|_2 = 1$, and let $\mathbf{X} \in \mathbb{R}^{d\times n}$ be a matrix of i.i.d. Gaussian $\mathcal{N}(0,1)$ variables. Let $y_q$ be a noisy observation of the value $\text{sgn}(\langle v, \mathbf{X}_{*,q}\rangle)$, such that the $y_q$'s are independent for different $q$. We say that the $y_q$'s are *symmetric* if $\mathbb{E}[y_q \mid \mathbf{X}_{*,q}] = \theta_q(\langle v, \mathbf{X}_{*,q}\rangle)$ for each $q \in [n]$. In other words, the expectation of the noisy label $y_q$ given the value of the sample $\mathbf{X}_{*,q}$ depends only on the value of the inner product $\langle v, \mathbf{X}_{*,q}\rangle$. We consider now the following requirement relating to the correlation between $y_q$ and $\text{sgn}(\langle v, \mathbf{X}_{*,q}\rangle)$.

$$\mathbb{E}_{g\sim\mathcal{N}(0,1)}\Big[\theta_q(g)g\Big] = \lambda_q \geq 0 \tag{4}$$

Note that the Gaussian $g$ in Equation 4 can be replaced with the identically distributed variable $\langle v, X_{*,q}\rangle$. In this case, Equation 4 simply asserts that there is indeed some correlation between the observed labels $y_q$ and the ground truth $\text{sgn}(\langle v, \mathbf{X}_{*,q}\rangle)$. When this is the case, the following convex program is proposed in Plan and Vershynin (2013) for recovery of $v$

$$\max_{w,\, \|w\|_2\leq 1} \sum_{q=1}^{n} y_q\langle w, X_{*,q}\rangle \tag{5}$$

We remark that we must generalize the results of Plan and Vershynin (2013) here in order to account for $\theta_q$ depending on $q$. Namely, since $\mathbf{E}$ is not identically distribution, we must demonstrate bounds on the solution to the above convex program for the range of parameters $\{\lambda_q\}_{q\in[n]}$.

Now fix a row $p \in [k]$ and let $v = \mathbf{V}^*_{p,*}$. We will write $\mathbf{G}' = \mathbf{G} + \mathbf{G}''$, where $\mathbf{G}''$ is an i.i.d. Gaussian matrix distributed $(\mathbf{G}'')_{i,j} \sim \mathcal{N}(0, \eta^2)$ for all $i, j \in [k] \times [n]$, where $\eta = 100\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n}$. For technical reasons, we replace the matrix $\mathbf{G}$ with $\mathbf{G}'$ be generating and adding $\mathbf{G}''$ to our matrix $(f(\mathbf{V}\mathbf{X}) + \mathbf{G} + \mathbf{B})$. Then the setting of Theorem 69, we have $y_q = \text{Sign}((f(\mathbf{V}^*\mathbf{X}) + \mathbf{G}' + \mathbf{B})_{p,q})$. Note that by the definition of $\eta$, at most $\frac{n}{100\omega}$ entries in $\mathbf{G}$ can be larger than $\eta/10 = 10\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n}$. Let $\mathbf{B}'$ be the matrix of entries of $\mathbf{G}$ which do not satisfy this, so we instead write $y_q = \text{Sign}((f(\mathbf{V}^*\mathbf{X}) + \mathbf{G}' + \mathbf{B}' + \mathbf{B})_{p,q})$, where $\mathbf{G}' = \mathbf{G} + \mathbf{G}'' - \mathbf{B}'$. Thus $\mathbf{G}'_{p,q} \sim \mathcal{N}(\mu_{p,q}, \eta^2)$ where $\mu_{p,q} < 10\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n} = \eta/10$. Note that $\mathbf{B}'$ is $\frac{n}{100\omega}$ sparse, as just argued.

Note that the above model does not fully align with the aforementioned model, because $\mathbf{B}$ is an arbitrary matrix that can depend potentially on $f(\mathbf{V}^*\mathbf{X})$, and not just $\langle \mathbf{V}^*_{p,*}, \mathbf{X}_{*,q}\rangle$. So instead, suppose hypothetically that in the place of $y_q$ we were given the labels $y'_q = \text{Sign}((f(\mathbf{V}^*\mathbf{X}) +$

$\mathbf{G}')_{p,q})$, which indeed satisfies the above model. Note that we have also removed $\mathbf{B}'$ from the definition of $y'_q$, since we will handle it at the same time as we handle $\mathbf{B}$. In this case we can write $\mathbb{E}[y'_q \mid \mathbf{X}_{*,q}] = \mathbb{E}_g[\text{sign}(f(\langle \mathbf{X}_{*,q}, \mathbf{V}^*_{p,*}\rangle) + g_{p,q}) \mid \langle \mathbf{X}_{*,q}, \mathbf{V}_{p,*}\rangle]$ where $g_{p,q} \sim \mathcal{N}(\mathbf{G}_{p,q} - \mathbf{B}'_{p,q}, \eta^2)$ is a Gaussian independent of $X$.

Proposition 72 gives the corresponding value of $\lambda$ for this model.

**Proposition 72** *The function $\theta_q$ as defined by the hypothetical labels $y'_q$ satisfies Equation 4 with $\lambda_q \geq \frac{c}{\eta}$ for some constant $c > 0$, where $\eta = 100\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n}$.*

**Proof** We can write $\mathbb{E}[y'_q \mid \mathbf{X}_{*,q}] = \mathbb{E}[\text{sign}(f(\langle \mathbf{X}_{*,q}, \mathbf{V}_{p,*}\rangle) + g_{p,q}) \mid \langle \mathbf{X}_{*,q}, \mathbf{V}_{p,*}\rangle]$ where $g_{p,q} \sim \mathcal{N}(\mathbf{G}_{p,q} - \mathbf{B}_{p,q}, \eta^2)$. Let $\mu_q = \mathbf{G}_{p,q} - \mathbf{B}_{p,q}$ (for a fixed row $p$. Then $\theta(z) = 1 - 2\Pr[g \leq -f(z)]$, and Equation 4 can be evaluated by integration by parts. Let $p_q(z) = \frac{1}{\sqrt{2\pi\eta^2}} e^{-\frac{(z-\mu_q)^2}{2\eta^2}}$ is the p.d.f. of $g_{p,q}$. Note by the prior paragraphs we have $\eta^2 > 10\mu_q^2$ for all $q$. Then we have

$$\begin{aligned}
\lambda &= \mathbb{E}[\theta'(g)] = \mathbb{E}[2p(-f(z))] \\
&= \mathbb{E}_{z\sim\mathcal{N}(0,1)}\left[\sqrt{\frac{2}{\pi(\eta^2)}} e^{-(f(z)+\mu_q)^2/(2(\eta^2))}\right] \\
&= \sqrt{\frac{2}{\pi(\eta^2)}}\mathbb{E}_{z\sim\mathcal{N}(0,1)}\left[e^{-\frac{f(z)^2+2\mu_q f(z)+\mu_q^2}{2\eta^2}}\right] \\
&= \Omega(\frac{1}{\eta})
\end{aligned}$$

■

Now for any $z \in \mathbb{R}^d$ with $\|z\|_2 \leq 1$, let $h(z) = \frac{1}{n}\sum_{q=1}^n y_q\langle z, X_{*,q}\rangle$, and let $h'(z) = \frac{1}{n}\sum_{q=1}^n y'_q\langle z, X_{*,q}\rangle$. Observe that the hypothetical function $h'$ corresponds to the objective function of Equation 5 with values of $y'_q$ which satisfy the model of 4, whereas $h$, corresponding to the labels $y_q$ which we actually observe, does not. Let $B_2^d = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$ and let $\mathcal{B}_2^d = B - B = \{x - z \mid x, y \in B\}$ be the Minkowski difference. The following follows immediately from Plan and Vershynin (2013).

**Lemma 73 (Lemma 4.1 Plan and Vershynin (2013))** *For any $z \in B_2^d$, we have $\mathbb{E}[h'(z)] = \frac{1}{n}\sum_{q=1}^n \lambda_q\langle z, \mathbf{V}_{p,*}\rangle$ and thus because $h'$ is a linear function, we have*

$$\mathbb{E}[h'(\mathbf{V}_{p,*}) - h'(z)] = \mathbb{E}[h'(\mathbf{V}_{p,*} - z)] = \frac{1}{n}\sum_{q=1}^n \lambda_q(1 - \langle \mathbf{V}_{p,*}, z\rangle) \geq \frac{1}{n}\sum_{q=1}^n \frac{\lambda_q}{2}\|\mathbf{V}_{p,*} - z\|_2^2$$

We now cite Proposition 4.2 of Plan and Vershynin (2013). We remark that while the proposition is stated for the concentration of the value of $h'(z)$ around its expectation when the $\lambda_q$ are are all uniformly the same $\lambda_q = \lambda$, we observe that this fact has no bearing on the proof of Proposition 74 below. This is because only the $y_q \in \{1, -1\}$ depend on the $\lambda_q$'s, and the concentration result of Proposition 74, in fact, holds for *any* possible values of the $y_q$'s. Thus one could replace $h'(z)$ below with any function of the form $\hat{h}(z) = \frac{1}{n}\sum_{q=1}^n y_q\langle z, g_q\rangle$ for any values of $y_q \in \{1, -1\}$, and the following concentration result would hold as long as $\{g_q\}_{q\in[n]}$'s is a collection of independent $\mathcal{N}(0, \mathbb{I}_d)$ variables.

**Proposition 74 (Proposition 4.2 Plan and Vershynin (2013))** *For each $t > 0$, we have*

$$\Pr\Big[\sup_{z\in\mathcal{B}_2^d} \big|h'(z) - \mathbb{E}[h'(z)]\big| \geq \frac{4\sqrt{d}}{\sqrt{n}} + t\Big] \leq 4\exp(-\frac{-nt^2}{8})$$

We now demonstrate how to utilize these technical results in our setting. First, however, we must bound $\sup_{z\in B}|h'(z) - h(z)|$, since in actuality we will need bounds on the value of $h(z)$. We first introduce a bound on the expected number of flips between the signs $y_{p,*}$ and $y'_{p,*}$.

**Proposition 75** *Let $T = \{q \in [n] \mid y_q \neq y'_q\}$. Then with probability $1 - e^{-10\sqrt{n}}$, we have $|T| \leq 11\frac{n}{\omega}$.*

**Proof** We have $\|\mathbf{B}_{p,*}\|_1 \leq \sqrt{n}\|\mathbf{B}_{p,*}\|_2 \leq n/\omega$ by the original assumption on $\mathbf{B}$ in Theorem 69. Then $\Pr[q \in T]$ is at most the probability $\mathbf{G}'_{p,q}$ is in some interval of size $2|\mathbf{B}_{p,q}|$, which is at most $2|\mathbf{B}_{p,q}|$ by the anti-concentration of Gaussians. Thus $\mathbb{E}[|T|] \leq 2\|\mathbf{B}_{p,*}\|_1 \leq 2n/\omega$, and by Chernoff bounds $\Pr[|T| > 10n/\omega] < e^{-10\sqrt{n}}$ as needed. To handle $\mathbf{B}'$, we simple recall that $\mathbf{B}'$ was $\frac{n}{100\omega}$ sparse, and thus can flip at most $\frac{n}{100\omega} < n/\omega$ signs. ■

**Proposition 76** *Let $h, h'$ be defined as above. Let $\hat{w} \in B_2^r$ be the solution to the optimization problem*

$$\max_{w,\,\|w\|_2} n\,h(w) = \max_{w,\,\|w\|_2} \sum_{q=1}^{n} y_q\langle w, \mathbf{X}_{*,q}\rangle \tag{6}$$

*Then if with probability $1 - \exp(-\sqrt{n})$ we have*

$$\Pr\Big[\sup_{z\in B}\big|h'(z) - h(z)\big| \leq \frac{3\log(\omega)}{\omega}\Big] \geq 1 - e^{-\sqrt{n}}$$

**Proof** Let $S \subset \{x \in \mathbb{R}^d \mid \|x\|_\infty \leq 1\}$ be an $\epsilon$-net for $\epsilon = 1/n^3$. Standard results demonstrate the existence of $S$ with $|S| < 2^{12d\log(n)}$ (see e.g. Vershynin (2010); Woodruff et al. (2014)). Fix $z \in S$ and observe $|h'(z) - h(z)| = \frac{2}{n}\sum_{q\in T}|\langle z, \mathbf{X}_{*,q}\rangle|$. Note that we can assume $\|z\|_2 = 1$, since increasing the norm to be on the unit sphere can only make $|h'(z) - h(z)|$ larger. By Proposition 75, we have $|T| \leq n/\tau$, where $\tau = \frac{\omega}{11}$ with probability $1 - e^{-10\sqrt{n}}$, so we can let $\mathcal{F} = \{T' \subset [n] \mid |T'| \leq n/\tau\}$. Note $|\mathcal{F}| \leq n(e\tau)^{n/\tau}$. Fix $T' \in \mathcal{F}$. The sum $\sum_{q\in T'}|\langle z, \mathbf{X}_{*,q}\rangle|$ is distributed as the $L_1$ of a Gaussian $\mathcal{N}(0,1)$ vector in $|T'|$, dimensions, and is $\sqrt{|T'|}$-Lipschitz with respect to $L_2$, i.e. $|\|x\|_1 - \|y\|_1| \leq \|x - y\|_1 \leq \sqrt{|T'|}\|x - y\|_2$. So by Lipschitz concentration (see Vershynin (2010) (Proposition 5.34)), we have $\Pr[\frac{1}{n}\sum_{q\in T'}|\langle z, \mathbf{X}_{*,q}\rangle| > \frac{\log(e\tau)}{\tau}] \leq \exp(-\log^2(e\tau)n/\tau)$. We can then union bound over all $T' \in \mathcal{F}$ and $z \in S$ to obtain the result with probability

$$1 - \exp\Big(-\frac{n\log^2(e\tau)}{\tau} + \frac{n\log(e\tau)}{\tau} + \log(n) + 12r\log(n)\Big) > 1 - \exp\Big(-\log^2(\tau)n/(2\tau)\Big)$$

So let $\mathcal{E}_1$ be the event that $\sum_{q\in T'}|\langle z, \mathbf{X}_{*,q}\rangle| < \sqrt{\log(\tau)}|T'|$ for all $T' \in \mathcal{F}$ and $z \in S$. Now fix $w \in \mathbb{R}^d$ with $\|w\|_2 \leq 1$, and let $y \in S$ be such that $\|y - z\|_2 \leq 1/n^3$. Observing that $h$ and $h'$ are linear functions, we have $|h(z) - h'(z)| \leq |h(y) - h'(y)| + |h(z - y) - h'(z - y)| \leq$

$\frac{\log(e\tau)}{\tau} + |h(z-y) - h'(z-y)|$. Now condition on the event $\mathcal{E}_2$ that $\|X\|_F^2 \leq 10nd$, where $\Pr[\mathcal{E}_2] > 1 - \exp(nd)$ by standard concentration results for $\chi^2$ distributions Laurent and Massart (2000). Conditioned on $\mathcal{E}_2$ we have $|h(z-y)| + |h'(z-y)| \leq 4\sqrt{10nd}/n^3 \leq 1/\tau$, giving $|h(z) - h'(z)| \leq \frac{3\log(e\tau)}{\tau} < \frac{3\log(\omega)}{\omega}$, from which the proposition follows after union bounding over the events $\mathcal{E}_1, \mathcal{E}_2$ and Proposition 75, which hold with probability $1 - (\exp(-\log^2(\tau)n/(2\tau)) + \exp(nd) + \exp(-10\sqrt{n})) > 1 - \exp(-\sqrt{n})$.

■

**Lemma 77** *Let $\hat{w}$ be the solution to the optimization Problem in Equation 5 for our input labels $y_q = sign((f(\mathbf{VX}) + \mathbf{G}' + \mathbf{B} + \mathbf{B}')_{p,q})$. Then w with probability $1 - e^{-n^{1/2}/10}$, we have $\|\hat{w} - \mathbf{V}_{p,*}\|_2^2 = O(\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n})(\frac{4\sqrt{d}}{\sqrt{n}} + \frac{1}{n^{1/4}} + \frac{6\log(\omega)}{\omega})$ for some constant c.*

**Proof** Applying Lemma 73, and a union bound over the probabilities of failure in Proposition 74 with $t = n^{1/4}$ and Proposition 76, we have

$$\begin{aligned}
0 &\leq h(\hat{w}) - h(\mathbf{V}_{p,*}) \\
&\leq h'(\hat{w}) - h'(\mathbf{V}_{p,*}) + \frac{6\log(\omega)}{\omega} \\
&= h'\big(\hat{w} - \mathbf{V}_{p,*}\big) + \frac{6\log(\omega)}{\omega} \\
&\leq \mathbb{E}[h'\big(\hat{w} - \mathbf{V}_{p,*}\big)] + \frac{4\sqrt{d}}{\sqrt{n}} + \frac{1}{n^{1/4}} + \frac{6\log(\omega)}{\omega} \\
&\leq -\frac{\lambda}{2}\|\hat{w} - \mathbf{V}_{p,*}\|_2^2 + \frac{4\sqrt{d}}{\sqrt{n}} + \frac{1}{n^{1/4}} + \frac{6\log(\omega)}{\omega}
\end{aligned}$$

Applying Proposition 72, which yields $\frac{1}{\lambda} = O(\eta) = O(\sqrt{\omega}\|\mathbf{G}\|_F/\sqrt{n})$ completes the proof. ■

**Proof** [Proof of Theorem 69] The proof of the theorem follows directly from Lemma 77. ■

### F.4. Completing the Analysis

We will now need the following straightforward lemma to complete the proof.

**Theorem 78** *Let $\mathbf{A} = \mathbf{U}^* f(\mathbf{VX}) + \mathbf{E}$ be the input, where each entry of $\mathbf{X} \in \mathbb{R}^{d\times n}$ is i.i.d. $\mathcal{N}(0,1)$ and $\mathbf{E}$ independent of $\mathbf{X}$. Then the algorithm in Figure 9 outputs $\mathbf{U} \in \mathbb{R}^{m\times k}, \mathbf{V} \in \mathbb{R}^{k\times d}$ in time $2^{O(k^2\log((1/\epsilon)\kappa))}poly(n,d)$ such that with probability $1 - \exp(-\sqrt{n})$ we have*

$$\|\mathbf{A} - \mathbf{U}f(\mathbf{VX})\|_F \leq \|\mathbf{E}\|_F + O\Big(\Big[\sigma_{\min}\epsilon\sqrt{nm}\|\mathbf{E}\|_2\Big]^{1/2}\Big)$$

*Where $\|\mathbf{E}\|_2$ is the spectral norm of $\mathbf{E}$.*

**Proof** Let $\mathbf{W}^i \in \mathbb{R}^{k\times r}$ be as in Corollary 71. Then, taking $n = \text{poly}(d, \kappa, \frac{1}{\epsilon})$ large enough, we have $\mathbf{W}^i = \mathbf{V} + \mathbf{\Gamma}$ where $\|\mathbf{\Gamma}_{p,*}\|_F^2 \leq \frac{\epsilon k\sqrt{m}\|\mathbf{E}\|_2}{\sigma_{\min}\sqrt{n}}$ for each row $p$ with probability $1 - \exp(-r^4)$ by Corollary 71. Then applying the spectral norm bound on Gaussian matrices from Proposition

47, we obtain that $\|\mathbf{V}_{p,*}\mathbf{X} - \mathbf{W}^i_{p,*}\mathbf{X}\|_F^2 = O(\sqrt{n}\frac{\epsilon k\sqrt{m}\|\mathbf{E}\|_2}{\sigma_{\min}})$ with probability at least $1 - e^{-9n}$. Since $f$ just takes the maximum of 0 and the input, it follows that $\|f(\mathbf{W}^i\mathbf{X}) - f(\mathbf{V}\mathbf{X})\|_F^2 = O(\sqrt{n}\frac{\epsilon k\sqrt{m}\|\mathbf{E}\|_2}{\sigma_{\min}})$, and therefore $\|\mathbf{U}^* f(\mathbf{W}^i\mathbf{X}) - \mathbf{U}^* f(\mathbf{V}\mathbf{X})\|_F^2 = O(\sigma_{\max}^2\sqrt{n}\frac{\epsilon k\sqrt{m}\|\mathbf{E}\|_2}{\sigma_{\min}})$, which is at most $O(\sigma_{\min}\epsilon\sqrt{nm}\|\mathbf{E}\|_2)$ after rescaling $\epsilon$ by a $\frac{1}{\kappa^2 k}$ factor. Now if $\mathbf{U}$ is the minimizer to the regression problem $\min_{\mathbf{U}} \|\mathbf{A} - \mathbf{U}f(\mathbf{W}^i\mathbf{X})\|_F^2$ in step 5 of Figure 9, then note

$$\|\mathbf{A} - \mathbf{U}f(\mathbf{W}^i\mathbf{X})\|_F \leq \|\mathbf{A} - \mathbf{U}^* f(\mathbf{W}^i\mathbf{X})\|_F \leq \|\mathbf{E}\|_F + O\Big(\Big[\sigma_{\min}\epsilon\sqrt{nm}\|\mathbf{E}\|_2\Big]^{1/2}\Big)$$

as needed.

For the probability of failure, note that Corollary 71 holds with probability $1 - \exp(-\Omega(\sqrt{n}))$. To apply this, we needed only to condition on the fact that $\mathbf{S}$ was a subspace embedding for $\mathbf{U}$, which occurs with probability $99/100$ for a single attempt. Running the algorithm $O(n)$ times, by Hoeffding bounds at least one trial will be successful with probability $1 - \exp(-\Omega(\sqrt{n}))$ as desired. To analyze runtime, note that we try at most $\text{poly}(nd)$ guesses of $\mathbf{S}$ and guesses of $\sigma_{\min}$ and $\kappa$. Moreover, there are at most $\nu = (\frac{\kappa}{\epsilon})^{O(k^2)}$ guesses $\mathbf{M}^i$ carried out in Step 2 of Figure 9). For every such guess, we run the optimization program in step $3c$. Since the program has a linear function and a convex unit ball constraint, it is will known that such programs can be solved in polynomial time Boyd and Vandenberghe (2004). Finally, the regression problem in step 4 is linear, and thus can be solved in $\text{poly}(n)$ time, which completes the proof.

■

## Appendix G. A Polynomial Time Algorithm for Exact Weight Recovery with Sparse Noise

In this section, we examine recovery procedures for the weight matrices of a *low-rank* neural network in the presence of arbitrarily large sparse noise. Here, by low rank, we mean that $m > k$. It has frequently been observed in practice that many pre-trained neural-networks exhibit correlation and a low-rank structure Denil et al. (2013); Denton et al. (2014). Thus, in practice it is likely that $k$ need not be as large as $m$ to well-approximate the data.

More formally, we are given $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ where $\mathbf{E}$ is some sparse noise matrix with possibly very large entries. We show that under the assumption that $\mathbf{U}^*$ has orthonormal columns and satisfies an incoherence assumptions (which is fairly standard in the numerical linear algebra community) Candes and Romberg (2007); Candès and Recht (2009); Keshavan et al. (2010); Candès et al. (2011); Jain et al. (2013); Hardt (2014), we can recover the weights $\mathbf{U}^*$, $\mathbf{V}^*$ exactly, even when the sparsity of the matrices is a constant fraction of the number of entries. Our algorithm utilizes results on the recovery of low-rank matrices in the presence of a sparse noise. The error matrix $\mathbf{E} \in \mathbb{R}^{m\times n}$ is a sparse matrix whose non-zero entries are uniformly chosen from the set of all coordinates of an arbitrary matrix $\overline{\mathbf{E}}$. Formally, we define the following noise procedure:

**Definition 79** *(Sparse Noise.) A matrix $\mathbf{E}$ is said to be generated from a $s$-sparse-noise procedure if there is an arbitrary matrix $\overline{\mathbf{E}}$, such that $\mathbf{E}$ is generated by setting all but $s \leq mn$ entries of $\overline{\mathbf{E}}$ to be 0 uniformly at random.*

**Definition 80** *(Incoherence.) A rank $k$ matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ is said to be $\mu$-incoherent if $\texttt{svd}(\mathbf{M}) = \mathbf{P}\Sigma\mathbf{Q}$ is the singular value decomposition of $\mathbf{M}$ and*

$$\begin{aligned} \max_i \|\mathbf{P}^T e_i\|_2^2 &\leq \frac{\mu k}{m} \\ \max_i \|\mathbf{Q} e_i\|_2^2 &\leq \frac{\mu k}{n} \end{aligned} \tag{7}$$

*and*

$$\max_i \|\mathbf{PQ}\|_\infty \leq \sqrt{\frac{\mu k}{nm}} \tag{8}$$

**Remark 81** *The values $\|\mathbf{P}e_i\|_2^2$ and $\|\mathbf{Q}e_i\|_2^2$ are known as the (left and right, respectively)* leverage-scores *of $\mathbf{M}$. For an excellent survey on leverage scores, we refer the reader to Mahoney et al. (2011). We note that the set of leverage scores of $\mathbf{M}$ does not depend on the choice of orthonormal basis $\mathbf{P}$ or $\mathbf{Q}$ Woodruff et al. (2014). Thus, to obtain the bounds given in Equation 7, it suffices let $\mathbf{P}$ be any matrix with orthonormal columns which spans the columns of $\mathbf{M}$, and similarly it suffices to let $\mathbf{Q}$ be any matrix with orthonormal rows which spans the rows of $\mathbf{M}$.*

**Lemma 82** *The entire matrix $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, where $\mathbf{X}$ is i.i.d. Gaussian, $(\mathbf{U}^*)^T, \mathbf{V}^*$ have orthonormal rows, and $\mathbf{U}^*$ is $\mu$-incoherent, meaning $\max_i \|(\mathbf{U}^*)^T e_i\|_2^2 \leq \frac{\mu k}{m}$, is $\overline{\mu}$-incoherent for*

$$\overline{\mu} = O\big((\kappa(\mathbf{V}^*))^2 \sqrt{k \log(n) \mu} + \mu + (\kappa(\mathbf{V}^*))^4 \log(n)\big)$$

**Proof** For $t \in \{\max, \min\}$, let $\sigma_t = \sigma_t(\mathbf{U}^* f(\mathbf{V}^*\mathbf{X}))$. Let $\mathbf{P}\Sigma\mathbf{Q}$ be the SVD of $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$, and let For any $i$, since $\mathbf{U}^*$ and $\mathbf{V}^*$ are orthonormal we have

$$\begin{aligned} \|\mathbf{Q}^T e_i\|_2^2 \leq \frac{\|\Sigma\mathbf{Q}^T e_i\|_2^2}{\sigma_{min}^2} &= \frac{\|\mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) e_i\|_2^2}{\sigma_{min}^2} \\ &= \frac{\|f(\mathbf{V}^*\mathbf{X}) e_i\|_2^2}{\sigma_{min}^2} \\ &\leq \frac{\|\mathbf{V}^*\mathbf{X} e_i\|_2^2}{\sigma_{min}^2} \end{aligned}$$

Now each entry in of $\mathbf{V}^*\mathbf{X}$ is an i.i.d. Gaussian, and so is at most $10\sqrt{\log(n)}$ with probability $1 - e^{-10n}$, so $\|\mathbf{V}^*\mathbf{X}e_i\|_2^2 \leq 100k\log(n)$ with probability $1 - e^{-9n}$ by a union bound. Since the columns of $\mathbf{U}^*$ are orthonormal, $\sigma_{\min}^2 = \sigma_{\min}^2(f(\mathbf{V}^*\mathbf{X}))$, which is at least $\frac{n}{(\kappa(\mathbf{V}^*))^4}$ by Lemma 58. Thus we have that $\|\mathbf{Q}^T e_i\|_2^2 = O(k(\kappa(\mathbf{V}^*))^4 \log(n)/n)$. This shows the $O((\kappa(\mathbf{V}^*))^4 \log(n))$-incoherehnce for the second part of Equation 7, and the first part follows from the $\mu$-incoherence assumption on $U$. The incoherence bound of $(\kappa(\mathbf{V}^*))^2\sqrt{k\log(n)\mu}$ for Equation 8 follows by applying Cauchy Schwartz to the LHS and using the bounds just obtained for Equation 7. ■

**Theorem 83** *(Extending Theorem 1.1 in Candès et al. (2011).) If $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X}) + \mathbf{E}$ where $\mathbf{E}$ is produced by the sparsity procedure outlined above with $s \leq \gamma nm$ for a fixed constant $\gamma > 0$. Then if $\mathbf{U}^*$ has orthonormal columns, is $\mu$-incoherent, $\mathbf{X}$ is Gaussian, and the sample complexity satisfies $n = poly(d, m, k, \kappa(\mathbf{V}^*))$, then there is a polynomial time algorithm which, given only $\mathbf{A}$, outputs both matrices $\mathbf{M} = \mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ and $\mathbf{E}$, given that $k \leq \frac{m}{\overline{\mu}\log^2(n)}$, where $\overline{\mu} = O\big((\kappa(\mathbf{V}^*))^2\sqrt{k\log(n)\mu} + \mu + (\kappa(\mathbf{V}^*))^4\log(n)\big)$.*

**Proof** The results of Candès et al. (2011) demonstrate that solving

$$\begin{array}{ll} \min_{\mathbf{Y}} & \|\mathbf{A}-\mathbf{Y}\|_F^2 \\ \text{s.t.} & \text{rank}(\mathbf{Y}) \leq k \end{array}$$

recovers the optimal low-rank matrix given that conditions of the previous lemma are satisfied. That is, if we do not care about running time, the above optimization problem recovers $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ exactly. However, the above problem is highly non-convex and instead we optimize over the nuclear norm.

$$\begin{array}{ll} \min_{\mathbf{Y},\mathbf{E}} & \|\mathbf{Y}\|_* + \|\mathbf{E}\|_1 \\ \text{s.t.} & \mathbf{Y}+\mathbf{E}=\mathbf{A} \end{array}$$

By Theorem 1.1 in Candès et al. (2011), we know that the solution to the above problem is unique and equal to $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$. It remains to show that the above optimization problem can be solved in polynomial time. Note, the objective function is convex. As mentioned in Lee et al. (2015), we can then run an interior point algorithm and it is well known that in order to achieve additive error $\epsilon$, we need to iterate poly$(\log(1/\epsilon))$ times. Observe, for exact recovery we require a dual certificate that can verify optimality. Section 2.3 in Candès et al. (2011) uses a modified analysis of the golfing scheme introduced by Gross (2011) to create a dual certificate for the aforementioned convex program. We observe that this construction of the dual is independent of the kind of factorization we desire and only requires $\mathbf{Y}$ to be rank $k$. Given that $\mathbf{U}^*, \mathbf{V}^*, \mathbf{X}, \mathbf{E}$ have polynomially bounded bit complexity, this immediately implies a polynomial time algorithm to recover $\mathbf{U}^* f(\mathbf{V}^*\mathbf{X})$ in unfactored form. ■

As an immediate corollary of the above theorem, our exact algorithms of Section C can be applied to the matrix $\mathbf{M}$ of Theorem 83 to recover $\mathbf{U}^*, \mathbf{V}^*$. Formally,

**Corollary 84** *Let $\mathbf{U}^* \in \mathbb{R}^{m\times k}, \mathbf{V}^* \in \mathbb{R}^{k\times d}$ be rank $k$ matrices, where $\mathbf{U}^*$ has orthonormal columns, $\max_i \|(\mathbf{U}^*)^T e_i\|_2^2 \leq \frac{\mu k}{m}$ for some $\mu$, and $k \leq \frac{m}{\overline{\mu}\log^2(n)}$, where $\overline{\mu} = O\big((\kappa(\mathbf{V}^*))^2\sqrt{k\log(n)}\mu + \mu + (\kappa(\mathbf{V}^*))^4\log(n)\big)$. Here $\kappa(\mathbf{V}^*)$ is the condition number of $\mathbf{V}^*$. Let $\mathbf{E}$ be generated from the $s$-sparsity procedure with $s=\gamma nm$ for some constant $\gamma > 0$ and let $\mathbf{A} = \mathbf{U}^* f(\mathbf{V}\mathbf{X}) + \mathbf{E}$. Suppose the sample complexity satisfies $n = poly(d, m, k, \kappa(\mathbf{V}^*))$ Then on i.i.d. Gaussian input $\mathbf{X}$ there is a poly$(n)$ time algorithm that recovers $\mathbf{U}^*, \mathbf{V}^*$ exactly up to a permutation and positive scaling with high probability.*

### G.1. Generalizing to nonlinear activation functions $f$

As remarked in the introduction, for the setting where $\mathbf{X}$ has Gaussian marginals, with several slight changes made to the proofs of our main exact and noisy algorithms, we can generalize our results to any activation function $f$ of the form:

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ \phi(x) & \text{otherwise} \end{cases}$$

where $\phi(x) : [0,\infty] \to [0,\infty]$ is a smooth (on $(0,\infty)$), injective function. We now describe the modifications to our proofs required to generalize to such a function. Note that the primary property of the ReLU used is it's sparsity patterns, i.e. $f(\mathbf{V}\mathbf{X})$ is zero where-ever $\mathbf{V}\mathbf{X}$ is negative. This is the

only property (along with the invertibility of $f(x)$ for $x$ positive) which allows for exact recovery of $\mathbf{U}^*, \mathbf{V}^*$ in Section C. Thus the Lemmas which guarantee the uniqueness of the sign patterns of the rows of $f(\mathbf{V}^*\mathbf{X})$ in their rowspan, namely Lemmas 31 and 33, hold without any modification. Note that continuity of $f$ is required for Lemma 33.

Given this, the linear systems in Algorithm 4, and similarly in the more general exact Algorithm 6, are unchanged. Note that these systems require the injectivity, and thus invertibility, of $f$ for $x > 0$. For instance, in step 8 of Algorithm 4, instead of solving the linear system $(\mathbf{W}_{i,*})_{S_i} = \mathbf{V}_{i,*}\overline{\mathbf{X}}_{S_i}$ for $\mathbf{V}$, we would need to solve $f^{-1}((\mathbf{W}_{i,*})_{S_i}) = \mathbf{V}_{i,*}\overline{\mathbf{X}}_{S_i}$, where $f^{-1}$ is the inverse of $f$ for $x > 0$. Moreover, note that as long as $\phi(x)$ admits a polynomial approximation (with a polynomial of degree at most $\text{poly}(n)$ in the range $[0, \text{poly}(n)]$, the bounds given by the Tensor decomposition algorithms in Section C will still hold.

One detail to note is that if $\phi$ is not multiplicative (i.e. $\phi(xy) = \phi(x)\phi(y)$), then $f$ will no longer commute with positive scalings, so we can no longer pull positive diagonal matrices out of $f(\cdot)$. Observe that $\phi(x) = x^c$ for any $c > 0$ is multiplicative, and this is a non-issue for such $\phi$. On the other hand, note that this does not effect the fact that we recover the true sign pattern needed. Thus in the exact algorithms of Sections C and E, once we multiply by our guessed left inverse of $\mathbf{U}^*$ and we obtain $\mathbf{D}f(\mathbf{V}^*\mathbf{X}) + \mathbf{Z}$ where $\mathbf{Z}$ is some small, negligible error matrix, As long as the entries of $\mathbf{D}$ are not too extreme (that is, exponentially small, see Lipschitz discussion below, as for any of the example functions that follow this will be the case), we will still recover the true sign pattern of $f(\mathbf{V}^*\mathbf{X})$ after rounding. Thus we can always recover $\mathbf{V}^*$ from this sign pattern without using any properties of $f$. Note that our noisy algorithm of Section D does not need to run any such linear system to find $\mathbf{V}^*$, and so this is a non-issue here.

Finally, perhaps the important modification which must be made involves a Lipschitz property of $\phi$. Namely, we frequently use the fact that for the ReLU $f$, if $\|\mathbf{V}\mathbf{X} - \mathbf{V}^*\mathbf{X}\|_F^2 < \epsilon$, then $\|f(\mathbf{V}\mathbf{X}) - f(\mathbf{V}^*\mathbf{X})\|_F^2 < \epsilon$. Here $\mathbf{V}, \mathbf{V}^*$ have unit norm rows, $\mathbf{X} \in \mathbb{R}^{d\times\ell}$ is i.i.d. Gaussian, $\mathbf{V}$ refers to our approximation of $\mathbf{X}$ returned by tensor decomposition, $\epsilon < 1/\text{poly}(\ell)$, and $\ell > \text{poly}(d, m, k, \kappa)$. Note also that we will have $n = \text{poly}(\ell)$, since to our error $\epsilon$ depends on the sample complexity $n$. So once we obtain our estimate $\mathbf{V}$ using $n$ samples, we thereafter restrict our attention to a smaller subset of $\ell$ samples. Now observe that if $\phi(x)$ grows faster than $x$, this bound will no longer hold as is. However, using the fact that the entries of $\mathbf{V}\mathbf{X}$ and $\mathbf{V}^*\mathbf{X}$ are Gaussian, and thus have sub-Gaussian tails, we will be able to bound the blow up. First note that with high probability, we have $\|\mathbf{V}\mathbf{X}\|_\infty + \|\mathbf{V}^*\mathbf{X}\|_\infty < O(\sqrt{\log(\ell)})$. Thus, for a fixed $\phi$, define the $B$-bounded Lipschitz constant $L_B(\phi)$ by

$$L_B(\phi) = \sup_{x\neq y, |x|<B, |y|<B} \frac{|f(x) - f(y)|}{|x - y|}$$

Note if $\phi(x) = x^c$ for some constant $c \in \mathbb{N}$, we have $L_B(\phi) < cB^{c-1}$. Given this, it follows that if $\|\mathbf{V}\mathbf{X} - \mathbf{V}^*\mathbf{X}\|_F^2 < \epsilon$, then $\|f(\mathbf{V}\mathbf{X}) - f(\mathbf{V}^*\mathbf{X})\|_F^2 \leq (L_B(\phi)k\ell)^2\epsilon$ (here we can use bounds between $L_1^2$ and $L_2^2$ of $k\ell$ on these matrices to explicitly relate this difference in Frobenius norm to the Lipschitz constant). Now observe that in all cases where we have $\|\mathbf{V}\mathbf{X} - \mathbf{V}^*\mathbf{X}\|_F^2 < \epsilon$, and need a bound on $\|f(\mathbf{V}\mathbf{X}) - f(\mathbf{V}^*\mathbf{X})\|_F^2$, we can handle a $\text{poly}(\ell)$ blow-up, as $\epsilon < 1/\text{poly}(\ell)$ can be made arbitrarily large by increasing the sample complexity on which the algorithm which originally recovered $\mathbf{V}$ was run on. Thus, we claim that we can handle any function $\phi(x)$ such that $L_{\Theta(\sqrt{\log(\ell)})}(\phi) < \text{poly}(\ell)$. Note that this includes all polynomials $\phi(x) = x^c$ of constant degree, as well as even *exponential* functions $\phi(x) = e^x - 1$ or $\phi(x) = e^{x^2} - 1$.

However, importantly, in addition to the $L_{\Theta(\sqrt{\log(\ell)})}(\phi)$ blow-up in runtime, for the specific case of our noisy algorithm of Theorem 62, our our runtime also blows up by a $\phi(\kappa)^2$ factor. This is because the projection bounds in Corollary 59 will become

$$\|f(\mathbf{V}^*_{i,*}\mathbf{X})\mathbf{P}_{S_{i,-}}\|_2 = \|f(\mathbf{V}^*_{i,*}\mathbf{X})\|_2\Big(1-\Omega(\frac{1}{\phi(\kappa(\mathbf{V}^*))^2\text{poly}(k)})\Big)$$

instead of

$$\|f(\mathbf{V}^*_{i,*}\mathbf{X})\mathbf{P}_{S_{i,-}}\|_2 = \|f(\mathbf{V}^*_{i,*}\mathbf{X})\|_2\Big(1-\Omega(\frac{1}{\kappa(\mathbf{V}^*)^2\text{poly}(k)})\Big)$$

Thus we must make $\ell, 1/\epsilon >> \phi(\kappa(\mathbf{V}^*))^2$ in order to recover the correct signs in our projection based algorithm (Algorithm 7).

To summarize , the primary change that occurs is blow-up the runtime by the bounded Lipschitz constant $L_{\Theta(\sqrt{\log(n)})}(\phi)$ of $\phi$ in the runtime of our exact recovery algorithms for the noiseless case, which is polynomial as long as $\phi(x) = O(e^{x^2})$. This also holds for the case of our fixed parameter tractable noiseless algorithm of Section E, and the fixed parameter tractable noisy algorithm of Section F. For the noisy case of Theorem 62, which is our polynomial time algorithm for sub-Gaussian noise, we also get a blowup of $\phi(\kappa(\mathbf{V}^*))^2$ in the runtime, which is still polynomial as long as $\phi(x)$ is bounded by some constant degree polynomial.