Neural Network Approximation of Continuous Functions in High Dimensions with Applications to Inverse Problems

Santhosh Karnik, Rongrong Wang, and Mark Iwen *

October 11, 2023

Abstract

The remarkable successes of neural networks in a huge variety of inverse problems have fueled their adoption in disciplines ranging from medical imaging to seismic analysis over the past decade. However, the high dimensionality of such inverse problems has simultaneously left current theory, which predicts that networks should scale exponentially in the dimension of the problem, unable to explain why the seemingly small networks used in these settings work as well as they do in practice. To reduce this gap between theory and practice, we provide a general method for bounding the complexity required for a neural network to approximate a Hölder (or uniformly) continuous function defined on a high-dimensional set with a low-complexity structure. The approach is based on the observation that the existence of a Johnson-Lindenstrauss embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of a given high-dimensional set $\mathcal{S} \subset \mathbb{R}^D$ into a low dimensional cube $[-M, M]^d$ implies that for any Hölder (or uniformly) continuous function $f : \mathcal{S} \to \mathbb{R}^p$, there exists a Hölder (or uniformly) continuous function $g : [-M, M]^d \to \mathbb{R}^p$ such that $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. Hence, if one has a neural network which approximates $g : [-M, M]^d \to \mathbb{R}^p$, then a layer can be added that implements the JL embedding $\boldsymbol{A}$ to obtain a neural network that approximates $f : \mathcal{S} \to \mathbb{R}^p$. By pairing JL embedding results along with results on approximation of Hölder (or uniformly) continuous functions by neural networks, one then obtains results which bound the complexity required for a neural network to approximate Hölder (or uniformly) continuous functions on high dimensional sets. The end result is a general theoretical framework which can then be used to better explain the observed empirical successes of smaller networks in a wider variety of inverse problems than current theory allows.

1 Introduction

At present various network architectures (NN, CNN, ResNet) achieve state-of-the-art performance for a broad range of inverse problems including matrix completion [1–4], image-deconvolution [5–7], low-dose CT-reconstitution [8], and electric and magnetic inverse Problems [9] (seismic analysis, electromagnetic scattering). However, since these problems are very high dimensional, classical universal approximation theory for such networks provides very pessimistic estimates of the network sizes required to learn such inverse maps (i.e., as being much larger than what standard computers can store, much less train). As a result, a gap still exists between the widely observed successes of networks in practice and the network size bounds provided by current theory in many inverse problem applications. The purpose of this paper is to provide a refined bound on the size of networks in a wide range of such applications and to show that the network size is indeed affordable in many inverse problem settings. In particular, the bound developed herein depends on the model complexity of the domain of the forward map instead of the domain's extrinsic input dimension, and therefore is much smaller in a wide variety of model settings.

To be more specific, recall in most inverse problems one aims to recover some signal $\boldsymbol{x}$ from its measurements $\boldsymbol{y} = F(\boldsymbol{x})$. Here $\boldsymbol{y}$ and $\boldsymbol{x}$ could both be high dimensional vectors, or even matrices and tensors, and F, which is called the forward map/operator, could either be linear or nonlinear with various regularity conditions depending on the application. In all cases, however, recovering $\boldsymbol{x}$ from $\boldsymbol{y}$ amounts to inverting

*Santhosh Karnik, Rongrong Wang, and Mark Iwen are with the Department of Computational Mathematics, Science, and Engineering at Michigan State University. Rongrong Wang and Mark Iwen are also with the Department of Mathematics at Michigan State University (e-mail: karniksa@msu.edu, wangron6@msu.edu, iwenmark@msu.edu).

F. In other words, one aims to find the operator F^{-1}, that sends every measurement $\boldsymbol{y}$ back to the original signal $\boldsymbol{x}$. Depending on the specific application of interest, there are various commonly considered forms of the forward map F. For example, F could be a linear map from high to low dimensions as in compressive sensing applications; F could be a convolution operator that computes the shifted local blurring of an image in an image deblurring setting; F could be a mask that filters out the unobserved entries of the data as in the matrix completion application; or F could also be the source-to-solution map of a differential equation as in ODE/PDE based inverse problems.

In most of these applications, the inverse operator F^{-1} does not possess a closed-form expression. As a result, in order to approximate the inverse one commonly uses analytical approaches that involve solving, e.g., an optimization problem. Take sparse recovery as an example. With prior knowledge that the true signal $\boldsymbol{x} \in \mathbb{R}^n$ is sparse, it is known that one can recover it from under-determined measurements, $\mathbb{R}^m \ni \boldsymbol{y} = \boldsymbol{\Phi x}$ with $m < n$, by solving the optimization problem

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{z}} \|\boldsymbol{z}\|_0, \quad \boldsymbol{\Phi z} = \boldsymbol{y}.$$

As a result, one can see that linear measurement map F defined by $F(\boldsymbol{x}) = \boldsymbol{\Phi x} = \boldsymbol{y}$, with its domain restricted to the low-complexity domain of s-sparse vectors $\Sigma_s \subset \mathbb{R}^n$, has an inverse, $F^{-1} : \boldsymbol{\Phi}(\Sigma_s) \to \Sigma_s$. And, the minimizer $\widehat{\boldsymbol{x}}$ above satisfies $\widehat{\boldsymbol{x}} = F^{-1}(\boldsymbol{y}) = F^{-1}(\boldsymbol{\Phi x}) = \boldsymbol{y}$ for all $\boldsymbol{x} \in \Sigma_s$.

Note that traditional optimization-based approaches could be extremely slow for large-scale problems (e.g., for n large above). Alternatively, we can approximate the inverse operator by a neural network. Amortizing the initial cost of an expensive training stage, the network can later achieve unprecedented speed over time at the test stage leading to better total efficiency over its lifetime. To realize this goal, however, we need to first find a neural network architecture f_θ, and train it to approximate F^{-1}, so that the approximation error $\max_{\boldsymbol{y}} \|f_\theta(\boldsymbol{y}) - F^{-1}(\boldsymbol{y})\| = \|f_\theta(\boldsymbol{y}) - \boldsymbol{x}\|$ is small. The purpose of this paper is to provide a unified way to give a meaningful estimation of the size of the network that one can use in situations where the domain of F is low-complexity, as is the case in, e.g., compressive sensing, low-rank matrix completion, deblurring with low-dimensional signal assumptions, etc..

2 Main Results

We begin by stating a few definitions. We say that a neural network ϵ-*approximates* a function f if the function implemented by the neural network $\widehat{f}$ satisfies $\|\widehat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\|_\infty \leq \epsilon$ for all $\boldsymbol{x}$ in the domain of f. We say that a neural network architecture ϵ-approximates a function class $\mathcal{F}$ if for any function $f \in \mathcal{F}$, there exists a choice of edge weights and node bias parameters such that the function $\widehat{f}$ implemented by the neural network with that choice of edge weights and node bias parameters satisfies $\|\widehat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\|_\infty \leq \epsilon$ for all $\boldsymbol{x}$ in the domain of f. For brevity, we refer to a feedforward neural network with at most $\mathcal{N}$ nodes, $\mathcal{E}$ edges[1], and $\mathcal{L}$ layers as a $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN. We also refer to a feedforward neural network architecture with at most $\mathcal{N}$ nodes, $\mathcal{E}$ edges, and $\mathcal{L}$ layers as a $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN architecture. Similarly, we will refer to a convolutional neural network with at most $\mathcal{N}$ nodes, $\mathcal{P}$ parameters, and $\mathcal{L}$ layers as a $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN. And, we will also refer to a convolutional neural network architecture with at most $\mathcal{N}$ nodes, $\mathcal{P}$ parameters, and $\mathcal{L}$ layers as a $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN architecture.

We say that a function $\Delta : [0, \infty) \to [0, \infty)$ is a modulus of continuity if it is non-decreasing and satisfies $\lim_{r \to 0^+} \Delta(r) = \Delta(0) = 0$. We say that a function f between Euclidean spaces admits a modulus of continuity Δ if Δ is a modulus of continuity, and if $\|f(\boldsymbol{x}) - f(\boldsymbol{x}')\|_2 \leq \Delta(\|\boldsymbol{x} - \boldsymbol{x}'\|_2)$ holds for all $\boldsymbol{x}, \boldsymbol{x}'$ in the domain of f. For brevity, we will refer to such a function as a $\mathcal{C}(\Delta(r))$-function.

For any constants $L > 0$ and $\alpha \in (0, 1]$, we say that a function f between Euclidean spaces is (L, α)-Hölder if f admits $\Delta(r) = Lr^\alpha$ as a modulus of continuity. We also say that a function f is α-Hölder if f is (L, α)-Hölder for some constant $L > 0$.

Finally, for any positive integers $d < D$, any set $\mathcal{S} \subset \mathbb{R}^D$, and any constant $\rho \in (0, 1)$, we say that a matrix $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ is a ρ-JL (Johnson-Lindenstrauss) embedding of $\mathcal{S}$ into $\mathbb{R}^d$ if

$$(1 - \rho)\|\boldsymbol{x} - \boldsymbol{x}'\|_2 \leq \|\boldsymbol{Ax} - \boldsymbol{Ax}'\|_2 \leq (1 + \rho)\|\boldsymbol{x} - \boldsymbol{x}'\|_2 \quad \text{holds for all} \quad \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}.$$

[1]Throughout this paper, we use the term "edges" to denote the number of connections between nodes in a feedforward neural network. Many papers refer to these as "weight parameters" or just "weights".

If we furthermore have $\boldsymbol{A}(\mathcal{S}) := \{\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x} \in \mathcal{S}\} \subset \mathcal{T}$, we say that $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ into $\mathcal{T}$. Intuitively, a ρ-JL embedding of $\mathcal{S}$ into $\mathbb{R}^d$ maps $\mathcal{S}$ from a high-dimensional space to a low-dimensional space without significantly distorting the Euclidean distances between points.

Contributions and Related Work: Existing universal approximation theorems for various types of neural networks are mainly stated for functions defined on an d-dimensional cube. Our main contribution is to generalize these results to functions that admit a specified modulus of continuity (e.g., Hölder continuous functions) that are defined on arbitrary JL-embeddable subsets of very high dimensional Euclidean space. We then demonstrate how our results can be applied to various example inverse problems in order to obtain reasonable estimates of the network sizes needed in each case we consider.

More explicitly, we show that if there exists a ρ-JL embedding of a high-dimensional set $\mathcal{S} \subset \mathbb{R}^D$ into a low-dimensional cube $[-M, M]^d$, then we can use any neural network architecture which can ϵ-approximate all $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder functions on $[-M, M]^d$ to construct a new neural network architecture which can ϵ-approximate all (L, α)-Hölder functions defined on $\mathcal{S}$. To establish this, we show that if there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S} \subset \mathbb{R}^D$ into d-dimensions, then for any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$, there exists a $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$ (where $M = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{x}\|_\infty$) such that $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. Hence, if we have a neural network which can approximate $g : [-M, M]^d \to \mathbb{R}^p$, then we can compose it with a neural network which implements the JL embedding $\boldsymbol{A}$ to obtain a neural network which approximates $f : \mathcal{S} \to \mathbb{R}^p$. By pairing JL embedding existence results along with results on the approximation of Hölder functions by neural networks, we obtain results which bound the complexity required for a neural network to approximate Hölder functions on low-complexity high-dimensional sets. We can also generalize the above argument to extend results for functions with a specified modulus of continuity to higher dimensions in an analogous fashion.

The expressive power of neural networks is important in applications as a means of both guiding network architecture design choices, as well as for providing confidence that good network solutions exist in general situations. As a result, numerous results about neural network approximation power have been established in recent years (see, e.g., [10–14]). Most results concern the approximation of functions on all of $\mathbb{R}^D$, however, and yield network sizes that increase exponentially with the input dimension D. As a result, the high dimensionality of many inverse problems leads to bounds from most of the existing literature which are too large to explain the observed empirical success of neural network approaches in such applications.

A similar high-dimensional scaling issue arises in many image classification tasks as well. Motivated by this setting [15] refined previous approximation results for ReLU networks, and showed that input data that is close to a low-dimensional manifold leads to network sizes that only grow exponentially with respect to the intrinsic dimension of the manifold. However, this improved bound relies on the data fitting a manifold assumption, which is quite strong in the setting of inverse problems. For example, even the "simple" compressive sensing/sparse recovery problem discussed above does not have a domain/range that forms a manifold (note that the intersections of s-dimensional subspaces in Σ_s prevent it from being a manifold). Therefore, to study the expressive power of networks in the context of a class of inverse problems which is at least large enough to include compressive sensing, one needs to remove such strict manifold assumptions. Another mild issue with such manifold results is that the number of neurons also depends on the curvature of the manifold in question which can be difficult to estimate. Furthermore, such curvature dependence is unavoidable for manifold results and needs to be incorporated into any valid bounds.[2]

The idea of using a JL embedding to reduce the dimensionality of the input before feeding it into a neural network was studied in [16]. There, the authors perform experiments on the webspam [17], url [18], KDD2010-a, and KDD2010-b datasets [19], which have input data with millions of dimensions. They show that a deep neural network with an untrained random projection layer that reduces the dimensionality down to 1000 can improve on the state of the art results or achieve competitive performance on these datasets. Furthermore, the dimensionality reduction allows the neural network to be trained efficiently. However, [16] did not provide

[2]To see why, e.g., curvature dependence is unavoidable, consider any discrete training dataset in a compact ball. There always exists a 1-dimensional manifold, namely a curve, that goes through all the data points. Thus, the mere existence of the 1-dimensional manifold does not mean the data complexity is low. Curvature information and other manifold properties matter as well!

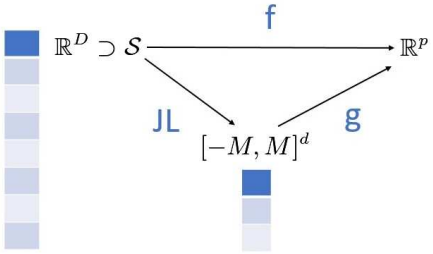

Figure 1: If there exists a ρ-JL embedding of $\mathcal{S} \subset \mathbb{R}^D$ into $[-M, M]^d$, then we can write the target function $f : \mathcal{S} \to \mathbb{R}^p$ as $f = g \circ JL$ where $g : [-M, M]^d \to \mathbb{R}^p$. So, we can then construct a neural network approximation of f by using a neural network approximation of g and adding a layer to implement the JL embedding.

any theoretical guarantees for the performance of a neural network with a linear dimensionality reduction layer.

We note that [20] have recently independently proposed a more general framework where a geometric deep learning model first applies a continuous and injective map ϕ to transform the input space to a manifold, after which a composition of exponential maps and DNNs is used to obtain the output. Applying a JL-transform to $\boldsymbol{x} \in \mathcal{S}$ as done herein is similar, but there are some key differences. In our work, the JL-embedding is implemented by the first few layers of the neural network in order to compress the input data before it goes through the rest of the network. Their more general map ϕ is focused on feature extraction, and it is not implemented by a neural network. So, both ϕ and $\mathcal{S}$ must be known a priori in practice. More specifically, our results only require an upper bound on the intrinsic dimensionality of $\mathcal{S}$ in order to utilize theory guaranteeing the existence of linear JL-embeddngs with few rows. Thus, neither detailed a priori knowledge of $\mathcal{S} \subset \mathbb{R}^D$ nor of the linear map is required herein. As a result, our approach is significantly simpler to implement and apply in practice, and can often be utilized even when S is only partially and/or approximately known.

We now state our main technical theorems which allow us to prove our main results. First, we give a theorem for extending feed-forward neural network approximation results from uniformly continuous functions on a low-dimensional hypercube to a high-dimensional set. See Appendix A for the proof.

Theorem 1. *Let $d < D$ be positive integers, and let $M > 0$, $\rho \in (0, 1)$, and $\epsilon > 0$ be constants. Let $\Delta(r)$ be a modulus of continuity. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ of $\mathcal{S}$ into $[-M, M]^d$.*

(a) Suppose that any $\mathcal{C}(\sqrt{p}\Delta(\frac{r}{1-\rho}))$-function $g : [-M, M]^d \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN. Then, any $\mathcal{C}(\Delta(r))$ function $f : \mathcal{S} \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N} + D, \mathcal{E} + Dd, \mathcal{L} + 1)$-FNN.

(b) Furthermore, if there exists a single $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN architecture that can ϵ-approximate every $\mathcal{C}(\sqrt{p}\Delta(\frac{r}{1-\rho}))$-function $g : [-M, M]^d \to \mathbb{R}^p$, then there also exists another $(\mathcal{N} + D, \mathcal{E} + Dd, \mathcal{L} + 1)$-FNN architecture that can ϵ-approximate every $\mathcal{C}(\Delta(r))$-function $f : \mathcal{S} \to \mathbb{R}^p$.

We also provide a version of this theorem for extending approximation results for CNNs. See Appendix A for the proof.

Theorem 2. *Let $d < D$ be positive integers, and let $M > 0$, $\rho \in (0, 1)$, and $\epsilon > 0$ be constants. Let $\Delta(r)$ be a modulus of continuity. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ of $\mathcal{S}$ into $[-M, M]^d$ which is of the form $\boldsymbol{A} = \boldsymbol{MD}$, where $\boldsymbol{M}$ is a partial circulant matrix, and $\boldsymbol{D}$ is a diagonal matrix with ± 1 on its diagonal.*

(a) Suppose that any $\mathcal{C}(\sqrt{p}\Delta(\frac{r}{1-\rho}))$-function $g : [-M, M]^d \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN. Then, any $\mathcal{C}(\Delta(r))$-function $f : \mathcal{S} \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N} + 4D, \mathcal{P} + 2D, \mathcal{L} + 4)$-CNN.

(b) Furthermore, if there exists a single $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN that can ϵ-approximate every $\mathcal{C}(\sqrt{p}\Delta(\frac{r}{1-\rho}))$-function $g : [-M, M]^d \to \mathbb{R}^p$, then there also exists another $(\mathcal{N} + 4D, \mathcal{P} + 2D, \mathcal{L} + 4)$-CNN that can ϵ-approximate every $\mathcal{C}(\Delta(r))$-function $f : \mathcal{S} \to \mathbb{R}^p$.

We also provide improvements for both of the above theorems which state that for the special case of Hölder functions, whose modulus of continuity is $\Delta(r) = Lr^\alpha$, the $\sqrt{p}$ factor in $\sqrt{p}\Delta(\frac{r}{1-\rho})$ can be removed. See Appendix B for the proofs.

Theorem 3. *Let $d < D$ be positive integers, and let $L > 0$, $M > 0$, $\alpha \in (0, 1]$, $\rho \in (0, 1)$, and $\epsilon > 0$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into $[-M, M]^d$.*

(a) Suppose that any $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN. Then, any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N} + D, \mathcal{E} + Dd, \mathcal{L} + 1)$-FNN.

(b) Furthermore, if there exists a single $(\mathcal{N}, \mathcal{E}, \mathcal{L})$-FNN architecture that can ϵ-approximate every $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$, then there also exists another $(\mathcal{N} + D, \mathcal{E} + Dd, \mathcal{L} + 1)$-FNN architecture that can ϵ-approximate every (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$.

Theorem 4. *Let $d < D$ be positive integers, and let $L > 0$, $M > 0$, $\alpha \in (0, 1]$, $\rho \in (0, 1)$, and $\epsilon > 0$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into $[-M, M]^d$ which is of the form $\boldsymbol{A} = \boldsymbol{MD}$, where $\boldsymbol{M}$ is a partial circulant matrix, and $\boldsymbol{D}$ is a diagonal matrix with ± 1 on its diagonal.*

(a) Suppose that any $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN. Then, any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$ can be ϵ-approximated by a $(\mathcal{N} + 4D, \mathcal{P} + 2D, \mathcal{L} + 4)$-CNN.

(b) Furthermore, if there exists a single $(\mathcal{N}, \mathcal{P}, \mathcal{L})$-CNN architecture that can ϵ-approximate every $\left(\frac{L}{(1-\rho)^\alpha}, \alpha\right)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$, then there also exists another $(\mathcal{N} + 4D, \mathcal{P} + 2D, \mathcal{L} + 4)$-CNN architecture that can ϵ-approximate every (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$.

Remark 1. These theorems ensure that the network size for approximating f grows exponentially with the compressed dimension d instead of growing exponentially with the input dimension D. The task now reduces to making the compressed dimension d as small as possible while still ensuring that a ρ-JL embedding of $\mathcal{S}$ into $[-M, M]^d$ exists.

Remark 2. These theorems are quite general as parts (a) and (b) are not restricted to any particular type of network or activation function. In Section 2.3, we provide three corollaries of these theorems which establish the expressive power of the feedforward and convolutional neural networks.

Remark 3. If an inverse operator is Hölder continuous and there exists a ρ-JL embedding of the set of possible observations $\mathcal{S}$ into d dimensions, then the theorem gives us a bound on the complexity of a neural network architecture required to approximate the inverse operator.

Remark 4. A key ingredient in the above theorems is that for any uniformly (or Hölder) continuous function $f : \mathcal{S} \to \mathbb{R}^p$ and any ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into a hypercube $[-M, M]^d$, there exists a uniformly (or Hölder) continuous function $g : [-M, M]^d \to \mathbb{R}^p$ such that $g(\boldsymbol{Ax}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. Unfortunately, this is not the case if we replace uniform (or Hölder) continuity with differentiability. In Appendix C, we provide an example of a set $\mathcal{S} \subset \mathbb{R}^D$ and a smooth function $f : \mathcal{S} \to \mathbb{R}$ for which there is no ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ (with $d < D$) of $\mathcal{S}$ into a hypercube $[-M, M]^d$ and differentiable function $g : [-M, M]^d \to \mathbb{R}^p$ such that $g(\boldsymbol{Ax}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. As such, we are not able to use our main idea to extend approximation results about differentiable functions.

2.1 JL embeddings, covering numbers and Gaussian width

As the existence of the JL map is a critical assumption of our theorem, in this section, we discuss the sufficient conditions for this assumption to hold. In addition, we also care about the structures of the JL maps, as they will end up being the first layer of the final neural network. For example, if the neural network is of convolution type, we need to make sure that a circulant JL matrix exists.

Existence of ρ-JL maps: It is well-known that for finite sets $\mathcal{S}$, the existence of a ρ-JL embedding can be guaranteed by the Johnson-Lindenstrauss Lemma. For sets $\mathcal{S}$ with infinite cardinality, the

Johnson-Lindenstrauss lemma cannot be directly used. In the following proposition, we extend the Johnson-Lindenstrauss lemma from a finite set of n points to a general set $\mathcal{S}$. See Appendix D for its proof.

Proposition 1. *Let $\rho \in (0,1)$. For $\mathcal{S} \subseteq \mathbb{R}^D$, define*

$$U_{\mathcal{S}} := \overline{\left\{ \frac{\boldsymbol{x}-\boldsymbol{x}'}{\|\boldsymbol{x}-\boldsymbol{x}'\|_2} \ : \ \boldsymbol{x},\boldsymbol{x}' \in \mathcal{S} \text{ s.t. } \boldsymbol{x} \neq \boldsymbol{x}' \right\}}$$

to be the closure of the set of unit secants of $\mathcal{S}$, and $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$ to be the covering number of $U_{\mathcal{S}}$ with δ-balls. Then, there exists a set $\mathcal{S}_1$ with $|\mathcal{S}_1| = 2\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$ points such that if a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ is a ρ-JL embedding of $\mathcal{S}_1$, then $\boldsymbol{A}$ is also a $(\rho + 2\|\boldsymbol{A}\|_2\delta)$-JL embedding of $\mathcal{S}$.

The proposition guarantees that whenever we have a JL-map for finite sets, we can extend it to a JL-map for infinite sets with similar level of complexity measured in terms of the covering numbers. There are many known JL-maps for finite sets that we can extend from, including sub-Gaussian matrix [21], Gaussian circulant matrices with random sign flip [22], etc. We present some of the related results here.

Proposition 2 ([21])**.** *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^D$. Let $\rho \in (0, \frac{1}{2})$ and $\beta \in (0,1)$. Let $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ be a random matrix whose entries are i.i.d. from a subgaussian distribution with mean 0 and variance 1. Then, there exists a constant $C > 0$ depending only on the subgaussian distribution such that if $d \geq C\rho^{-2}\log\frac{n}{\beta}$, then $\frac{1}{\sqrt{d}}\boldsymbol{A}$ will be a ρ-JL embedding of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ with probability at least $1-\beta$.*

Proposition 3 (Corollary 1.3 in [22])**.** *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^D$. Let $\rho \in (0, \frac{1}{2})$, and let $d = O(\rho^{-2}\log^{1+\alpha} n)$ for some $\alpha > 0$. Let $\boldsymbol{A} = \frac{1}{\sqrt{d}}\boldsymbol{M}\boldsymbol{D}$ where $\boldsymbol{M} \in \mathbb{R}^{d\times D}$ is a random Gaussian circulant matrix and $\boldsymbol{D} \in \mathbb{R}^{D\times D}$ is a random Rademacher diagonal matrix. Then, with probability at least $\frac{2}{3}\left(1-(D+d)e^{-\log^{\alpha} n}\right)$, $\boldsymbol{A}$ is a ρ-JL embedding of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.*

Note that the α in the proposition can be set to be any positive number making the probability of failure less than 1. Combining the results of Propositions 2 and 3 with Proposition 1, we have the following existence result for the JL map of an arbitrary set $\mathcal{S}$. See Appendix E for its proof.

Proposition 4. *Let $\rho \in (0, \frac{1}{2})$ be a constant. For $\mathcal{S} \subseteq \mathbb{R}^D$, let $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$ to be the covering number with δ-balls of the unit secant $U_{\mathcal{S}}$ of $\mathcal{S}$ defined in Proposition 1. Then*

(a) If $D \geq d \gtrsim \rho^{-2}\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$, then there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ which is a ρ-JL embedding of $\mathcal{S}$.

(b) If $D \geq d \gtrsim \rho^{-2}\log(4D+4d)\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$, then there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ in the form of $\boldsymbol{M}\boldsymbol{D}$ and of size $d \times D$ that works as ρ-JL map for $\mathcal{S}$, where $\boldsymbol{M}$ is a partial circulant matrix and $\boldsymbol{D}$ is a diagonal matrix with ± 1 on its diagonal.

The above proposition characterizes the compressibility of a set $\mathcal{S}$ by a JL-mapping in terms of the covering number. Alternatively, one can also characterize it using the Gaussian width. For example, in [23] it is shown using methods from [24] that if the set of unit secants of $\mathcal{S}$ has a low Gaussian width, then with high probability a subgaussian random matrix with provide a low-distortion linear embedding, and the dimension d required scales quadratically with the Gaussian width of the set of unit secants of $\mathcal{S}$.

Proposition 5 (Corollary 2.1 in [23])**.** *Let $\rho, \beta \in (0,1)$ be constants. Let $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ be a matrix whose rows $\boldsymbol{a}_1^T, \ldots, \boldsymbol{a}_d^T$ are independent, isotropic ($\mathbb{E}[\boldsymbol{a}_i\boldsymbol{a}_i^T] = \mathbf{I}$), and subgaussian random vectors. Let $\mathcal{S} \subset \mathbb{R}^D$, and let*

$$\omega(U_{\mathcal{S}}) := \mathbb{E}\sup_{\boldsymbol{u}\in U_{\mathcal{S}}} \langle \boldsymbol{u}, \boldsymbol{z}\rangle, \quad \boldsymbol{z} \sim \mathit{Normal}(0, \mathbf{I})$$

be the Gaussian width of $U_{\mathcal{S}}$. Then, there exists a constant $C > 0$ depending only on the distribution of the rows of $\boldsymbol{A}$ such that if

$$d \geq \frac{C}{\rho^2}\left(\omega(U_{\mathcal{S}}) + \sqrt{\log\tfrac{2}{\beta}}\right)^2,$$

then $\frac{1}{\sqrt{d}}\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ with probability at least $1-\beta$.

If $\mathcal{S}$ is known, one can use either the log-covering number (Proposition 2) or the Gaussian width (Proposition 3) to compute a lower bound for d. If one only has samples from $\mathcal{S}$, one may still estimate the covering number. In [25], the authors demonstrate a practical method for estimating the intrinsic dimension of a set by using a greedy algorithm to estimate the log-packing number for several different radii δ, and then extrapolating the linear region of the graph of the log-packing number vs. $\log \delta$ to estimate the packing number at finer radii. Then using the fact that the covering numbers of a set may be bounded by its packing numbers, one may in turn obtain bounds for the log-covering number.

2.2 Universal approximator neural networks for uniformly/Hölder continuous functions on d-dimensional cubes

In Theorems 1 and 2, we showed that with the help of JL, approximation rate of neural networks for functions defined on an arbitrary set $\mathcal{S}$ can be derived from their approximation rates for functions defined on the cube $[-M, M]^d$. In this section, we review known results for the latter, so that they can be used in combination with Theorems 1 and 2 to provide useful approximation results for network applications to various inverse problems. Specifically, we review two types of universal approximation results for functions defined on the cube $[-M, M]^d$. One is for Feedforward ReLU networks and the other is for Resnet type CNNs.

Feedforward ReLU network: The fully connected feedforward neural network with ReLU activation is known to be a universal approximator of any uniformly continuous function on the box $[-M, M]^d$. Moreover, for such networks, the non-asymptotic approximation error has also been established, allowing us to get an estimate of the network size. The proposition below is a variant of Proposition 1 in [13], and the proof uses an approximating function that uses the same ideas as in [13]. See Appendix F for its proof.

Proposition 6. *Let d and p be positive integers, and let $M > 0$ be a constant. Let $\Delta(r)$ be a modulus of continuity. Then, for any positive integer N, there exists a ReLU NN architecture with at most*

$$(p + C_1)(2N + 1)^d \text{ edges}, \ C_2(2N + 1)^d + p \text{ nodes}, \text{ and } \lceil \log_2(d + 1) \rceil + 2 \text{ layers}$$

that can $\Delta(\frac{M\sqrt{d}}{N})$-approximate the class of $\mathcal{C}(\Delta(r))$-functions $g : [-M, M]^d \to \mathbb{R}^p$. Here, $C_1, C_2 > 0$ are universal constants. Also for each edge of the ReLU NN, the corresponding weight is either independent of g, or is of the form $g_i(\boldsymbol{y})$ for some fixed $\boldsymbol{y} \in [-M, M]^d$ and coordinate $i = 1, \ldots, p$.

By setting $\Delta(r) = Lr^\alpha$ for some constants $L > 0$ and $\alpha \in (0, 1]$, and setting $N = \left\lceil \frac{M\sqrt{d}}{(\epsilon/L)^{1/\alpha}} \right\rceil$ for some $\epsilon > 0$, we get the following corollary for approximating Hölder functions.

Corollary 1. *Let d and p be positive integers, and let $L, M, \epsilon > 0$ and $\alpha \in (0, 1]$ be constants. Then, there exists a ReLU NN architecture with at most*

$$(p + C_1)\left(2\left\lceil \frac{M\sqrt{d}}{(\epsilon/L)^{1/\alpha}} \right\rceil + 1\right)^d \text{ edges}, \ C_2\left(2\left\lceil \frac{M\sqrt{d}}{(\epsilon/L)^{1/\alpha}} \right\rceil + 1\right)^d + p \text{ nodes}, \text{ and } \lceil \log_2(d + 1) \rceil + 2 \text{ layers}$$

that can ϵ-approximate the class of (L, α)-Hölder functions $g : [-M, M]^d \to \mathbb{R}^p$. Again, $C_1, C_2 > 0$ are universal constants. Also for each edge of the ReLU NN, the corresponding weight is either independent of g, or is of the form $g_i(\boldsymbol{y})$ for some fixed $\boldsymbol{y} \in [-M, M]^d$ and coordinate $i = 1, \ldots, p$.

Convolutional Neural Network: As many successful network applications on inverse problems result from the use of filters in the CNN architectures [26], we are particularly interested in the expressive power of CNN in approximating the Hölder functions. Currently, known non-asymptotic results for CNNs include [10–12], but they are all established under stricter assumptions about f than mere Lipschitz continuity. On the other hand, the ResNet-based CNN with the following architecture has been shown to possess a good convergence rate for even Hölder continuous functions:

$$CNN_\theta^\sigma := FC_{W,b} \circ (\mathrm{Conv}^\sigma_{\boldsymbol{\omega_M}, \mathbf{b_M}} + \mathrm{id}) \circ \cdots \circ (\mathrm{Conv}^\sigma_{\boldsymbol{\omega_1}, \mathbf{b_1}} + \mathrm{id}) \circ P, \tag{1}$$

where σ is the activation function, each $\text{Conv}_{\omega_\mathbf{m},\mathbf{b_m}}$ is a convolution layer with L_m filters ω_m^1,...,$\omega_m^{L_m}$ stored in $\omega_\mathbf{M}$ and L_m bias $b_m^1, ..., b_m^{L_m}$ stored in $\mathbf{b_m}$. The addition by the identity map, $\text{Conv}^\sigma_{\omega_\mathbf{M},\mathbf{b_M}} + \text{id}$, makes it a residual block. Here $FC_{W,b}$ represents a fully connected layer appended to the final layer of the network, and $P : \mathbb{R}^d \to \mathbb{R}^{d\times C} : x \to (x, 0, \cdots 0)$ is a padding operation that adds zeros to align the number of channels in the first and second layers. One can see that the ResNet-based CNN is essentially a normal CNN with skip connections.

The following asymptotic result is proven in [27].

Proposition 7 (Corollary 4 from [27])**.** *Let $f : [-1, 1]^d \to \mathbb{R}$ be an α-Hölder function. Then, for any $K \in \{2, ..., d\}$, there exists a CNN $f^{(CNN)}$ with $O(N)$ residual blocks, each of which has depth $O(\log N)$ and $O(1)$ channels, and whose filter size is at most K, such that $\|f - f^{(CNN)}\|_\infty \leq \widetilde{O}(N^{-\alpha/d})$, where the $\widetilde{O}$ denotes that $\log N$ factors have been suppressed.*

2.3 Our Main results

We can now pair results guaranteeing the existence of a ρ-JL embedding of $\mathcal{S} \subset \mathbb{R}^D$ into $[-M, M]^d$ with results for approximating functions on $[-M, M]^d$ to obtain results for approximating functions on $\mathcal{S} \subset \mathbb{R}^D$.

By combining Proposition 6 with Propositions 4(a) and 5, we obtain the following result for approximating continuous functions on a high-dimensional set by a feedforward ReLU NN. The proof of this theorem is given in Appendix G.

Theorem 5. *Let $d < D$ be positive integers, and let $\rho \in (0, \frac{1}{2})$ be a constant. Let $\Delta(r)$ be a modulus of continuity. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded set and $U_\mathcal{S}$ be its set of unit secants. Suppose that*

$$d \gtrsim \min\left\{\rho^{-2}\log \mathcal{N}(U_\mathcal{S}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}),\quad \rho^{-2}\left(\omega(U_\mathcal{S})\right)^2\right\},$$

where $\mathcal{N}(U_\mathcal{S}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$ is the covering number and $\omega(U_\mathcal{S})$ is the Gaussian width of $U_\mathcal{S}$. Then, for any positive integer N, there exists a ReLU neural network architecture with at most

$$(p + C_1)(2N + 1)^d + Dd \text{ edges},$$
$$C_2(2N + 1)^d + p + D \text{ nodes},$$
$$\text{and } \lceil \log_2(d + 1)\rceil + 3 \text{ layers}$$

that can $\sqrt{p}\Delta(\frac{M\sqrt{d}}{(1-\rho)N})$-approximate the class of $\mathcal{C}(\Delta(r))$-functions $f : \mathcal{S} \to \mathbb{R}^p$, where $M = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{x}\|_\infty$.

By applying Corollary 1 instead of Proposition 6, we obtain a variant of the previous theorem for approximating Hölder functions on a high-dimensional set by a feedforward ReLU NN. The proof of this theorem is given in Appendix H.

Theorem 6. *Let $d < D$ be positive integers, and let $L > 0$, $\alpha \in (0, 1]$, and $\rho \in (0, \frac{1}{2})$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded set and $U_\mathcal{S}$ be its set of unit secants. Suppose that*

$$d \gtrsim \min\left\{\rho^{-2}\log \mathcal{N}(U_\mathcal{S}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}),\quad \rho^{-2}\left(\omega(U_\mathcal{S})\right)^2\right\},$$

where $\mathcal{N}(U_\mathcal{S}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$ is the covering number and $\omega(U_\mathcal{S})$ is the Gaussian width of $U_\mathcal{S}$. Then, there exists a ReLU neural network architecture with at most

$$(p + C_1)\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d + Dd \text{ edges},$$
$$C_2\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d + p + D \text{ nodes},$$
$$\text{and } \lceil \log_2(d + 1)\rceil + 3 \text{ layers}$$

that can ϵ-approximate the class of (L, α)-Hölder functions $f : \mathcal{S} \to \mathbb{R}^p$, where $M = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{x}\|_\infty$.

By combining Proposition 7 with Proposition 4(b), we obtain the following result for approximating Hölder functions on a high-dimensional set by a ResNet type CNN. The proof of this theorem is given in Appendix I.

Theorem 7. *Let $d < D$ be positive integers, and let $\alpha \in (0, 1]$ and $\rho \in (0, \frac{1}{2})$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded set and $U_{\mathcal{S}}$ be its set of unit secants. Suppose that*

$$d \gtrsim \rho^{-2} \log(4D + 4d) \log \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}).$$

Then, for any α-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$, there exists a ResNet type CNN $f^{(CNN)}$ in the form of (1) with $O(N)$ residual blocks, each of which has a depth $O(\log N)$ and $O(1)$ channels, and whose filter size is at most K such that $\|f - f^{(CNN)}\|_\infty \leq \widetilde{O}(N^{-\alpha/d})$, where the $\widetilde{O}$ denotes that $\log N$ factors have been suppressed.

3 Applications to Inverse Problems

Now we focus on inverse problems and demonstrate how the main theorems can be used to provide a reasonable estimate of the size of the neural networks needed to solve some classical inverse problems in signal processing. The problems we consider here are sparse recovery, blind deconvolution, and matrix completion. In all these inverse problems, we want to recover some signal $\boldsymbol{x} \in \mathcal{S}$ from its forward measurement $\boldsymbol{y} = F(\boldsymbol{x})$, where the forward map F is assumed to be known. The minimal assumption we have to impose on F is that it has an inverse with a known modulus of continuity. As we shall see, however, in the following examples we can in fact assume Lipschitz continuity of the inverse.

Assumption 1 (invertibility of the forward map): Let $\mathcal{S}$ be the domain of the forward map F, and $\mathcal{Y} = F(\mathcal{S})$ be the range. Assume that the inverse operator $F^{-1} : \mathcal{Y} \to \mathcal{S}$ exists and is Lipschitz continuous with constant L, so that

$$\|F^{-1}(\boldsymbol{y}_1) - F^{-1}(\boldsymbol{y}_2)\|_2 \leq L\|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2, \quad \text{for all} \quad \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{Y}.$$

For any inverse problems satisfying Assumption 1, Theorems 6 and 7 provide ways to estimate the size of the universal approximator networks for the inverse map. When applying the theorems to each problem, we need to first estimate the covering number of $U_{\mathcal{Y}}$. Depending on the problem, one may estimate the covering number either numerically or theoretically. If the domain $\mathcal{Y}$ of the inverse map is irregular and discrete, then it may be easier to compute the covering number numerically. If the domain has a nice mathematical structure, then we may be able to estimate it theoretically. In the three examples below we will use theoretical estimation. From them, we see that it is quite common for inverse problems to have a small intrinsic complexity. As a result, Theorems 6 and 7 can significantly reduce the required size of the network below the sizes provided by previously known results.

We begin by emphasizing that the covering number that the theorems use is for the unit secants of $\mathcal{Y}$, which can be appreciably larger than the covering number of $\mathcal{Y}$ itself.

Sparse recovery: Sparsity is now one of the most commonly used priors in inverse problems as signals in many real applications possess a certain level of sparsity with respect to a given basis. For simplicity, we consider strictly sparse signals that have a small number of nonzero entries. Let Σ_s^N be the set of $s-$sparse vectors of length N. Assume a sparse vector is measured linearly $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} \equiv F(\boldsymbol{x})$, where $\boldsymbol{\Phi} \in \mathbb{R}^{m\times N}$. The inverse problem amounts to recovering $\boldsymbol{x}$ from $\boldsymbol{y}$. Now that we want to use a network to approximate the inverse map $F^{-1} : \boldsymbol{\Phi}(\Sigma_s^N) \to \Sigma_s^N$, and estimate the size of the network through the theorems, we need to estimate the covering number of the unit secant $U_{\boldsymbol{\Phi}(\Sigma_s^N)}$.

Proposition 8. *Suppose $\boldsymbol{\Phi}$ satisfies the restricted isometry property so that the inverse map $F^{-1} : \boldsymbol{\Phi}(\Sigma_s^N) \to \Sigma_s^N$ is Lipschitz continuous with Lipschitz constant L. Then, for any $\rho \in (0, \frac{1}{2})$, there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times m}$ of the set of possible observations*

$$\mathcal{Y} = \{\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} : \boldsymbol{x} \in \Sigma_s^N\},$$

into $\mathbb{R}^d$*, provided that*

$$d \gtrsim \rho^{-2} s \log \frac{N\sqrt{m}}{s\rho}.$$

For such choice of d*, we have that for any bounded subset* $\mathcal{S} \subset \mathcal{Y}$*, there exists a ReLU neural network architecture with at most*

$$(N + C_1)\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + md \text{ edges},$$

$$C_2\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + m + N \text{ nodes},$$

$$\text{and } \lceil \log_2(d+1)\rceil + 3 \text{ layers}$$

which can ϵ*-approximate the inverse map* F^{-1} *over the set* $\mathcal{S}$*. Here* $M = \max_{\boldsymbol{y}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{y}\|_2$*.*

Blind deconvolution: Blind deconvolution concerns the recovery of a signal $\boldsymbol{x}$ from its blurry measurements

$$\boldsymbol{y} = \boldsymbol{k} \otimes \boldsymbol{x} \tag{2}$$

when the kernel $\boldsymbol{k}$ is also unknown. Here $\otimes$ denotes the convolution operator. Note that blind-deconvolution is an ill-posed problem due to the existence of a scaling ambiguity between $\boldsymbol{x}$ and $\boldsymbol{k}$, namely, if $(\boldsymbol{k}, \boldsymbol{x})$ is a solution, then $(\alpha\boldsymbol{k}, \frac{1}{\alpha}\boldsymbol{x})$ with $\alpha \neq 0$ is also a solution. To resolve this issue, we focus on recovering the outer product $\boldsymbol{x}\boldsymbol{k}^T$, where $\boldsymbol{x}$ and $\boldsymbol{k}$ here are both column vectors.

The recovery of the outer product $\boldsymbol{x}\boldsymbol{k}^T$ from the convolution $\boldsymbol{y} = \boldsymbol{k} \otimes \boldsymbol{x}$ can be well-posed in various settings [28,29]. For example, [29] showed that if we assume $\boldsymbol{x} = \boldsymbol{\Phi}\boldsymbol{u}$ and $\boldsymbol{k} = \boldsymbol{\Psi}\boldsymbol{v}$, where $\boldsymbol{\Phi} \in \mathbb{R}^{N\times n}(n < N)$ is i.i.d. Gaussian matrix and $\boldsymbol{\Psi} \in \mathbb{R}^{N\times m}(m < N)$ is a matrix of small coherence, then for large enough N, the outer-product $\boldsymbol{x}\boldsymbol{k}^T$ can be stably recovered from $\boldsymbol{y}$ in the following sense. For any two signal-kernel pairs $(\boldsymbol{x}, \boldsymbol{k})$, $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{k}})$ and their corresponding convolutions $\boldsymbol{y}$, $\tilde{\boldsymbol{y}}$, we have

$$\left\|\boldsymbol{x}\boldsymbol{k}^T - \tilde{\boldsymbol{x}}\tilde{\boldsymbol{k}}^T\right\| \leq L\|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|_2 \tag{3}$$

for some L. To determine the size of a neural network to approximate the inverse map by $F^{-1} : \boldsymbol{y} \to \boldsymbol{x}\boldsymbol{k}^T$, we need to estimate the covering number of the unit secant cone of $\mathcal{Y} = \{\boldsymbol{y} = \boldsymbol{x} \otimes \boldsymbol{k}, \boldsymbol{x} \in \boldsymbol{\Phi}\Sigma_s^N, \boldsymbol{k} \in \text{span}\boldsymbol{\Psi}\}$, which is done in the following proposition. The proof of this proposition is given in Appendix K.

Proposition 9. *Suppose the inverse map* $F^{-1} : \boldsymbol{y} \to \boldsymbol{x}\boldsymbol{k}^T$ *is Lipschitz continuous with Lipschitz constant* L*. Then, for any* $\rho \in (0, \frac{1}{2})$*, there exists a* ρ*-JL embedding* $\boldsymbol{A} \in \mathbb{R}^{d\times N}$ *of the set of possible observations*

$$\mathcal{Y} = \{\boldsymbol{y} = \boldsymbol{x} \otimes \boldsymbol{k}, \boldsymbol{x} \in \textit{span}(\boldsymbol{\Phi}), \boldsymbol{k} \in \textit{span}(\boldsymbol{\Psi})\},$$

into $\mathbb{R}^d$*, provided that*

$$d \gtrsim \rho^{-2} \max\{m, n\} \log \frac{L\sqrt{N}}{\rho}.$$

For such choice of d*, we have that for any bounded subset* $\mathcal{S} \subset \mathcal{Y}$*, there exists a ReLU neural network architecture with at most*

$$(N^2 + C_1)\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + Nd \text{ edges},$$

$$C_2\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + N^2 + N \text{ nodes},$$

$$\text{and } \lceil \log_2(d+1)\rceil + 3 \text{ layers}$$

which can ϵ*-approximate the inverse map* F^{-1} *over the set* $\mathcal{S}$*. Here* $M = \max_{\boldsymbol{y}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{y}\|_2$

Matrix completion: Matrix Completion is a central task in machine learning where we want to recover a matrix from its partially observed entries. It arises from a number of applications including image super resolution [30,31], image/video denoising [32], recommender systems [1,2], and gene-expression prediction [33], etc.. Recently neural network models have achieved state-of-the-art performance [1–4], but a general existence result in the non-asymptotic regime is still missing.

In this setting, the measurements $\boldsymbol{Y} = P_\Omega(\boldsymbol{X})$ consists of a set of observed entries of the unknown low-rank matrix $\boldsymbol{X}$, where Ω is the index set of the observed entries and P_Ω is the mask that sets all but entries in Ω to 0. Let $M_r^{n,m}$ be the set of $n \times m$ matrices with rank at most r, and $\boldsymbol{X} \in M_r^{n,m}$. If the mask is random, and the left and right eigenvectors $\boldsymbol{U}, \boldsymbol{V}$ of $\boldsymbol{X}$ are incoherent in the sense that

$$\max_{1\le i\le n} \left\|\boldsymbol{U}^T \boldsymbol{e}_i\right\|_2 \le \sqrt{\frac{\mu_0 r}{n}}, \quad \max_{1\le i\le m} \left\|\boldsymbol{V}^T \boldsymbol{e}_i\right\|_2 \le \sqrt{\frac{\mu_0 r}{m}}, \text{ and} \tag{4}$$
$$\max_{1\le i\le n, 1\le j\le m} \left|(\boldsymbol{U}\boldsymbol{V}^T)_{i,j}\right| \le \sqrt{\frac{\mu_1 r}{nm}}$$

all hold, then it is known (see, e.g., [34]) that the inverse map $F^{-1} : \boldsymbol{Y} \to \boldsymbol{X}$ exists and is Lipschitz continuous with overwhelming probability provided that the number of observations

$$|\Omega| \gtrsim \mu_0 r \max\{m, n\} \log^2 \max\{m, n\}.$$

Let us denote the set of low-rank matrices satisfying (4) by $\mathcal{C}$. To estimate the complexity of the inverse map, we wiil now compute the covering numbers of $U_\mathcal{Y}$ for $\mathcal{Y} = \{\boldsymbol{Y} = P_\Omega(\boldsymbol{X}) : \boldsymbol{X} \in M_r^{m,n} \cap \mathcal{C}\}$. This is done using the following proposition, which is proven in Appendix L.

Proposition 10. *Suppose the mask is chosen so that the inverse map $F^{-1} : \boldsymbol{Y} = P_\Omega(\boldsymbol{X}) \to \boldsymbol{X}$ is Lipschitz continuous with Lipschitz constant L. Then, for any $\rho \in (0, \frac{1}{2})$, there exists a ρ-JL embedding $A : \mathbb{R}^{m\times n} \to \mathbb{R}^d$ of the set of possible observations*

$$\mathcal{Y} = \{P_\Omega(\boldsymbol{X}) : \boldsymbol{X} \in M_r^{m,n} \cap \mathcal{C}\},$$

into $\mathbb{R}^d$ provided that

$$d \gtrsim \rho^{-2} r(m+n) \log \frac{L\sqrt{mn}}{\rho}.$$

For such choice of d, we have that for any bounded subset $\mathcal{S} \subset \mathcal{Y}$, there exists a ReLU neural network architecture with at most

$$(mn + C_1)\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + mnd \text{ edges},$$

$$C_2\left(2\left\lceil \frac{LM\sqrt{d}}{(1-\rho)\epsilon}\right\rceil + 1\right)^d + 2mn \text{ nodes},$$

$$\text{and } \lceil \log_2(d+1)\rceil + 3 \text{ layers}$$

which can ϵ-approximate the inverse map F^{-1} over the set $\mathcal{S}$. Here $M = \max_{\boldsymbol{Y}\in\mathcal{S}} \|vec(A(\boldsymbol{Y}))\|_\infty$

Remark 5. In each of Propositions 8, 9, and 10, it is shown that the number of nodes and edges in a neural network to solve a matrix completion problem scale (with respect to ϵ) exponentially with the intrinsic dimension of the problem d and not the much larger ambient dimension.

Remark 6. One can also get bounds on the size of a ResNet CNN needed for each of these inverse problems by enlarging d by a factor of the logarithm of the ambient dimension (which ensures a circulant ρ-JL embedding exists), and then applying Theorem 7 instead of Theorem 6.

4 Conclusions, Limitations, and Discussion

The main message of this paper is that when neural networks are used to approximate Hölder continuous functions, the size of the network only needs to grow exponentially with respect to the intrinsic complexity of the input set measured using either its Gaussian width or its covering numbers. Therefore, it is often more optimistic than previous estimates that require the size of the network to grow exponentially with respect to the extrinsic input dimension.

We note that when the domain of the input is a manifold, our techniques would yield results that are slightly worse than optimal. Specifically, [23,35] both show that a k-dimensional manifold has a ρ-JL embedding into $d = \widetilde{O}(k/\rho^2)$ dimensions. As such, our results for the size of a neural network that approximates α-Hölder functions on a manifold would scale with respect to ϵ like $O(\epsilon^{-d/\alpha})$ where $d = \widetilde{O}(k/\rho^2)$ as compared to the $O(\epsilon^{-k/\alpha} \log \frac{1}{\epsilon})$ scaling in [15]. This is of course to be expected as the neural network in [15] is constructed carefully with respect to the structure of the manifold, while our construction is more oblivious to the domain of the function. In addition, as explained in Section C, our result only holds for Hölder indices $0 < \alpha \leq 1$ as opposed to all $\alpha > 0$ in [15]. However, the use of the JL embedding allows our result to be stated under a more general (non-manifold) assumption of the input set and for a much broader class of neural networks – although we only stated it for feedforward neural networks and the ResNet type of convolutional neural networks, the same idea naturally applies to other types of networks as long as an associated JL-map exists.

The estimates the results herein provide for the network size ultimately depend on the complexity of the input set, measured by either the covering numbers, or by the Gaussian width, of its set of unit secants. The computation of these quantities varies case by case, and in some cases might be rather difficult. This is a possible limitation of the proposed method. In particular, if the estimation of the input set complexity is not tight enough, the results herein may again become overly pessimistic. Having said that, for many classical inverse problems, the covering numbers and the Gaussian width estimates are not too difficult to calculate. As we demonstrated in Section 4, there are many known properties that one can use to facilitate the calculation. And, when a training dataset is given, one can even approximate covering numbers numerically with off-the-shelf algorithms.

Finally, although the applications of neural networks to inverse problems are seeing a lot of current success, there are also failed attempts that don't work for unknown reasons. One common explanation is that the size of the network in use is not large enough for the targeted application. Since inverse problems models usually have a much higher intrinsic dimensionality than, say, image classification models, the required network sizes might indeed be much larger. Classical universal approximation theorems only guarantee small errors when the network size approaches infinity, therefore are not very helpful in the non-asymptotic regime where one has to choose the network size. And, such parameter choices are now known to be critical to good performance. We hope the presented results provide more insight in this regard.

Acknowledgments

Rongrong Wang was supported in part by NSF CCF-2212065. Mark Iwen was supported in part by NSF DMS 2106472. We would also like to note that we became aware of similar independent work by Demetrio Labate and Ji Shi [36] during a poster session at Texas A&M as part of the Inaugural CAMDA Conference (May 22 - 25, 2023) after the initial submission of this paper. We regard this independent and parallel development of similar results and proof strategies to be an indication of their timeliness, utility, and general interest.

References

[1] Yin Zheng, Bangsheng Tang, Wenkui Ding, and Hanning Zhou. A neural autoregressive approach to collaborative filtering. In *International Conference on Machine Learning*, pages 764–773. PMLR, 2016.

[2] Federico Monti, Michael Bronstein, and Xavier Bresson. Geometric matrix completion with recurrent multi-graph neural networks. *Advances in neural information processing systems*, 30, 2017.

[3] Gintare Karolina Dziugaite and Daniel M Roy. Neural network matrix factorization. *arXiv preprint arXiv:1511.06443*, 2015.

[4] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[5] Li Xu, Jimmy S Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. *Advances in neural information processing systems*, 27, 2014.

[6] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018.

[7] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017.

[8] Hu Chen, Yi Zhang, Weihua Zhang, Peixi Liao, Ke Li, Jiliu Zhou, and Ge Wang. Low-dose ct via convolutional neural network. *Biomedical optics express*, 8(2):679–694, 2017.

[9] Enzo Coccorese, Raffaele Martone, and F Carlo Morabito. A neural network approach for the solution of electric and magnetic inverse problems. *IEEE transactions on magnetics*, 30(5):2829–2839, 1994.

[10] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and computational harmonic analysis*, 48(2):787–794, 2020.

[11] Philipp Petersen and Felix Voigtlaender. Equivalence of approximation by convolutional neural networks and fully-connected networks. *Proceedings of the American Mathematical Society*, 148(4):1567–1581, 2020.

[12] Dmitry Yarotsky. Universal approximations of invariant maps by neural networks. *Constructive Approximation*, 55(1):407–474, 2022.

[13] Dmitry Yarotsky. Optimal approximation of continuous functions by very deep relu networks. In *Conference on learning theory*, pages 639–649. PMLR, 2018.

[14] Hongzhou Lin and Stefanie Jegelka. Resnet with one-neuron hidden layers is a universal approximator. *Advances in neural information processing systems*, 31, 2018.

[15] Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Efficient approximation of deep relu networks for functions on low dimensional manifolds. *Advances in neural information processing systems*, 32, 2019.

[16] Piotr Iwo Wójcik and Marcin Kurdziel. Training neural networks on high-dimensional data using random projection. *Pattern Analysis and Applications*, 22:1221–1231, 2019.

[17] Steve Webb, James Caverlee, and Calton Pu. Introducing the webb spam corpus: Using email spam to identify web spam automatically. In *CEAS*, 2006.

[18] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 681–688, 2009.

[19] Hsiang-Fu Yu, Hung-Yi Lo, Hsun-Ping Hsieh, Jing-Kai Lou, Todd G McKenzie, Jung-Wei Chou, Po-Han Chung, Chia-Hua Ho, Chun-Fu Chang, Yin-Hsuan Wei, et al. Feature engineering and classifier ensemble for kdd cup 2010. In *KDD cup*, 2010.

[20] Anastasis Kratsios and Leonie Papon. Universal approximation theorems for differentiable geometric deep learning. *Journal of Machine Learning Research*, 23(196):1–73, 2022.

[21] Jiří Matoušek. On variants of the johnson–lindenstrauss lemma. *Random Structures & Algorithms*, 33(2):142–156, 2008.

[22] Lizhi Cheng and Hui Zhang. New bounds for circulant johnson-lindenstrauss embeddings. *Communications in Mathematical Sciences*, 12(4):695–705, 2014.

[23] M.A. Iwen, Benjamin Schmidt, and Arman Tavakoli. On fast johnson-lindenstrauss embeddings of compact submanifolds of $\mathbb{R}^n$ with boundary. *Discrete & Computational Geometry*, 2022.

[24] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

[25] Balázs Kégl. Intrinsic dimension estimation using packing numbers. *Advances in neural information processing systems*, 15, 2002.

[26] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[27] Kenta Oono and Taiji Suzuki. Approximation and non-parametric estimation of resnet-type convolutional neural networks. In *International Conference on Machine Learning*, pages 4922–4931. PMLR, 2019.

[28] Kiryung Lee, Yanjun Li, Marius Junge, and Yoram Bresler. Stability in blind deconvolution of sparse signals and reconstruction by alternating minimization. In *2015 International Conference on Sampling Theory and Applications (SampTA)*, pages 158–162. IEEE, 2015.

[29] Ali Ahmed, Benjamin Recht, and Justin Romberg. Blind deconvolution using convex programming. *IEEE Transactions on Information Theory*, 60(3):1711–1732, 2013.

[30] Feng Shi, Jian Cheng, Li Wang, Pew-Thian Yap, and Dinggang Shen. Low-rank total variation for image super-resolution. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 155–162. Springer, 2013.

[31] Feilong Cao, Miaomiao Cai, and Yuanpeng Tan. Image interpolation via low-rank matrix completion and recovery. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(8):1261–1270, 2014.

[32] Hui Ji, Chaoqiang Liu, Zuowei Shen, and Yuhong Xu. Robust video denoising using low rank matrix completion. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1791–1798. IEEE, 2010.

[33] Arnav Kapur, Kshitij Marwah, and Gil Alterovitz. Gene expression prediction using low-rank matrix completion. *BMC bioinformatics*, 17(1):1–13, 2016.

[34] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[35] Richard G Baraniuk and Michael B Wakin. Random projections of smooth manifolds. *Foundations of computational mathematics*, 9(1):51–77, 2009.

[36] Demetrio Labate and Ji Shi. Low dimensional approximation and generalization of multivariate functions on smooth manifolds using deep neural networks, 2023. Preprint available at http://doi.org/10.2139/ssrn.4545106.

[37] George J Minty. On the extension of lipschitz, lipschitz-hölder continuous, and monotone functions. *Bulletin of the American Mathematical Society*, 76(2):334–339, 1970.

[38] Jacob T Schwartz. *Nonlinear functional analysis*, volume 4. CRC Press, 1969.

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

[40] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

[41] Emmanuel J Candes and Yaniv Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57(4):2342–2359, 2011.

A Proof of Theorems 1 and 2

First, we prove the following lemma.

Lemma 1. *Let $d < D$ and p be positive integers, and let $M > 0$ and $\rho \in (0,1)$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into $[-M,M]^d$. Let $\Delta(r)$ be a modulus of continuity. Then, for any function $f : \mathcal{S} \to \mathbb{R}^p$ that admits $\Delta(r)$ as a modulus of continuity, there exists a function $g : [-M,M]^d \to \mathbb{R}^p$ that admits $\sqrt{p}\Delta(\frac{r}{1-\rho})$ as a modulus of continuity such that $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$.*

Proof. For any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}$, if $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x}'$ then since $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$, we have that $\|\boldsymbol{x} - \boldsymbol{x}'\|_2 \leq \frac{1}{1-\rho}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\boldsymbol{x}'\|_2 = \frac{1}{1-\rho}\|\boldsymbol{0}\|_2 = 0$, and so, $\|\boldsymbol{x} - \boldsymbol{x}'\|_2 = 0$, i.e., $\boldsymbol{x} = \boldsymbol{x}'$. Therefore, the map $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$ from $\mathcal{S}$ to $\boldsymbol{A}(\mathcal{S}) := \{\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x} \in \mathcal{S}\}$ is invertible. We define $A^{-1} : \boldsymbol{A}(\mathcal{S}) \to \mathcal{S}$ to be the inverse of the map $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$.

Now, for any function $f : \mathcal{S} \to \mathbb{R}^p$ which admits $\Delta(r)$ as a modulus of continuity, we define $\widetilde{g} : \boldsymbol{A}(\mathcal{S}) \to \mathbb{R}^p$ by $\widetilde{g} = f \circ A^{-1}$. Then, for any $\boldsymbol{y}, \boldsymbol{y}' \in \boldsymbol{A}(\mathcal{S})$, we have

$$\begin{aligned}
\|\widetilde{g}(\boldsymbol{y}) - \widetilde{g}(\boldsymbol{y}')\|_2 &= \left\|f(A^{-1}(\boldsymbol{y})) - f(A^{-1}(\boldsymbol{y}'))\right\|_2 && \text{since } g = f \circ A^{-1} \\
&\leq \Delta\left(\left\|A^{-1}(\boldsymbol{y}) - A^{-1}(\boldsymbol{y}')\right\|_2\right) && \text{since } f \text{ admits } \Delta(r) \text{ as a M.O.C.} \\
&\leq \Delta\left(\tfrac{1}{1-\rho}\left\|\boldsymbol{A}A^{-1}(\boldsymbol{y}) - \boldsymbol{A}A^{-1}(\boldsymbol{y}')\right\|_2\right) && \text{since } \boldsymbol{A} \text{ is a } \rho\text{-JL embedding of } \mathcal{S} \\
&= \Delta\left(\tfrac{1}{1-\rho}\left\|\boldsymbol{y} - \boldsymbol{y}'\right\|_2\right). && \text{since } A^{-1} \text{ is the inverse of } \boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}
\end{aligned}$$

Therefore, $\widetilde{g} : \boldsymbol{A}(\mathcal{S}) \to \mathbb{R}^p$ admits $\Delta(\frac{r}{1-\rho})$ as a modulus of continuity. Then, since $\boldsymbol{A}(\mathcal{S}) \subset [-M,M]^d$, we can extend $\widetilde{g}$ to a function $g : [-M,M]^d \to \mathbb{R}^p$ via the definition

$$g_i(\boldsymbol{y}) := \inf_{\boldsymbol{z}\in\boldsymbol{A}(\mathcal{S})}\left[\widetilde{g}_i(\boldsymbol{z}) + \Delta\left(\tfrac{1}{1-\rho}\|\boldsymbol{y} - \boldsymbol{z}\|_2\right)\right] \quad \text{for} \quad i = 1,\ldots,p.$$

Since $\widetilde{g}$ admits a modulus of continuity of $\Delta(\frac{r}{1-\rho})$, so does each coordinate $\widetilde{g}_i$. The above extension formula preserves the modulus of continuity of each coordinate g_i, and so g admits modulus of continuity $\sqrt{p}\Delta(\frac{r}{1-\rho})$. Also, this extension satisfies $g(\boldsymbol{y}) = \widetilde{g}(\boldsymbol{y})$ for all $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{S})$. Finally, for any $\boldsymbol{x} \in \mathcal{S}$, we have $\boldsymbol{A}\boldsymbol{x} \in \boldsymbol{A}(\mathcal{S})$, and so, $g(\boldsymbol{A}\boldsymbol{x}) = \widetilde{g}(\boldsymbol{A}\boldsymbol{x}) = f(A^{-1}(\boldsymbol{A}\boldsymbol{x})) = f(\boldsymbol{x})$, as required. □

With Lemma 1, we can now prove each of the parts of Theorems 1 and 2. As a reminder, we assume that $\mathcal{S} \subset \mathbb{R}^D$ is a bounded set for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into $[-M,M]^d$.

1a) Let $f : \mathcal{S} \to \mathbb{R}^p$ be a function that admits $\Delta(r)$ as a modulus of continuity. By Lemma 1, there exists a function $g : [-M,M]^d \to \mathbb{R}^p$ that admits $\sqrt{p}\Delta(\frac{r}{1-\rho})$ as a modulus of continuity such that $f(\boldsymbol{x}) = g(\boldsymbol{A}\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. By assumption, g can be ϵ-approximated by a feedforward neural network with at most $\mathcal{N}$ nodes, $\mathcal{E}$ edges, and $\mathcal{L}$ layers. In other words, there exists a function $\widehat{g}$ such that $\|\widehat{g}(\boldsymbol{y}) - g(\boldsymbol{y})\|_\infty \leq \epsilon$ for all $\boldsymbol{y} \in [-M,M]^d$, and $\widehat{g}$ can be implemented by a feedforward neural network with at most $\mathcal{N}$ nodes, $\mathcal{E}$ edges, and $\mathcal{L}$ layers.

Define another function $\widehat{f} = \widehat{g} \circ \boldsymbol{A}$, i.e., $\widehat{f}(\boldsymbol{x}) = \widehat{g}(\boldsymbol{A}\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. Since $\boldsymbol{A}(\mathcal{S}) \subset [-M,M]^d$ by assumption, we have that $\boldsymbol{A}\boldsymbol{x} \in [-M,M]^d$ for all $\boldsymbol{x} \in \mathcal{S}$. Then, $\|\widehat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\|_\infty = \|\widehat{g}(\boldsymbol{A}\boldsymbol{x}) - g(\boldsymbol{A}\boldsymbol{x})\|_\infty \leq \epsilon$ for all $\boldsymbol{x} \in \mathcal{S}$, i.e., $\widehat{f}$ is an ϵ-approximation of f.

Furthermore, we can construct a feedforward neural network to implement $\widehat{f} = \widehat{g} \circ \boldsymbol{A}$ by having a linear layer to implement the map $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$, and then feeding this into the neural network implementation of $\widehat{g}$. The map $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$ can be implemented with D nodes for the input layer, and Dd edges between the input nodes and the first hidden layer. By assumption, $\widehat{g}$ can be implemented by a feedforward neural network with at most $\mathcal{N}$ nodes, $\mathcal{E}$ edges, and $\mathcal{L}$ layers. Hence, $\widehat{f} = \widehat{g} \circ \boldsymbol{A}$ can be implemented by a feedforward neural network with at most $\mathcal{N} + D$ nodes, $\mathcal{E} + Dd$ edges, and $\mathcal{L} + 1$ layers, as desired.

1b) If the same feedforward neural network architecture ϵ-approximates every function $g : [-M,M]^d \mapsto \mathbb{R}^p$ that admits $\sqrt{p}\Delta(\frac{r}{1-\rho})$ as a modulus of continuity, then our construction of a feedforward neural network that implements $\widehat{f} = \widehat{g} \circ \boldsymbol{A}$ has the same architecture for every function $f : \mathcal{S} \to \mathbb{R}^p$ that admits $\Delta(r)$ as a modulus of continuity. Hence, the same bounds on the number of nodes, edges, and layers hold.

2a) In a similar manner as 1a), we form a CNN that can approximate $f = g \circ \boldsymbol{A}$ by first implementing the linear map $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$ with a CNN and feeding this into a CNN that approximates g.

The JL matrix $\boldsymbol{A} = \boldsymbol{M}\boldsymbol{D}$ can be represented by a Resnet-CNN structure as follows. Let $\boldsymbol{x}$ be the input of the network, then $\boldsymbol{D}\boldsymbol{x}$, the random sign flip of the input can be realized by setting the weight/kernel w_1 of the first two layers to be the delta function, and the bias vectors to take large values at the location where $\boldsymbol{D}$ has a 1, and small values where $\boldsymbol{D}$ has a -1. Then with the help of the ReLU activation, we can successfully flip the signs. More explicitly, set $T = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{x}\|_\infty$ so that $\boldsymbol{x} \in [-T, T]^D$ for all $\boldsymbol{x} \in \mathcal{S}$. Let $\boldsymbol{b}_i$ be the bias to be added to the ith coordinate of the input. We design a 2 layer Resnet-CNN, $\ell(\boldsymbol{x})$, as follows

$$\ell(\boldsymbol{x})_i = \mathrm{ReLU}(2\boldsymbol{x}_i + \boldsymbol{b}_i) - \mathrm{ReLU}(\boldsymbol{b}_i) - \boldsymbol{x}_i, \quad i = 1, ..., D.$$

The bias $\boldsymbol{b}_i$ is chosen to realize the sign flip as follows. If $\boldsymbol{D}_{ii}$ contains a 1, then we set $\boldsymbol{b}_i = 2T$, which will make $\ell(\boldsymbol{x})_i = \boldsymbol{x}_i$. If $\boldsymbol{D}_{ii}$ contains a -1, then we set $\boldsymbol{b}_i = -2T$, which will make $\ell(\boldsymbol{x})_i = -\boldsymbol{x}_i$, thus realizing the sign-flip. A similar architecture can also be used to realize the application of $\boldsymbol{M}$ to $\boldsymbol{D}\boldsymbol{x}$, which is a convolution followed by a mask (i.e., setting certain entries of $\boldsymbol{x}$ to 0). Specifically, we let $\boldsymbol{m} \in \mathbb{R}^D$ denote the first row of the partial circulant matrix $\boldsymbol{M}$, and we set $T' = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{m} \otimes \boldsymbol{D}\boldsymbol{x}\|_\infty$ so that $\boldsymbol{m} \otimes \boldsymbol{D}\boldsymbol{x} \in [-T', T']^D$. Then, we let $\boldsymbol{b}' \in \mathbb{R}^D$ be a bias vector whose first d entries are T' and whose last $D - d$ entries are $-T'$. Then, we pass $\boldsymbol{D}\boldsymbol{x} = \ell(\boldsymbol{x})$ through the following Resnet-CNN

$$\ell'(\boldsymbol{D}\boldsymbol{x}) = \mathrm{ReLU}(\boldsymbol{m} \otimes \boldsymbol{D}\boldsymbol{x} + \boldsymbol{b}') - \mathrm{ReLU}(\boldsymbol{b}').$$

This convolves $\boldsymbol{D}\boldsymbol{x}$ with $\boldsymbol{m}$ and then sets the last $D - d$ entries to 0.

This Resnet-CNN that implements $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$ requires $2D$ nodes, D parameters, and 2 layers to apply $\boldsymbol{D}$ to $\boldsymbol{x}$, and an additional $2D$ nodes, D parameters, and 2 layers to apply $\boldsymbol{M}$ to $\boldsymbol{D}\boldsymbol{x}$. By adding this to the $\mathcal{N}$ nodes, $\mathcal{P}$ parameters, and $\mathcal{L}$ layers needed for a CNN to approximate the function $g : [-M, M]^d \to \mathbb{R}^p$ that admits $\sqrt{p}\Delta(\frac{r}{1-\rho})$ as a modulus of continuity, we obtain that the function $f : \mathcal{S} \to \mathbb{R}^p$ that admits $\Delta(r)$ as a modulus of continuity can be approximated by a Resnet-CNN with $\mathcal{N} + 4D$ nodes, $\mathcal{P} + 2D$ parameters, and $\mathcal{L} + 4$ layers.

2b) If the same convolutional neural network architecture ϵ-approximates every function $g : [-M, M]^d \mapsto \mathbb{R}^p$ that admits $\sqrt{p}\Delta(\frac{r}{1-\rho})$ as a modulus of continuity, then our construction of a convolutional neural network that implements $\widehat{f} = \widehat{g} \circ \boldsymbol{A}$ has the same architecture for every function $f : \mathcal{S} \to \mathbb{R}^p$ that admits $\Delta(r)$ as a modulus of continuity. Hence, the same bounds on the number of nodes, parameters, and layers hold.

B Proofs of Theorems 3 and 4

We can strengthen Lemma 1 for the special case where $\Delta(r) = Lr^\alpha$ for some constants $L > 0$ and $\alpha \in (0, 1]$, (f is (L, α)-Hölder) in a way that removes the $\sqrt{p}$ factor. This is done by using a theorem which allows us to extend an $\mathbb{R}^p$-valued Hölder function instead of extending each coordinate separately.

Lemma 2. *Let $d < D$ and p be positive integers, and let $L, M > 0$, $\alpha \in (0, 1]$, and $\rho \in (0, 1)$ be constants. Let $\mathcal{S} \subset \mathbb{R}^D$ be a bounded subset for which there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ into $[-M, M]^d$. Then, for any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$, there exists an $(\frac{L}{(1-\rho)^\alpha}, \alpha)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$ such that $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$.*

Proof. For any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$, we can use the same argument as in the proof of Lemma 1 to show that the function $\widetilde{g} : \boldsymbol{A}(\mathcal{S}) \to \mathbb{R}^p$ defined by $\widetilde{g} = f \circ A^{-1}$ satisfies

$$\begin{aligned}
\|\widetilde{g}(\boldsymbol{y}) - \widetilde{g}(\boldsymbol{y}')\|_2 &= \left\|f(A^{-1}(\boldsymbol{y})) - f(A^{-1}(\boldsymbol{y}'))\right\|_2 && \text{since } g = f \circ A^{-1} \\
&\le L \left\|A^{-1}(\boldsymbol{y}) - A^{-1}(\boldsymbol{y}')\right\|_2^\alpha && \text{since } f \text{ is } (L, \alpha)\text{-Hölder} \\
&\le \frac{L}{(1-\rho)^\alpha} \left\|\boldsymbol{A}A^{-1}(\boldsymbol{y}) - \boldsymbol{A}A^{-1}(\boldsymbol{y}')\right\|_2^\alpha && \text{since } \boldsymbol{A} \text{ is a } \rho\text{-JL embedding of } \mathcal{S} \\
&= \frac{L}{(1-\rho)^\alpha} \|\boldsymbol{y} - \boldsymbol{y}'\|_2^\alpha . && \text{since } A^{-1} \text{ is the inverse of } \boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}
\end{aligned}$$

for all $\boldsymbol{y}, \boldsymbol{y}' \in \boldsymbol{A}(\mathcal{S})$. Therefore, $\widetilde{g} : \boldsymbol{A}(\mathcal{S}) \to \mathbb{R}^p$ is $(\frac{L}{(1-\rho)^\alpha}, \alpha)$-Hölder. Then, since $\boldsymbol{A}(\mathcal{S}) \subset [-M, M]^d$, by Theorem 1(ii) in [37] (which is a generalization of the Kirszbraun theorem [38]), there exists a $(\frac{L}{(1-\rho)^\alpha}, \alpha)$-Hölder extension of $\widetilde{g}$ to $[-M, M]^d$, i.e., a function $g : [-M, M]^d \to \mathbb{R}^p$ which is $(\frac{L}{(1-\rho)^\alpha}, \alpha)$-Hölder on $[-M, M]^d$ and satisfies $g(\boldsymbol{y}) = \widetilde{g}(\boldsymbol{y})$ for all $\boldsymbol{y} \in \mathcal{S}$. Finally, for any $\boldsymbol{x} \in \mathcal{S}$, we have $\boldsymbol{A}\boldsymbol{x} \in \boldsymbol{A}(\mathcal{S})$, and so, $g(\boldsymbol{A}\boldsymbol{x}) = \widetilde{g}(\boldsymbol{A}\boldsymbol{x}) = f(A^{-1}(\boldsymbol{A}\boldsymbol{x})) = f(\boldsymbol{x})$, as required. □

The proofs of each of the parts of Theorems 3 and 4 are identical to the proofs of the corresponding parts of Theorems 1 and 2 respectively, except they use Lemma 2 instead of Lemma 1.

C Theorems 1-4 cannot be generalized to differentiable functions

Fix positive integers $D \geq 3$ and $d < D$. Define a set $\mathcal{S} \subset \mathbb{R}^D$ by $\mathcal{S} = \{\boldsymbol{x} \in [-2, 2]^D : \|\boldsymbol{x}\|_0 \leq 2\}$, define the function $f : \mathcal{S} \to \mathbb{R}$ given by $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_2^2$, which is smooth. Now, suppose there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ of $\mathcal{S}$ into some hypercube $[-M, M]^d$ (with $d < D$) and a differentiable function $g : [-M, M]^d \to \mathbb{R}$ which satisfies $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$.

Then, for any differentiable path $\phi(t) \in \mathcal{S}$, we must have

$$g(\boldsymbol{A}\phi(t)) = f(\phi(t)) = \|\phi(t)\|_2^2$$

and thus,

$$\nabla g(\boldsymbol{A}\phi(t))^T \boldsymbol{A}\phi'(t) = 2\phi(t)^T \phi'(t).$$

Let $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_D \in \mathbb{R}^D$ be the Euclidean basis vectors. For any indices $i, j \in \{1, \ldots, D\}$, we can apply the above result for the differentiable path $\phi(t) = \boldsymbol{e}_i + t\boldsymbol{e}_j$ and evaluate at $t = 0$ to obtain

$$\nabla g(\boldsymbol{A}\phi(0))^T \boldsymbol{A}\phi'(0) = 2\phi(0)^T \phi'(0)$$

$$\nabla g(\boldsymbol{A}\boldsymbol{e}_i)^T \boldsymbol{A}\boldsymbol{e}_j = 2\boldsymbol{e}_i^T \boldsymbol{e}_j$$

Since this holds for all indices $i, j \in \{1, \ldots, D\}$, we have that

$$\nabla g(\boldsymbol{A}\boldsymbol{e}_i)^T \boldsymbol{A} = 2\boldsymbol{e}_i^T,$$

for all indices $i \in \{1, \ldots, D\}$, which implies that $2\boldsymbol{e}_i$ is in the rowspace of $\boldsymbol{A}$ for all $i \in \{1, \ldots, D\}$. However, this is impossible since $\boldsymbol{A}$ has $d < D$ rows. Hence, there cannot exist a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ of $\mathcal{S}$ into some hypercube $[-M, M]^d$ (with $d < D$) and a differentiable function $g : [-M, M]^d \to \mathbb{R}$ which satisfies $g(\boldsymbol{A}\boldsymbol{x}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$.

D Proof of Proposition 1

Consider a covering of $U_{\mathcal{S}}$ by $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$ balls of radius δ. Each ball must intersect $U_{\mathcal{S}}$ as otherwise we could remove that ball from the covering and obtain a covering of $U_{\mathcal{S}}$ with only $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta) - 1$ balls of radius δ, which contradicts the definition of $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$. Enumerate these balls $i = 1, \ldots, \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$. For each i, pick a point $\boldsymbol{u}_i \in U_{\mathcal{S}}$ which is also in the i-th ball, and then pick points $\boldsymbol{x}_i, \boldsymbol{x}_i' \in \mathcal{S}$ with $\boldsymbol{x}_i \neq \boldsymbol{x}_i'$ such that $\frac{\boldsymbol{x}_i - \boldsymbol{x}_i'}{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2} = \boldsymbol{u}_i$. Then, set $\mathcal{S}_1 = \{\boldsymbol{x}_i\}_i \cup \{\boldsymbol{x}_i'\}_i$ so $|\mathcal{S}_1| \leq 2\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$.

Suppose $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ is a ρ-JL embedding of $\mathcal{S}_1$. Then, by definition of a ρ-JL embedding,

$$(1-\rho)\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2 \leq \|\boldsymbol{A}\boldsymbol{x}_i - \boldsymbol{A}\boldsymbol{x}_i'\|_2 \leq (1+\rho)\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2, \quad \text{for} \quad i = 1, \ldots, \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta)$$

Now, for any two points $\boldsymbol{y}, \boldsymbol{y}' \in \mathcal{S}$ with $\boldsymbol{y} \neq \boldsymbol{y}'$, there exists an index i such that $\frac{\boldsymbol{y} - \boldsymbol{y}'}{\|\boldsymbol{y} - \boldsymbol{y}'\|_2} \in U_{\mathcal{S}}$ lies in the i-th ball of our covering of $U_{\mathcal{S}}$. Since $\frac{\boldsymbol{x}_i - \boldsymbol{x}_i'}{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2}$ is also in the i-th ball, we have that

$$\left\| \frac{\boldsymbol{x}_i - \boldsymbol{x}_i'}{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2} - \frac{\boldsymbol{y} - \boldsymbol{y}'}{\|\boldsymbol{y} - \boldsymbol{y}'\|_2} \right\|_2 \leq 2\delta.$$

For simplicity of notation, we set $a = \|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2, b = \|\boldsymbol{y} - \boldsymbol{y}'\|_2$. Then we immediately have

$$\begin{aligned}
\|\boldsymbol{A}(\boldsymbol{y} - \boldsymbol{y}')\|_2 &= \left\| \frac{b}{a}\boldsymbol{A}(\boldsymbol{x}_i - \boldsymbol{x}_i') + \boldsymbol{A}(\boldsymbol{y} - \boldsymbol{y}') - \frac{b}{a}\boldsymbol{A}(\boldsymbol{x}_i - \boldsymbol{x}_i') \right\|_2 \\
&\leq \frac{b}{a}\|\boldsymbol{A}(\boldsymbol{x}_i - \boldsymbol{x}_i')\|_2 + \left\| \boldsymbol{A}(\boldsymbol{y} - \boldsymbol{y}' - \frac{b}{a}(\boldsymbol{x}_i - \boldsymbol{x}_i')) \right\|_2 \\
&\leq \frac{b}{a}(1+\rho)\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2 + \|\boldsymbol{A}\|_2 \left\| \boldsymbol{y} - \boldsymbol{y}' - \frac{b}{a}(\boldsymbol{x}_i - \boldsymbol{x}_i') \right\|_2 \\
&\leq (1+\rho)b + 2\|\boldsymbol{A}\|_2\delta b = (1 + \rho + 2\|\boldsymbol{A}\|_2\delta)\|\boldsymbol{y} - \boldsymbol{y}'\|_2,
\end{aligned}$$

where the second inequality used the previous two formulae. The other side of the bi-Lipschitz formula can be proved similarly. Hence, $\boldsymbol{A}$ is also a $(\rho + 2\|\boldsymbol{A}\|_2\delta)$-JL embedding of $\mathcal{S}$.

E Proof of Proposition 4

a) By Proposition 1, there exists a finite set $\mathcal{S}_1$ with at most $|\mathcal{S}_1| \leq 2\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$ points such that any $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ is also a $(\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}})$-JL embedding of $\mathcal{S}$.

We now show that there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ with $\|\boldsymbol{A}\|_2 \leq \sqrt{3D}$ which is $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ by generating a random $\boldsymbol{A}$ and showing that the probability of $\|\boldsymbol{A}\|_2 \leq \sqrt{3D}$ and $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ both occurring is greater than zero.

Let $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ be a random matrix whose entries are i.i.d. from a subgaussian distribution with mean 0 and variance $\frac{1}{d}$. Since

$$\mathbb{E}\|\boldsymbol{A}\|_F^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\mathbb{E}\boldsymbol{A}_{i,j}^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\frac{1}{d} = D,$$

we have that

$$\mathbb{P}\left\{\|\boldsymbol{A}\|_F^2 \geq 3D\right\} \leq \frac{\mathbb{E}\|\boldsymbol{A}\|_F^2}{3D} = \frac{1}{3}.$$

Furthermore, since

$$d \gtrsim \rho^{-2}\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}) \gtrsim \left(\frac{\rho}{2}\right)^{-2}\log(3|\mathcal{S}_1|),$$

by Proposition 2, $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ with probability at least $1 - \frac{1}{3} = \frac{2}{3}$. Therefore, $\boldsymbol{A}$ is both a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ and satisfies $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_F \leq \sqrt{3D}$ with probability at least $\frac{2}{3} - \frac{1}{3} = \frac{1}{3} > 0$.

Hence, there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ such that $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ and satisfies $\|\boldsymbol{A}\|_2 \leq \sqrt{3D}$. Finally, by Proposition 1, since $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$, it is also a $(\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}})$-JL embedding of $\mathcal{S}$. Since $\|\boldsymbol{A}\|_2 \leq \sqrt{3D}$, we have $\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}} \leq \rho$, and thus, $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$, as desired.

b) Again, by Proposition 1, there exists a finite set $\mathcal{S}_1$ with at most $|\mathcal{S}_1| \leq 2\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \frac{\rho}{4\sqrt{3D}})$ points such that any $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ is also a $(\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}})$-JL embedding of $\mathcal{S}$.

Let $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ be a random matrix of the form $\boldsymbol{M}\boldsymbol{D}$ where $\boldsymbol{D} \in \mathbb{R}^{D\times D}$ is a diagonal matrix whose entries are independent Rademacher random variables, and $\boldsymbol{M} \in \mathbb{R}^{d\times D}$ is a random circulant matrix whose entries are Gaussian random variables with mean 0 and variance $\frac{1}{d}$ and entries in different diagonals are independent. Again, we can show that

$$\mathbb{E}\|\boldsymbol{A}\|_F^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\mathbb{E}\boldsymbol{A}_{i,j}^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\mathbb{E}\boldsymbol{M}_{i,j}^2\boldsymbol{D}_{j,j}^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\mathbb{E}\boldsymbol{M}_{i,j}^2 = \sum_{i=1}^{d}\sum_{j=1}^{D}\frac{1}{d} = D,$$

and so,

$$\mathbb{P}\left\{\|\boldsymbol{A}\|_F^2 \geq 3D\right\} \leq \frac{\mathbb{E}\|\boldsymbol{A}\|_F^2}{3D} = \frac{1}{3}.$$

Now, set $\alpha = \log(\log(4D+4d))/\log(\log|\mathcal{S}_1|)$ so that $\log^\alpha|\mathcal{S}_1| = \log(4D+4d)$. Then, since

$$d \gtrsim \rho^{-2}\log(4D+4d)\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}) \gtrsim \left(\frac{\rho}{2}\right)^{-2}\log^{1+\alpha}|\mathcal{S}_1|,$$

by Proposition 3, $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ with probability at least

$$\frac{2}{3}\left(1-(D+d)e^{-\log^\alpha|\mathcal{S}_1|}\right) = \frac{2}{3}\left(1-(D+d)e^{-\log(4D+4d)}\right) = \frac{2}{3}\left(1-\frac{1}{4}\right) = \frac{1}{2}.$$

Therefore, $\boldsymbol{A}$ is both a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ and satisfies $\|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_F \le \sqrt{3D}$ with probability at least $\frac{1}{2}-\frac{1}{3} = \frac{1}{6} > 0$.

Hence, there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ such that $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$ and satisfies $\|\boldsymbol{A}\|_2 \le \sqrt{3D}$. Again, by Proposition 1, since $\boldsymbol{A}$ is a $\frac{\rho}{2}$-JL embedding of $\mathcal{S}_1$, it is also a $(\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}})$-JL embedding of $\mathcal{S}$. Since $\|\boldsymbol{A}\|_2 \le \sqrt{3D}$, we have $\frac{\rho}{2} + \|\boldsymbol{A}\|_2\frac{\rho}{2\sqrt{3D}} \le \rho$, and thus, $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$, as desired.

F Proof of Proposition 6

We first construct a function $\widehat{g}$ that is an ϵ-approximation of g. To do this, we first define a compactly supported "spike" function $\phi : \mathbb{R}^d \to [0,1]$ by

$$\phi(\boldsymbol{z}) = \max\left\{1 + \min\left\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_d, 0\right\} - \max\left\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_d, 0\right\}, 0\right\}.$$

Then, for any positive integer N, define an approximation $\widehat{g} : [-M, M]^d \to \mathbb{R}^p$ to g by

$$\widehat{g}(\boldsymbol{y}) := \sum_{\boldsymbol{n}\in\{-N,\ldots,N\}^d} g(\tfrac{M\boldsymbol{n}}{N})\phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}).$$

Similarly to what was done in [13], it can be shown that the scaled and shifted spike functions $\{\phi(\frac{N\boldsymbol{y}}{M} - \boldsymbol{n})\}_{\boldsymbol{n}\in\{-N,\ldots,N\}^d}$ form a partition of unity, i.e.

$$\sum_{\boldsymbol{n}\in\{-N,\ldots,N\}^d} \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) = 1 \quad \text{for all} \quad \boldsymbol{y} \in [-M, M]^d.$$

Trivially, $\phi(\boldsymbol{y}) \ge 0$ for all $\boldsymbol{y} \in \mathbb{R}^d$. Also, one can check that $\mathrm{supp}(\phi) \subseteq [-1,1]^d$, and thus, $\phi(\frac{N\boldsymbol{y}}{M} - \boldsymbol{n}) = 0$ for all $\boldsymbol{n}$ such that $\|\frac{N\boldsymbol{y}}{M} - \boldsymbol{n}\|_\infty > 1$. Furthermore, for any $\boldsymbol{n}$ such that $\|\frac{N\boldsymbol{y}}{M} - \boldsymbol{n}\|_\infty \le 1$, we have

$$\left\|g(\boldsymbol{y}) - g(\tfrac{M\boldsymbol{n}}{N})\right\|_2 \le \Delta\left(\|\boldsymbol{y} - \tfrac{M\boldsymbol{n}}{N}\|_2\right) \le \Delta\left(\sqrt{d}\|\boldsymbol{y} - \tfrac{M\boldsymbol{n}}{N}\|_\infty\right) = \Delta\left(\tfrac{M\sqrt{d}}{N}\|\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}\|_\infty\right) \le \Delta\left(\tfrac{M\sqrt{d}}{N}\right).$$

Hence, we can bound the approximation error for any $\boldsymbol{y} \in [-M, M]^d$ as follows:

$$
\begin{aligned}
\|\widehat{g}(\boldsymbol{y}) - g(\boldsymbol{y})\|_2 &= \left\| \sum_{\boldsymbol{n}\in\{-N,\dots,N\}^d} g(\tfrac{M\boldsymbol{n}}{N})\phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) - g(\boldsymbol{y}) \right\|_2 \\
&= \left\| \sum_{\boldsymbol{n}\in\{-N,\dots,N\}^d} \left(g(\tfrac{M\boldsymbol{n}}{N}) - g(\boldsymbol{y})\right) \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) \right\|_2 \\
&\leq \sum_{\boldsymbol{n}\in\{-N,\dots,N\}^d} \left\|g(\tfrac{M\boldsymbol{n}}{N}) - g(\boldsymbol{y})\right\|_2 \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) \\
&= \sum_{\left\|\frac{N\boldsymbol{y}}{M}-\boldsymbol{n}\right\|_\infty \leq 1} \left\|g(\tfrac{M\boldsymbol{n}}{N}) - g(\boldsymbol{y})\right\|_2 \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) \\
&\leq \sum_{\left\|\frac{N\boldsymbol{y}}{M}-\boldsymbol{n}\right\|_\infty \leq 1} \Delta\left(\tfrac{M\sqrt{d}}{N}\right) \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) \\
&\leq \sum_{\boldsymbol{n}\in\{-N,\dots,N\}^d} \Delta\left(\tfrac{M\sqrt{d}}{N}\right) \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) \\
&= \Delta\left(\tfrac{M\sqrt{d}}{N}\right).
\end{aligned}
$$

So $\|\widehat{g}(\boldsymbol{y}) - g(\boldsymbol{y})\|_\infty \leq \|\widehat{g}(\boldsymbol{y}) - g(\boldsymbol{y})\|_2 \leq \Delta\left(\tfrac{M\sqrt{d}}{N}\right)$ for all $\boldsymbol{y} \in [-M, M]^d$, i.e., $\widehat{g}$ is a $\Delta\left(\tfrac{M\sqrt{d}}{N}\right)$-approximation of g.

We now focus on constructing a ReLU NN architecture which can implement the ϵ-approximation $\widehat{g}$ for any function g that admits $\Delta(r)$ as a modulus of continuity. We do this by first constructing a ReLU NN that is independent of g which implements the map $\Phi : \mathbb{R}^d \to \mathbb{R}^{(2N+1)^d}$ defined by $(\Phi(\boldsymbol{y}))_{\boldsymbol{n}} = \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n})$. Then, we add a final layer which outputs the appropriate linear combination of the $\phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n})$'s.

Lemma 3. *For any integers $N, d \geq 1$, the maps $m_d : \mathbb{R}^d \to \mathbb{R}^{(2N+1)^d}$ and $M_d : \mathbb{R}^d \to \mathbb{R}^{(2N+1)^d}$ defined by*

$$
(m_d(\boldsymbol{y}))_{\boldsymbol{n}} := \min\left\{\tfrac{N\boldsymbol{y}_1}{M} - \boldsymbol{n}_1, \dots, \tfrac{N\boldsymbol{y}_d}{M} - \boldsymbol{n}_d, 0\right\} \quad \textit{for} \quad \boldsymbol{n} \in \{-N, \dots, N\}^d
$$

and

$$
(M_d(\boldsymbol{y}))_{\boldsymbol{n}} := \max\left\{\tfrac{N\boldsymbol{y}_1}{M} - \boldsymbol{n}_1, \dots, \tfrac{N\boldsymbol{y}_d}{M} - \boldsymbol{n}_d, 0\right\} \quad \textit{for} \quad \boldsymbol{n} \in \{-N, \dots, N\}^d,
$$

can both be implemented by a ReLU NN with $O((2N+1)^d)$ weights, $O((2N+1)^d)$ nodes, and $\lceil \log_2(d+1) \rceil$ layers.

Proof. First, we note that we can write

$$
\begin{aligned}
(m_d(\boldsymbol{y}))_{\boldsymbol{n}} = \min\Big\{&\min\{\tfrac{N\boldsymbol{y}_1}{M} - \boldsymbol{n}_1, \dots, \tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil}\}, \\
&\min\{\tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil+1}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil+1}, \dots, \tfrac{N\boldsymbol{y}_d}{M} - \boldsymbol{n}_d, 0\}\Big\}
\end{aligned}
$$

and

$$
\begin{aligned}
(M_d(\boldsymbol{y}))_{\boldsymbol{n}} = \max\Big\{&\max\{\tfrac{N\boldsymbol{y}_1}{M} - \boldsymbol{n}_1, \dots, \tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil}\}, \\
&\max\{\tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil+1}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil+1}, \dots, \tfrac{N\boldsymbol{y}_d}{M} - \boldsymbol{n}_d, 0\}\Big\}
\end{aligned}
$$

In [39], it is shown that for any positive integer k, the maps $(\boldsymbol{z}_1, \dots, \boldsymbol{z}_k) \mapsto \min\{\boldsymbol{z}_1, \dots, \boldsymbol{z}_k\}$ and $(\boldsymbol{z}_1, \dots, \boldsymbol{z}_k) \mapsto \max\{\boldsymbol{z}_1, \dots, \boldsymbol{z}_k\}$ can be implemented by a ReLU NN with at most $c_1 k$ edges, $c_2 k$ nodes,

and $\lceil \log_2 k \rceil$ layers, where $c_1, c_2 > 0$ are universal constants. So to construct the map m_d, we first implement the $(2N+1)^{\lceil d/2 \rceil}$ maps

$$(\boldsymbol{y}_1, \dots, \boldsymbol{y}_{\lceil d/2 \rceil}) \mapsto \min\{\tfrac{N\boldsymbol{y}_1}{M} - \boldsymbol{n}_1, \dots, \tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil}\} \tag{5}$$

for $(\boldsymbol{n}_1, \dots, \boldsymbol{n}_{\lceil d/2 \rceil}) \in \{-N, \dots, N\}^{\lceil d/2 \rceil}$. Implementing each of these maps requires $c_1 \left\lceil \frac{d}{2} \right\rceil$ edges, $c_2 \left\lceil \frac{d}{2} \right\rceil$ nodes, and $\left\lceil \log_2 \left\lceil \frac{d}{2} \right\rceil \right\rceil$ layers. Next, we implement the $(2N+1)^{\lfloor d/2 \rfloor}$ maps

$$(\boldsymbol{y}_{\lceil d/2 \rceil + 1}, \dots, \boldsymbol{y}_d) \mapsto \min\{\tfrac{N\boldsymbol{y}_{\lceil d/2 \rceil + 1}}{M} - \boldsymbol{n}_{\lceil d/2 \rceil + 1}, \dots, \tfrac{N\boldsymbol{y}_d}{M} - \boldsymbol{n}_d, 0\} \tag{6}$$

for $(\boldsymbol{n}_{\lceil d/2 \rceil + 1}, \dots, \boldsymbol{n}_d) \in \{-N, \dots, N\}^{\lfloor d/2 \rfloor}$. Implementing each of these maps requires $c_1(\left\lfloor \frac{d}{2} \right\rfloor + 1)$ edges, $c_2(\left\lfloor \frac{d}{2} \right\rfloor + 1)$ nodes, and $\left\lceil \log_2(\left\lfloor \frac{d}{2} \right\rfloor + 1) \right\rceil$ layers. After placing these $(2N+1)^{\lceil d/2 \rceil} + (2N+1)^{\lfloor d/2 \rfloor}$ maps in parallel, we construct one final layer as follows. For each $\boldsymbol{n} = (\boldsymbol{n}_1, \dots, \boldsymbol{n}_d) \in \{-N, \dots, N\}^d$, we combine the output of the $(\boldsymbol{n}_1, \dots, \boldsymbol{n}_{\lceil d/2 \rceil})$-th map of the form in Equation 5 and the output of the $(\boldsymbol{n}_{\lceil d/2 \rceil + 1}, \dots, \boldsymbol{n}_d)$-th map of the form in Equation 6 by using them as inputs to a ReLU NN that implements the map $(a, b) \mapsto \min\{a, b\}$. Each of these requires at most $2c_1$ edges and $2c_2$ nodes.

The total number of edges used to implement m_d is

$$\begin{aligned}
&c_1 \left\lceil \tfrac{d}{2} \right\rceil (2N+1)^{\lceil d/2 \rceil} + c_1(\left\lfloor \tfrac{d}{2} \right\rfloor + 1)(2N+1)^{\lfloor d/2 \rfloor} + 2c_1(2N+1)^d \\
\leq & c_1(\left\lceil \tfrac{d}{2} \right\rceil + \left\lfloor \tfrac{d}{2} \right\rfloor + 1)(2N+1)^{\lceil d/2 \rceil} + 2c_1(2N+1)^d \\
= & c_1(d+1)(2N+1)^{\lceil d/2 \rceil} + 2c_1(2N+1)^d \\
= & c_1 \left((d+1)(2N+1)^{-\lfloor d/2 \rfloor} + 2 \right) (2N+1)^d \\
\leq & c_1 \left((d+1) \cdot 3^{-\lfloor d/2 \rfloor} + 2 \right) (2N+1)^d \\
\leq & 4c_1(2N+1)^d,
\end{aligned}$$

where we have used the fact that $N \geq 1$ by definition, and the easily verifiable inequality $(d+1) \cdot 3^{-\lfloor d/2 \rfloor} \leq 2$ for all positive integers d.

A nearly identical calculation shows that the total number of nodes used to implement m_d is at most $4c_2(2N+1)^d$. Finally, since the $(2N+1)^{\lceil d/2 \rceil}$ maps of the form in Equation 5 and the $(2N+1)^{\lfloor d/2 \rfloor}$ maps of the form in Equation 6 are in parallel, the total number of layers used to implement m_d is

$$\max\left\{ \left\lceil \log_2 \left\lceil \tfrac{d}{2} \right\rceil \right\rceil, \left\lceil \log_2(\left\lfloor \tfrac{d}{2} \right\rfloor + 1) \right\rceil \right\} + 1 = \lceil \log_2(d+1) \rceil .$$

Hence, the map m_d can be implemented by a ReLU NN with at most $C_1(2N+1)^d$ edges, $C_2(2N+1)^d$ nodes, and $\lceil \log_2(d+1) \rceil$ layers, as desired. The proof for M_d is identical, except with min replaced by max. □

Next, we note that

$$(\Phi(\boldsymbol{y}))_{\boldsymbol{n}} = \phi(\tfrac{N\boldsymbol{y}}{M} - \boldsymbol{n}) = \max\{1 + (m_d(\boldsymbol{y}))_{\boldsymbol{n}} - (M_d(\boldsymbol{y}))_{\boldsymbol{n}}, 0\} \quad \text{for all} \quad \boldsymbol{n} \in \{-N, \dots, N\}^d.$$

So to construct a ReLU NN which implements Φ, we first place a ReLU NN that implements m_d in parallel with a ReLU NN that implements M_d. Then, we add an extra layer which has $(2N+1)^d$ nodes, where the $\boldsymbol{n}$-th node of this layer has two edges, one from the $\boldsymbol{n}$-th node of m_d and one from the $\boldsymbol{n}$-th node of M_d. Since m_d and M_d are in parallel and each can each be implemented with ReLU NNs with $O((2N+1)^d)$ edges, $O((2N+1)^d)$ nodes, and $\lceil \log_2(d+1) \rceil$ layers, and the last layer has $2(2N+1)^d$ edges and $(2N+1)^d$ nodes, the ReLU NN which implements Φ has $O((2N+1)^d)$ edges, $O((2N+1)^d)$ nodes, and $\lceil \log_2(d+1) \rceil + 1$ layers.

Finally, we can construct a ReLU NN which implements

$$\widehat{g}(\boldsymbol{x}) := \sum_{\boldsymbol{n} \in \{-N, \dots, N\}^d} g(\tfrac{M\boldsymbol{n}}{N}) \phi(\tfrac{N\boldsymbol{x}}{M} - \boldsymbol{n})$$

by using the ReLU NN which implements Φ, followed by a linear layer which computes the weighted sum for $\widehat{g}$. This last layer has p nodes, and $p(2N+1)^d$ edges. So the ReLU NN that implements $\widehat{g}$ has $(p + C_1)(2N+1)^d$ edges, $C_2(2N+1)^d + p$ nodes, and $\lceil \log_2(d+1) \rceil + 2$ layers, as desired.

G Proof of Theorem 5

By combining Proposition 4a and Proposition 5, we have that there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ with

$$d \gtrsim \min\left\{\rho^{-2}\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}),\quad \rho^{-2}\left(\omega(U_{\mathcal{S}})\right)^2\right\}.$$

Let $M = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{x}\|_\infty$ so that $\boldsymbol{A}(\mathcal{S}) \subset [-M, M]^d$, and so, $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ into $[-M, M]^d$. By Proposition 6, there exists a ReLU NN architecture with at most

$$\mathcal{E} = (p + C_1)(2N + 1)^d \text{ edges},$$
$$\mathcal{N} = C_2(2N + 1)^d + p \text{ nodes},$$
$$\text{and } \mathcal{L} = \lceil \log_2(d + 1)\rceil + 2 \text{ layers},$$

which can $\sqrt{p}\Delta(\frac{M\sqrt{d}}{(1-\rho)N})$-approximate any function $g : [-M, M]^d \to \mathbb{R}^p$ which admits a modulus of continuity $\sqrt{p}\Delta(\frac{r}{1-\rho})$. Finally, by applying Theorem 1b, we have that there exists a a ReLU NN architecture with at most

$$\mathcal{E} + Dd = (p + C_1)(2N + 1)^d + Dd \text{ edges},$$
$$\mathcal{N} + D = C_2(2N + 1)^d + p + D \text{ nodes},$$
$$\text{and } \mathcal{L} + 1 = \lceil \log_2(d + 1)\rceil + 3 \text{ layers},$$

which can $\sqrt{p}\Delta(\frac{M\sqrt{d}}{(1-\rho)N})$-approximate any function $f : \mathcal{S} \to \mathbb{R}^p$ that admits a modulus of continuity of $\Delta(r)$, as desired.

H Proof of Theorem 6

By combining Proposition 4a and Proposition 5, we have that there exists a ρ-JL embedding $\boldsymbol{A} \in \mathbb{R}^{d\times D}$ of $\mathcal{S}$ with

$$d \gtrsim \min\left\{\rho^{-2}\log\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}),\quad \rho^{-2}\left(\omega(U_{\mathcal{S}})\right)^2\right\}.$$

Let $M = \sup_{\boldsymbol{x}\in\mathcal{S}} \|\boldsymbol{A}\boldsymbol{x}\|_\infty$ so that $\boldsymbol{A}(\mathcal{S}) \subset [-M, M]^d$, and so, $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ into $[-M, M]^d$. By Corollary 1, there exists a ReLU NN architecture with at most

$$\mathcal{E} = (p + C_1)\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d \text{ edges},$$

$$\mathcal{N} = C_2\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d + p \text{ nodes},$$

$$\text{and } \mathcal{L} = \lceil \log_2(d + 1)\rceil + 2 \text{ layers},$$

which can ϵ-approximate any $(\frac{L}{(1-\rho)^\alpha}, \alpha)$-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$. Finally, by applying Theorem 3b, we have that there exists a ReLU NN architecture with at most

$$\mathcal{E} + Dd = (p + C_1)\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d + Dd \text{ edges},$$

$$\mathcal{N} + D = C_2\left(2\left\lceil \frac{M\sqrt{d}}{(1-\rho)(\epsilon/L)^{1/\alpha}}\right\rceil + 1\right)^d + p + D \text{ nodes},$$

$$\text{and } \mathcal{L} + 1 = \lceil \log_2(d + 1)\rceil + 3 \text{ layers},$$

which can ϵ-approximate any (L, α)-Hölder function $f : \mathcal{S} \to \mathbb{R}^p$, as desired.

I Proof of Theorem 7

Let $f : \mathcal{S} \to \mathbb{R}^p$ be the α-Hölder target function to approximate. By Proposition 4b, we have that there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d \times D}$ in the form $\boldsymbol{MD}$ where $\boldsymbol{M}$ is a partial circulant matrix and $\boldsymbol{D}$ is a diagonal matrix with ± 1 entries such that $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ with

$$d \gtrsim \rho^{-2} \log(4D + 4d) \log \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3D}}).$$

Let $M = \sup_{\boldsymbol{x} \in \mathcal{S}} \|\boldsymbol{Ax}\|_\infty$ so that $\boldsymbol{A}(\mathcal{S}) \subset [-M, M]^d$, and so, $\boldsymbol{A}$ is a ρ-JL embedding of $\mathcal{S}$ into $[-M, M]^d$.

By Lemma 2, there exists an α-Hölder function $g : [-M, M]^d \to \mathbb{R}^p$ such that $g(\boldsymbol{Ax}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{S}$. Let $g_i : [-M, M]^d \to \mathbb{R}$ be the i-th coordinate of g. Let $\widetilde{g}_i : [-1, 1]^d \to \mathbb{R}$ be defined by $\widetilde{g}_i(\boldsymbol{y}) = g_i(M\boldsymbol{y})$ for all $\boldsymbol{y} \in [-1, 1]^d$. Note that each g_i is α-Hölder, and so, each $\widetilde{g}_i$ is also α-Hölder.

Then, by Proposition 7, for each $\widetilde{g}_i$, there exists a CNN $\widetilde{g}_i^{(CNN)}$ with $O(N)$ residual blocks, each of which has depth $O(\log N)$ and $O(1)$ channels, and whose filter size is at most K such that $\|\widetilde{g}_i - \widetilde{g}_i^{(CNN)}\|_\infty \leq \widetilde{O}(N^{-\alpha/d})$.

Now, we construct a CNN to approximate f as follows. First, we implement the map $\boldsymbol{x} \mapsto \frac{1}{M}\boldsymbol{Ax}$ using the same 4 layer ResNet CNN described in the proof of Theorem 4c. Then, we pass the output of that Resnet CNN into p parallel CNNs which implement $\widetilde{g}_i^{(CNN)}$ for $i = 1, \ldots, p$. The output of the i-th of these parallel CNNs is $\widetilde{g}_i^{(CNN)}(\frac{1}{M}\boldsymbol{Ax})$, which is an $\widetilde{O}(N^{-\alpha/d})$-approximation of $\widetilde{g}_i(\frac{1}{M}\boldsymbol{Ax}) = g_i(\boldsymbol{Ax}) = f_i(\boldsymbol{x})$. Hence, the constructed CNN is a $\widetilde{O}(N^{-\alpha/d})$-approximation of f.

The CNN which implements the map $\boldsymbol{x} \mapsto \frac{1}{M}\boldsymbol{Ax}$ needs $O(1)$ residual blocks, each of which has depth $O(1)$ and $O(1)$ channels. Each of the p parallel CNNs which implement the $\widetilde{g}_i^{(CNN)}$'s have $O(N)$ residual blocks, each of which has depth $O(\log N)$ and $O(1)$ channels. So the overall network to approximate f has $O(pN)$ residual blocks, each of which has depth $O(\log N)$ and $O(1)$ channels.

J Proof of Proposition 8

Proof. By definition, the unit secants of $\mathcal{Y} = \boldsymbol{\Phi}(\Sigma_s^N)$ are defined to be

$$U_{\mathcal{Y}} = \left\{ \frac{\boldsymbol{y}_1 - \boldsymbol{y}_2}{\|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2}, \quad \boldsymbol{y}_1, \boldsymbol{y}_2 \in \boldsymbol{\Phi}(\Sigma_s^N) \right\}$$

which contains all unit vectors that are linear combinations of $2s$ columns of $\boldsymbol{\Phi}$. Letting T with $|T| = 2s$ be a fixed support set, the covering number of $\text{span}(\boldsymbol{\Phi}_T) \cap \mathbb{S}^{N-1}$ is $(\frac{3}{\delta})^{2s}$, so the covering number of $U_{\mathcal{Y}}$ is at most $\binom{N}{2s}(\frac{3}{\delta})^{2s} \leq (\frac{eN}{2s})^{2s}(\frac{3}{\delta})^{2s}$. Therefore,

$$\log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta) \lesssim s \log \frac{N}{s\delta}.$$

Now, let $\mathcal{S}$ be any bounded subset of $\mathcal{Y}$. Note that the inverse map F^{-1} is L-Lipschitz (i.e. (L, α)-Hölder for $\alpha = 1$). Since $\mathcal{S} \subset \mathcal{Y}$, we have that $U_{\mathcal{S}} \subset U_{\mathcal{Y}}$, and thus, $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta) \leq \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta)$ for any $\delta > 0$. Therefore, the choice of d in this proposition yields

$$d \gtrsim \rho^{-2} s \log \frac{N\sqrt{m}}{s\rho} \gtrsim \rho^{-2} \log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3m}}) \geq \rho^{-2} \log \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3m}}).$$

Therefore, the conditions of Theorem 6 are satisfied for $\alpha = 1$, and the result follows.

□

K Proof of Proposition 9

We first bound the logarithm of the covering number of the set of unit secants of $\mathcal{Y}$. Then, we apply Theorem 6 for the case of $\alpha = 1$, i.e. Lipschitz continuous functions.

By the $\sin\Theta$ theorem [40], we have

$$\left\| \frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|_2} - \frac{\boldsymbol{x}_2}{\|\boldsymbol{x}_2\|_2} \right\|_2 \leq \frac{\|\boldsymbol{x}_1\boldsymbol{k}_1^T - \boldsymbol{x}_2\boldsymbol{k}_2^T\|}{\|\boldsymbol{x}_1\|_2\|\boldsymbol{k}_1\|_2} \tag{7}$$

and

$$\left\| \frac{\boldsymbol{k}_1}{\|\boldsymbol{k}_1\|_2} - \frac{\boldsymbol{k}_2}{\|\boldsymbol{k}_2\|_2} \right\|_2 \leq \frac{\|\boldsymbol{x}_1\boldsymbol{k}_1^T - \boldsymbol{x}_2\boldsymbol{k}_2^T\|}{\|\boldsymbol{x}_2\|_2\|\boldsymbol{k}_2\|_2}. \tag{8}$$

Let us find a set whose covering number is easy to compute while containing the unit secant $U_{\mathcal{Y}}$ as a subset

$$\left\{ \frac{\boldsymbol{y}_1 - \boldsymbol{y}_2}{\|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2},\ \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{Y} \right\} = \left\{ \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2},\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}$$

$$= \left\{ \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2} + \frac{(\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2},\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}$$

$$\subseteq \left\{ \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2},\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, i = 1,2 \right\}$$

$$+ \left\{ \frac{(\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2},\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}.$$

For the first set in the sum, by using (3) and (7), we have

$$\left\{ \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - \boldsymbol{x}_2 \otimes \boldsymbol{k}_2\|_2},\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, i = 1,2 \right\}$$

$$\subseteq \left\{ t \cdot \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1}{\|\boldsymbol{x}_1\boldsymbol{k}_1^T - \boldsymbol{x}_2\boldsymbol{k}_2^T\|_2},\ t \in [0, L],\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}$$

$$\subseteq \left\{ t \cdot \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|}{\|\boldsymbol{x}_2\|}\boldsymbol{x}_2) \otimes \boldsymbol{k}_1}{\|\boldsymbol{x}_1 - \frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2\|\|\boldsymbol{k}_1\|},\ t \in [0, L],\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}$$

$$\subseteq \left\{ \left(\sqrt{t} \cdot \frac{\boldsymbol{x}_1 - \frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2}{\|\boldsymbol{x}_1 - \frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2\|_2} \right) \otimes \left(\sqrt{t} \cdot \frac{\boldsymbol{k}_1}{\|\boldsymbol{k}_1\|_2} \right), t \in [0, L],\ \boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{u}_i, \boldsymbol{k}_i = \boldsymbol{\Psi}\boldsymbol{v}_i, \boldsymbol{x}_i \otimes \boldsymbol{k}_i \in \mathcal{Y}, i = 1,2 \right\}$$

The covering number with ϵ balls of the set $\left\{ \sqrt{t} \cdot \frac{\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2)}{\|\boldsymbol{x}_1 \otimes \boldsymbol{k}_1 - (\frac{\|\boldsymbol{x}_1\|_2}{\|\boldsymbol{x}_2\|_2}\boldsymbol{x}_2)\|_2}, t \in [0, L] \right\}$ is $\left(\frac{3\sqrt{L}}{\epsilon}\right)^n$, and that for the set $\{\sqrt{t} \cdot \frac{\boldsymbol{k}_1}{\|\boldsymbol{k}_1\|_2}, t \in [0, L]\}$ is $\left(\frac{3\sqrt{L}}{\epsilon}\right)^m$. So the covering number with ϵ balls of S is

$$\left(\frac{6L}{\epsilon}\right)^n + \left(\frac{6L}{\epsilon}\right)^m.$$

The same argument holds for the second set in the sum. Therefore,

$$\log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta) \lesssim \max\{m, n\} \log \frac{L}{\delta}.$$

Now, let $\mathcal{S}$ be any bounded subset of $\mathcal{Y}$. Note that the inverse map F^{-1} is L-Lipschitz (i.e. (L, α)-Hölder for $\alpha = 1$). Since $\mathcal{S} \subset \mathcal{Y}$, we have that $U_{\mathcal{S}} \subset U_{\mathcal{Y}}$, and thus, $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta) \leq \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta)$ for any $\delta > 0$. Therefore, the choice of d in this proposition yields

$$d \gtrsim \rho^{-2} \max\{m, n\} \log \frac{L\sqrt{N}}{\rho} \gtrsim \rho^{-2} \log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3N}}) \geq \rho^{-2} \log \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3N}}).$$

Therefore, the conditions of Theorem 6 are satisfied for $\alpha = 1$, and the result follows.

L Proof of Proposition 10

By definition,

$$U_{\mathcal{Y}} = \left\{ \frac{\boldsymbol{y}_1 - \boldsymbol{y}_2}{\|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2}, \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{Y} \right\} = \left\{ \frac{P_\Omega(\boldsymbol{X}_1 - \boldsymbol{X}_2)}{\|P_\Omega(\boldsymbol{X}_1 - \boldsymbol{X}_2)\|_F}, \boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathcal{Y} \right\}$$

Since $\boldsymbol{y}_1 - \boldsymbol{y}_2 = P_\Omega(\boldsymbol{X}_1 - \boldsymbol{X}_2)$ and $\boldsymbol{X}$

$$\left\{ \frac{P_\Omega(\boldsymbol{X}_1 - \boldsymbol{X}_2)}{\|P_\Omega(\boldsymbol{X}_1 - \boldsymbol{X}_2)\|_F}, \boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathcal{Y} \right\} \subseteq \left\{ t \cdot P_\Omega \left(\frac{\boldsymbol{X}_1 - \boldsymbol{X}_2}{\|\boldsymbol{X}_1 - \boldsymbol{X}_2\|_F} \right), t \in [0, L], \boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathcal{Y} \right\}$$

Notice that $\frac{\boldsymbol{X}_1 - \boldsymbol{X}_2}{\|\boldsymbol{X}_1 - \boldsymbol{X}_2\|_F}$ are matrices of unit Frobenius norm with rank at most $2r$. By Lemma 3.1 in [41], they form a set whose covering number is at most $\left(\frac{9}{\delta}\right)^{r(m+n+1)}$. Hence, by dilating the set by a factor of L, the covering number is at most $\left(\frac{9L}{\delta}\right)^{r(m+n+1)}$. Therefore,

$$\log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta) \lesssim r(m+n) \log \frac{L}{\delta}.$$

Now, let $\mathcal{S}$ be any bounded subset of $\mathcal{Y}$. Note that the inverse map F^{-1} is L-Lipschitz (i.e. (L, α)-Hölder for $\alpha = 1$). Since $\mathcal{S} \subset \mathcal{Y}$, we have that $U_{\mathcal{S}} \subset U_{\mathcal{Y}}$, and thus, $\mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \delta) \leq \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \delta)$ for any $\delta > 0$. Therefore, the choice of d in this proposition yields

$$d \gtrsim \rho^{-2} r(m+n) \log \frac{L\sqrt{mn}}{\rho} \gtrsim \rho^{-2} \log \mathcal{N}(U_{\mathcal{Y}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3mn}}) \geq \rho^{-2} \log \mathcal{N}(U_{\mathcal{S}}, \|\cdot\|_2, \tfrac{\rho}{4\sqrt{3mn}}).$$

Therefore, the conditions of Theorem 6 are satisfied for $\alpha = 1$, and the result follows.