The difficulty of computing stable and accurate neural networks: On the barriers of deep learning and Smale's 18th problem

Matthew J. Colbrook[a,1,2], Vegard Antun[b,1,2], and Anders C. Hansen[a,b,2]

[a]Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge CB3 0WA, United Kingdom; and [b]Department of Mathematics, University of Oslo, 0316 Oslo, Norway

Edited by Ronald DeVore, Texas A&M University, College Station, TX; received April 16, 2021; accepted October 26, 2021

Deep learning (DL) has had unprecedented success and is now entering scientific computing with full force. However, current DL methods typically suffer from instability, even when universal approximation properties guarantee the existence of stable neural networks (NNs). We address this paradox by demonstrating basic well-conditioned problems in scientific computing where one can prove the existence of NNs with great approximation qualities; however, there does not exist any algorithm, even randomized, that can train (or compute) such a NN. For any positive integers $K > 2$ and L, there are cases where simultaneously 1) no randomized training algorithm can compute a NN correct to K digits with probability greater than 1/2; 2) there exists a deterministic training algorithm that computes a NN with $K - 1$ correct digits, but any such (even randomized) algorithm needs arbitrarily many training data; and 3) there exists a deterministic training algorithm that computes a NN with $K - 2$ correct digits using no more than L training samples. These results imply a classification theory describing conditions under which (stable) NNs with a given accuracy can be computed by an algorithm. We begin this theory by establishing sufficient conditions for the existence of algorithms that compute stable NNs in inverse problems. We introduce fast iterative restarted networks (FIRENETs), which we both prove and numerically verify are stable. Moreover, we prove that only $\mathcal{O}(|\log(\epsilon)|)$ layers are needed for an ϵ-accurate solution to the inverse problem.

stability and accuracy | AI and deep learning | inverse problems | Smale's 18th problem | solvability complexity index hierarchy

Deep learning (DL) has demonstrated unparalleled accomplishments in fields ranging from image classification and computer vision (1–3), to voice recognition and automated diagnosis in medicine (4–6), to inverse problems and image reconstruction (7–12). However, there is now overwhelming empirical evidence that current DL techniques typically lead to unstable methods, a phenomenon that seems universal and present in all of the applications listed above (13–21) and in most of the new artificial intelligence (AI) technologies. These instabilities are often detected by what has become commonly known in the literature as "adversarial attacks." Moreover, the instabilities can be present even in random cases and not just worst-case scenarios (22)—see Fig. 1 for an example of AI-generated hallucinations. There is a growing awareness of this problem in high-stakes applications and society as a whole (20, 23, 24), and instability seems to be the Achilles' heel of modern AI and DL (Fig. 2, *Top* row). For example, this is a problem in real-world clinical practice. Facebook and New York University's 2019 FastMRI challenge reported that networks that performed well in terms of standard image quality metrics were prone to false negatives, failing to reconstruct small, but physically relevant image abnormalities (25). Subsequently, the 2020 FastMRI challenge (26) focused on pathologies, noting, "Such hallucinatory features are not acceptable and especially problematic if they mimic normal structures that are either not present or actually abnormal. Neural network models can be unstable as demonstrated via adversarial perturbation studies (19)." For similar examples in microscopy, see refs. 27 and 28. The tolerance level for false positives/negatives varies within different applications. However, for scenarios with a high cost of misanalysis, it is imperative that false negatives/positives be avoided. AI-generated hallucinations therefore pose a serious danger in applications such as medical diagnosis.

Nevertheless, classical approximation theorems show that a continuous function can be approximated arbitrarily well by a neural network (NN) (29, 30). Thus, stable problems described by stable functions can always be solved stably with a NN. This leads to the following fundamental question:

Question. *Why does DL lead to unstable methods and AI-generated hallucinations, even in scenarios where one can prove that stable and accurate neural networks exist*?

Foundations of AI for Inverse Problems. To answer the above question we initiate a program on the foundations of AI, determining the limits of what DL can achieve in inverse problems. It is crucial to realize that an existence proof of suitable NNs does not always imply that they can be constructed by a training algorithm.

Significance

Instability is the Achilles' heel of modern artificial intelligence (AI) and a paradox, with training algorithms finding unstable neural networks (NNs) despite the existence of stable ones. This foundational issue relates to Smale's 18th mathematical problem for the 21st century on the limits of AI. By expanding methodologies initiated by Gödel and Turing, we demonstrate limitations on the existence of (even randomized) algorithms for computing NNs. Despite numerous existence results of NNs with great approximation properties, only in specific cases do there also exist algorithms that can compute them. We initiate a classification theory on which NNs can be trained and introduce NNs that—under suitable conditions—are robust to perturbations and exponentially accurate in the number of hidden layers.

Author contributions: M.J.C., V.A., and A.C.H. designed research, performed research, and wrote the paper.

The authors declare no competing interest.

This article is a PNAS Direct Submission.

[1]M.J.C. and V.A. contributed equally to this work.

[2]To whom correspondence may be addressed. Email: m.colbrook@damtp.cam.ac.uk, vegarant@math.uio.no, or a.hansen@damtp.cam.ac.uk.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2107151119/-/DCSupplemental.

Published March 16, 2022.

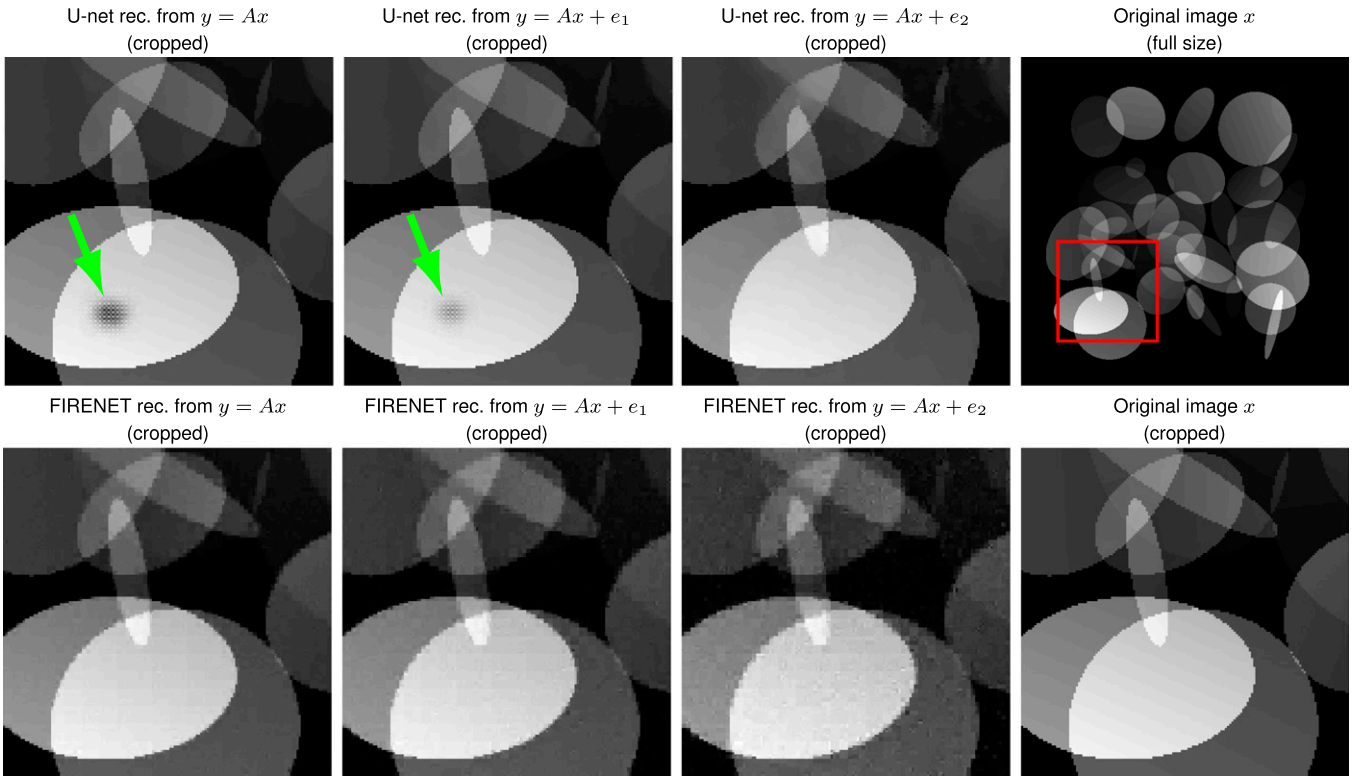

Fig. 1. AI-generated hallucinations. A trained NN, based on a U-net architecture and trained on a set of ellipses images, generates a black area in a white ellipse (*Top Left* image, shown as green arrow) when reconstructing the original image x from noiseless measurements. By adding random Gaussian noise e_1 and e_2 (where $\|e_1\|_{l^2}/\|e_2\|_{l^2} \approx 2/5$) to the measurements, we see that the trained NN removes the aspiring black ellipse (*Top* row, *Center Left* to *Center Right*). FIRENET on the other hand is completely stable with and without random Gaussian noise (*Bottom* row, *Left* to *Center Right*). In *Right* column, we show the original image x, with a red square (*Top Right*) indicating the cropped area. In this example, $A \in \mathbb{C}^{m\times N}$ is a subsampled discrete Fourier transform with $m/N \approx 0.12$.

Furthermore, it is not difficult to compute stable NNs. For example, the zero network is stable, but not particularly useful. The big problem is to compute NNs that are both stable and accurate (30, 31). Scientific computing itself is based on the pillars of stability and accuracy. However, there is often a trade-off between the two. There may be barriers preventing the existence of stable and accurate algorithms, and sometimes accuracy must be sacrificed to secure stability.

Main Results. We consider the canonical inverse problem of an underdetermined system of linear equations:

$$\text{Given measurements } y = Ax + e \in \mathbb{C}^m, \text{ recover } x \in \mathbb{C}^N. \quad \textbf{[1]}$$

Here, $A \in \mathbb{C}^{m\times N}$ represents a sampling model ($m < N$), such as a subsampled discrete Fourier transform in MRI, and x the unknown quantity. The problem in Eq. **1** forms the basis for much of inverse problems and image analysis. The vector e models noise or perturbations. Our results demonstrate fundamental barriers preventing NNs (despite their existence) from being computed by algorithms. This helps shed light on the intricate question of why current algorithms in DL produce unstable networks, despite the fact that stable NNs often exist in the particular application. We show the following:

1) *Theorems 1* and *2*: There are well-conditioned problems (suitable condition numbers bounded by 1) where, paradoxically, mappings from training data to suitable NNs exist, but no training algorithm (even randomized) can compute approximations of the NNs from the training data.
2) *Theorem 2*: The existence of algorithms computing NNs depends on the desired accuracy. For any $K \in \mathbb{Z}_{\geq 3}$, there are well-conditioned classes of problems where simultaneously 1) algorithms may compute NNs to $K-1$ digits of accuracy, but not K; 2) achieving $K-1$ digits of accuracy requires arbitrarily many training data; and 3) achieving $K-2$ correct digits requires only one training datum.
3) *Theorems 3* and *4*: Under specific conditions that are typically present in, for example, MRI, there are algorithms that compute stable NNs for the problem in Eq. **1**. These NNs, which we call fast iterative restarted networks (FIRENETs), converge exponentially in the number of hidden layers. Crucially, we prove that FIRENETs are robust to perturbations (Fig. 2, *Bottom* row), and they can even be used to stabilize unstable NNs (Fig. 3).
4) There is a trade-off between stability and accuracy in DL, with limits on how well a stable NN can perform in inverse problems. Fig. 4 demonstrates this with a U-net trained on images consisting of ellipses that is quite stable. However, when a detail not in the training set is added, it washes it out almost entirely. FIRENETs offer a blend of both stability and accuracy. However, they are by no means the end of the story. Tracing out the optimal stability vs. accuracy trade-off is crucial for applications and will no doubt require a myriad of different techniques to tackle different problems and stability tolerances.

Fundamental Barriers

We first consider basic mappings used in modern mathematics of information, inverse problems, and optimization. Given a matrix $A \in \mathbb{C}^{m\times N}$ and a vector $y \in \mathbb{C}^m$, we consider the following three popular minimization problems:

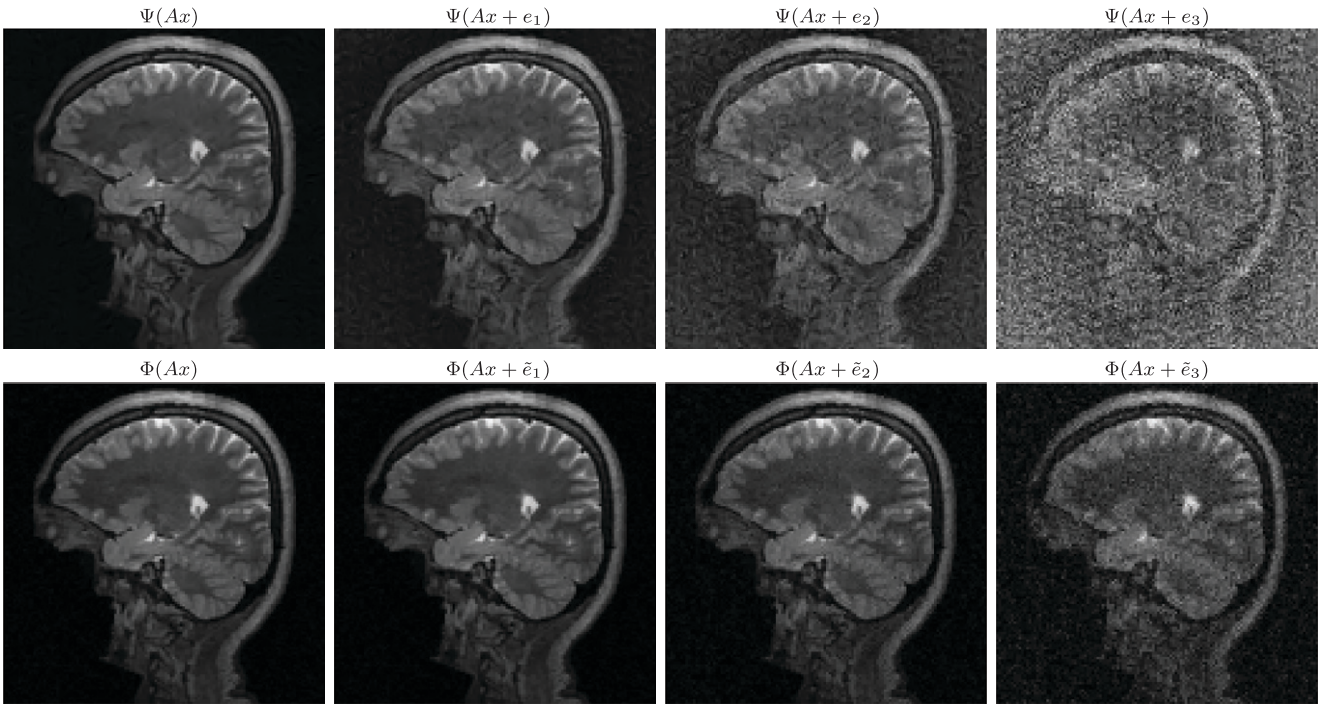

Fig. 2. *Top* row (unstable neural network in image reconstruction): The neural network AUTOMAP (60) represents the tip of the iceberg of DL in inverse problems. Ref. 60, pp. 1 and 487, promises that one can "…observe superior immunity to noise…." Moreover, the follow-up announcement (ref. 83, pp. 1 and 309) proclaims "A deep-learning-based approach improves speed, accuracy and robustness of biomedical image reconstruction." However, as we see in *Top* row, the AUTOMAP reconstruction $\Psi(Ax+e_j)$ from the subsampled noisy Fourier MRI data $Ax+e_j$ is completely unstable. Here, $A \in \mathbb{C}^{m\times N}$ is a subsampled Fourier transform, x is the original image, and the e_j s are perturbations meant to simulate the worst-case effect. Note that the condition number $\mathrm{cond}(AA^*)=1$, so the instabilities are not caused by poor condition. The network weights were provided by the authors of ref. 60, which trained and tested it on brain images from the Massachusetts General Hospital Human Connectome Project (MGH-USC HCP) dataset (84). The image x is taken from this dataset. *Bottom* row (the FIRENET is stable to worst-case perturbations): Using the same method, we compute perturbations $\tilde{e}_j$ to simulate the worst-case effect for the FIRENET $\Phi\colon \mathbb{C}^m \to \mathbb{C}^N$. As can be seen, FIRENET is stable to these worst-case perturbations. Here x and $A \in \mathbb{C}^{m\times N}$ are the same image and sampling matrix as for AUTOMAP. Moreover, for each $j=1,2,3$ we have ensured that $\|\tilde{e}_j\|_{l^2} \geq \|e_j\|_{l^2}$, where the e_j s are the perturbations for AUTOMAP (we have denoted the perturbations for FIRENET by $\tilde{e}_j$ to emphasize that these adversarial perturbations are sought for FIRENET and have nothing to do with the perturbations for AUTOMAP).

$$(P_1)\quad \mathrm{argmin}_{x\in\mathbb{C}^N}\ F_1^A(x) := \|x\|_{l_w^1}, \text{s.t.} \|Ax-y\|_{l^2} \leq \epsilon,$$

$$(P_2)\quad \mathrm{argmin}_{x\in\mathbb{C}^N}\ F_2^A(x,y,\lambda) := \lambda\|x\|_{l_w^1} + \|Ax-y\|_{l^2}^2,$$

$$(P_3)\quad \mathrm{argmin}_{x\in\mathbb{C}^N}\ F_3^A(x,y,\lambda) := \lambda\|x\|_{l_w^1} + \|Ax-y\|_{l^2},$$

known respectively as quadratically constrained basis pursuit [we always assume existence of a feasible x for (P_1)], unconstrained least absolute shrinkage and selection operator (LASSO), and unconstrained square-root LASSO. Such sparse regularization problems are often used as benchmarks for Eq. **1**, and we prove impossibility results for computing the NNs that can approximate these mappings. Our results initiate a classification theory on which NNs can be computed to a certain accuracy.

The parameters λ and ϵ are positive rational numbers, and the weighted l_w^1 norm is given by $\|x\|_{l_w^1} := \sum_{l=1}^N w_l|x_l|$, where each weight w_j is a positive rational. Throughout, we let

$$\Xi_j(A,y) \text{ be the set of minimizers for } (P_j). \qquad \textbf{[2]}$$

Let $A \in \mathbb{C}^{m\times N}$ and let $\mathcal{S} = \{y_k\}_{k=1}^R \subset \mathbb{C}^m$ be a collection of samples ($R \in \mathbb{N}$). We consider the following key question:

Question. *Given a collection Ω of pairs $(A,\mathcal{S})$, does there exist a neural network approximating Ξ_j, and if so, can such an approximation be trained or determined by an algorithm*?

To make this question precise, note that A and samples in $\mathcal{S}$ will typically never be exact, but can be approximated/stored to arbitrary precision. For example, this would occur if A was a subsampled discrete cosine transform. Thus, we assume access to rational approximations $\{y_{k,n}\}_{k=1}^R$ and A_n with

$$\|y_{k,n}-y_k\|_{l^2} \leq 2^{-n}, \quad \|A_n - A\| \leq 2^{-n}, \quad \forall n \in \mathbb{N}, \qquad \textbf{[3]}$$

where $\|\cdot\|$ refers to the usual Euclidean operator norm. The bounds 2^{-n} are simply for convenience and can be replaced by any other sequence converging to zero. We also assume access to rational $\{x_{k,n}\}_{k=1}^R$ with

$$\inf_{x^*\in\Xi_j(A_n,y_{k,n})} \|x_{k,n}-x^*\|_{l^2} \leq 2^{-n}, \quad \forall n \in \mathbb{N}. \qquad \textbf{[4]}$$

Hence, the training data associated with $(A,\mathcal{S}) \in \Omega$ must be

$$\iota_{A,\mathcal{S}} := \{(y_{k,n}, A_n, x_{k,n}) \mid k=1,\dots,R, \text{and } n \in \mathbb{N}\}. \qquad \textbf{[5]}$$

This set is formed of arbitrary precision rational approximations of finite collections of data associated with $(A,\mathcal{S})$. Given a collection Ω of pairs $(A,\mathcal{S})$, the class of all such admissible training data is denoted by

$$\Omega_{\mathcal{T}} := \{\iota_{A,\mathcal{S}} \text{ as in Eq. 5} \mid (A,\mathcal{S}) \in \Omega, \text{Eqs. 3 to 4 hold}\}.$$

Statements addressing the above question are summarized in *Theorems 1* and *2*. We use $\mathcal{N}_{m,N}$ to denote the class of NNs from $\mathbb{C}^m$ to $\mathbb{C}^N$. We use standard definitions of feedforward NNs (32), precisely given in *SI Appendix*.

Theorem 1. *For any collection Ω of such $(A, \mathcal{S})$ described above, there exists a mapping*

$$\mathcal{K}\colon \Omega_{\mathcal{T}} \to \mathcal{N}_{m,N}, \quad \mathcal{K}(\iota_{A,\mathcal{S}}) = \varphi_{A,\mathcal{S}},$$
$$s.t. \quad \varphi_{A,\mathcal{S}}(y) \in \Xi_j(A, y), \quad \forall y \in \mathcal{S}.$$

In words, $\mathcal{K}$ maps the training data $\Omega_{\mathcal{T}}$ to NNs that solve the optimization problem (P_j) for each $(A, \mathcal{S}) \in \Omega$.

Despite the existence of NNs guaranteed by *Theorem 1*, computing or training such a NN from training data is most delicate. The following is stated precisely and proved in *SI Appendix*. We also include results for randomized algorithms, which are common in DL (e.g., stochastic gradient descent).

Theorem 2. *Consider the optimization problem (P_j) for fixed parameters $\lambda \in (0, 1]$ or $\epsilon \in (0, 1/2]$ and $w_l = 1$, where $N \geq 2$ and $m < N$. Let $K > 2$ be a positive integer and let $L \in \mathbb{N}$. Then there exists an infinite class $\Omega = \Omega(K, L)$ of elements $(A, \mathcal{S})$ as above, with the following properties. The class Ω is well-conditioned with relevant condition numbers bounded by 1 independent of all parameters. However, the following hold simultaneously (where accuracy is measured in the l^2 norm):*

1) (*K digits of accuracy impossible*) *There does not exist any algorithm that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$, produces a NN with K digits of accuracy for any element of $\mathcal{S}$. Furthermore, for any $p > 1/2$, no probabilistic algorithm* (Blum–Shub–Smale [*BSS*], *Turing, or any model of computation*) *can produce a NN with K digits of accuracy with probability at least p.*
2) (*$K - 1$ digits of accuracy possible but requires arbitrarily many training data*) *There does exist a deterministic Turing machine that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$, produces a NN accurate to $K - 1$ digits over $\mathcal{S}$. However, for any probabilistic Turing machine, $M \in \mathbb{N}$ and $p \in \left[0, \frac{N-m}{N+1-m}\right)$ that produces a NN, there exists a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$ such that for all $y \in \mathcal{S}$, the probability of failing to achieve $K - 1$ digits or requiring more than M training data is greater than p.*
3) (*$K - 2$ digits of accuracy possible with L training data*) *There does exist a deterministic Turing machine that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$ and using only L training data from each $\iota_{A,\mathcal{S}}$, produces a NN accurate to $K - 2$ digits over $\mathcal{S}$.*

Remark 1 (condition and class size). The statement in Theorem 2 refers to the standard condition numbers used in optimization and scientific computing. For precise definitions, see *SI Appendix*. The class Ω we construct is infinite. Similarly, one can design a finite class Ω with the same conclusion by allowing the sample size R to be infinite.

Remark 2 (distributions on training data). In DL it is often the case that one assumes some probability distribution on the training data. This is not needed for Theorem 2. However, having a probability distribution on the training data $\iota_{A,\mathcal{S}}$ would not invalidate statement 1 in Theorem 2. In particular, there is no (computable) probability distribution that would make statement 1 in Theorem 2 cease to be true. This follows from the probabilistic part of statement 1 in Theorem 2, as the existence of such a (computable) distribution and an algorithm would yield a randomized algorithm violating statement 1 in Theorem 2.

Remark 3 (on the role of K in Theorem 2). The result should be understood as fixing an integer K (and L) and then $\Omega = \Omega(K, L)$ depends on K and L. However, given a particular Ω one can ask, what is the largest K such that one can compute K correct digits? Note that we typically have $K = \lfloor \log(\epsilon^{-1}) \rfloor$, where $\epsilon > 0$ is the so-called breakdown epsilon of the problem (33), i.e., the largest $\epsilon > 0$ for which all algorithms will fail to provide ϵ accuracy. When the breakdown epsilon $\epsilon > 0$, it is typically impossible to check whether an algorithm fails (33). Thus, even if an algorithm would succeed with probability $1/2$, one could never trust the output.

Table 1. Impossibility of computing approximations of the existing neural network to arbitrary accuracy

Ψ_{A_n}	Φ_{A_n}	$\|A_n - A\| \leq 2^{-n}$ $\|y_n - y\|_{l^2} \leq 2^{-n}$	10^{-K}	$\Omega(K)$
0.2999690	0.2597827	$n = 10$	10^{-1}	$K = 1$
0.3000000	0.2598050	$n = 20$	10^{-1}	$K = 1$
0.3000000	0.2598052	$n = 30$	10^{-1}	$K = 1$
0.0030000	0.0025980	$n = 10$	10^{-3}	$K = 3$
0.0030000	0.0025980	$n = 20$	10^{-3}	$K = 3$
0.0030000	0.0025980	$n = 30$	10^{-3}	$K = 3$
0.0000030	0.0000015	$n = 10$	10^{-6}	$K = 6$
0.0000030	0.0000015	$n = 20$	10^{-6}	$K = 6$
0.0000030	0.0000015	$n = 30$	10^{-6}	$K = 6$

We demonstrate statement 1 from Theorem 2 on FIRENETs Φ_{A_n} and LISTA networks Ψ_{A_n}. Shown is the shortest l^2 distance between the output from the networks and the true solution of the problem (P_3), with $w_l = 1$ and $\lambda = 1$, for different values of n and K. Note that none of the networks can compute the existing correct NN (that exists by Theorem 1 and coincides with Ξ_3) to 10^{-K} digits accuracy, while all of them are able to compute approximations that are accurate to 10^{-K+1} digits [for the input class $\Omega(K)$]. This agrees exactly with Theorem 2.

Remark 4 (Gödel, Turing, Smale, and Theorem 2). Theorems 1 and 2 demonstrate basic limitations on the existence of algorithms that can compute NNs despite their existence. This relates to Smale's 18th problem, "What are the limits of intelligence, both artificial and human?", from the list of mathematical problems for the 21st century (34), which echoes the Turing test from 1950 (35). Smale's discussion is motivated by the results of Gödel (36) and Turing (37) establishing impossibility results on what mathematics and digital computers can achieve (38). Our results are actually stronger, however, than what can be obtained with Turing's techniques. Theorem 2 holds even for any randomized Turing or BSS machine that can solve the halting problem. It immediately opens up for a classification theory on which NNs can be computed by randomized algorithms. Theorem 3 is a first step in this direction. See also the work by Niyogi, Smale, and Weinberger (39) on existence results of algorithms for learning.

Numerical Example. To highlight the impossibility of computing NNs (Theorem 2)—despite their existence by Theorem 1—we consider the following numerical example: Consider the problem (P_3), with $w_l = 1$ and $\lambda = 1$. Theorem 2 is stated for a specific input class $\Omega = \Omega(K)$ depending on the accuracy parameter K, and in this example we consider three different such classes. In Theorem 2, we required that $K > 2$ so that $K - 2 > 0$, but this is not necessary to show the impossibility statement 1, so we consider $K = 1, 3, 6$. Full details of the following experiment are given in *SI Appendix*.

To show that it is impossible to compute NNs that can solve (P_3) to arbitrary accuracy we consider FIRENETs Φ_{A_n} (the NNs in Theorem 3) and learned ISTA (LISTA) networks Ψ_{A_n} based on the architecture choice from ref. 40. The networks are trained to high accuracy on training data on the form of Eq. **5** with $R = 8,000$ training samples and n given as in Table 1. In all cases $N = 20$, $m = N - 1$, and the $x_{k,n}$ s minimizing (P_3) with input data $(y_{k,n}, A_n)$ are all 6-sparse. The choice of N, m, and sparsity is to allow for fast training; other choices are certainly possible.

Table 1 shows the errors for both LISTA and FIRENETs. Both network types are given input data (y_n, A_n), approximating the true data (y, A). As is clear from Table 1, none of the networks are able to compute an approximation to the true minimizer in $\Xi_3(A, y)$ to K digits accuracy. However, both networks compute an approximation with $K - 1$ digits accuracy. These observations agree precisely with Theorem 2.

The Subtlety and Difficulty of Removing Instabilities and the Need for Additional Assumptions. Theorem 2 shows that the problems (P_j) cannot, in general, be solved by any training algorithm. Hence, any attempt at using the problems (P_j) as approximate solution maps of the general inverse problem in Eq. **1**, without additional assumptions, is doomed to fail. This is not just the case for reconstruction using sparse regularization, but also applies to other methods. In fact, any stable and accurate reconstruction procedure must be "kernel aware" (22), a property that most DL methods do not enforce. A reconstruction method $\Psi : \mathbb{C}^m \to \mathbb{C}^N$ lacks kernel awareness if it approximately recovers two vectors

$$\|\Psi(Ax) - x\| \leq \epsilon \quad \text{and} \quad \|\Psi(Ax') - x'\| \leq \epsilon \qquad \textbf{[6]}$$

whose difference $\|x - x'\| \gg 2\epsilon$ is large, but where the difference lies close to the null space of A (which is nontrivial due to $m < N$) so that $\|A(x - x')\| < \epsilon$. In particular, by applying Eq. **6** and the triangle inequality twice, we have that

$$\|\Psi(Ax) - \Psi(Ax')\| \geq \|x - x'\| - 2\epsilon \qquad \textbf{[7]}$$

implying instability, as it requires only a perturbation $e = A(x' - x)$ of size $\|e\| < \epsilon$ for $\Psi(Ax + e) = \Psi(Ax')$ to reconstruct the wrong image. The issue here is that if we want to accurately recover x and x', i.e., we want Eq. **6** to hold, then we cannot simultaneously have that $x - x'$ lies close to the kernel. Later we shall see conditions that circumvent this issue for our model class, thereby allowing us to compute stable and accurate NNs.

While training can encourage the conditions in Eq. **6** to hold, it is not clear how many of the defense techniques in DL, simultaneously, will protect against the condition $\|A(x - x')\| < \epsilon$. One standard attempt to remedy instabilities is adversarial training (41). However, while this strategy can potentially avoid Eq. **6**, it may yield poor performance. For example, consider the following optimization problem, which generates a reconstruction in the form of a NN given samples $\Theta = \{(y_s, x_s) : s = 1, \dots, R, Ax_s = y_s\}$ and $\epsilon, \lambda > 0$:

$$\min_{\phi \in \mathcal{N}_{m,N}} \sum_{s=1}^{R} \max_{\|e\|_{l^2} \leq \epsilon} \{\|x_s - \phi(y_s)\|_{l^2}^2 + \lambda \|x_s - \phi(y_s + e)\|_{l^2}^2\}. \qquad \textbf{[8]}$$

In other words, for each training point $(y, x) \in \Theta$ we find the worst-case perturbation e in the ϵ -ball around y. This is a simplified model of what one might do using generative adversarial networks (GANs) to approximate adversarial perturbations (42, 43). For simplicity, assume that A has full row rank m and that we have access to exact measurements $y_s = Ax_s$. Suppose that our sample is such that $\min_{i \neq j} \|y_i - y_j\|_{l^2} > 2\epsilon$. In this case, ϕ minimizes Eq. **8** if and only if $\phi(y_s + e) = x_s$ for all e with $\|e\|_{l^2} \leq \epsilon$. A piecewise affine network achieving this can easily be constructed using ReLU (rectified linear unit) activation functions. Now suppose that x_2 is altered so that $x_1 - x_2$ lies in the kernel of A. Then for any minimizer ϕ, we must have $\phi(y_1 + e) = \phi(y_2 + e) = (x_1 + x_2)/2$ for any e with $\|e\|_{l^2} \leq \epsilon$, and hence we can never be more than $\|x_1 - \phi(y_1)\| = \|x_1 - x_2\|_{l^2}/2$ accurate over the whole test sample. Similar arguments apply to other methods aimed at improving robustness such as adding noise to training samples (known as "jittering") (Fig. 4). Given such examples and Theorem 2, we arrive at the following question:

Question. *Are there sufficient conditions on A that imply the existence of an algorithm that can compute a neural network that is both stable and accurate for the problem in Eq.* **1**?

Sufficient Conditions for Algorithms to Compute Stable and Accurate NNs

Sparse regularization, such as the problems (P_j), forms the core of many start-of-the-art reconstruction algorithms for inverse problems. We now demonstrate a sufficient condition (from compressed sensing) guaranteeing the existence of algorithms for stable and accurate NNs. Sparsity in levels is a standard sparsity model for natural images (44–47) as images are sparse in levels in X-lets (wavelets, curvelets, shearlets, etc.).

Definition 1 (Sparsity in Levels). *Let $\mathbf{M} = (M_1, \dots, M_r) \in \mathbb{N}^r$, $1 \leq M_1 < \dots < M_r = N$, and $\mathbf{s} = (s_1, \dots, s_r) \in \mathbb{N}_0^r$, where $s_l \leq M_l - M_{l-1}$ for $l = 1, \dots, r$ ($M_0 = 0$). $x \in \mathbb{C}^N$ is $(\mathbf{s}, \mathbf{M})$ -sparse in levels if $|\mathrm{supp}(x) \cap \{M_{l-1} + 1, \dots, M_l\}| \leq s_l$ for $l = 1, \dots, r$. The total sparsity is $s = s_1 + \dots + s_r$. We denote the set of $(\mathbf{s}, \mathbf{M})$ -sparse vectors by $\Sigma_{\mathbf{s},\mathbf{M}}$. We also define the following measure of distance of a vector x to $\Sigma_{\mathbf{s},\mathbf{M}}$ by*

$$\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} = \inf\{\|x - z\|_{l_w^1} : z \in \Sigma_{\mathbf{s},\mathbf{M}}\}.$$

This model has been used to explain the effectiveness of compressed sensing (46, 48–52) in real-life applications (53). For simplicity, we assume that each $s_l > 0$ and that $w_i = w_{(l)}$ if $M_{l-1} + 1 \leq i \leq M_l$ (the weights in the l_w^1 norm are constant in each level). For a vector c that is compressible in the wavelet basis, $\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1}$ is expected to be small if x is the vector of wavelet coefficients of c and the levels correspond to wavelet levels (54). In general, the weights are a prior on anticipated support of the vector (55), and we discuss some specific optimal choices in *SI Appendix*.

For $\mathcal{I} \subset \{1, \dots, N\}$, let $P_\mathcal{I} \in \mathbb{C}^{N \times N}$ denote the projection $(P_\mathcal{I} x)_i = x_i$ if $i \in \mathcal{I}$ and $(P_\mathcal{I} x)_i = 0$ otherwise. The key kernel-aware property that allows for stable and accurate recovery of $(\mathbf{s}, \mathbf{M})$ -spare vectors for the inverse problem Eq. **1** is the weighted robust null space property in levels (wrNSPL):

Definition 2 (wrNSPL). *Let $(\mathbf{s}, \mathbf{M})$ be local sparsities and sparsity levels, respectively. For weights $\{w_i\}_{i=1}^N$, $A \in \mathbb{C}^{m \times N}$ satisfies the wrNSPL of order $(\mathbf{s}, \mathbf{M})$ with constants $0 < \rho < 1$ and $\gamma > 0$ if for any $(\mathbf{s}, \mathbf{M})$ support set $\mathcal{I} \subset \{1, \dots, N\}$ (with complement $\mathcal{I}^c = \{1, \dots, N\} \backslash \mathcal{I}$),*

$$\|P_\mathcal{I} x\|_{l^2} \leq \frac{\rho \|P_{\mathcal{I}^c} x\|_{l_w^1}}{\sqrt{\sum_{l=1}^r w_{(l)}^2 s_l}} + \gamma \|Ax\|_{l^2}, \qquad \text{for all } x \in \mathbb{C}^N.$$

We highlight that if A satisfies the wrNSPL, then

$$\|x - x'\|_{l^2} \leq C \|A(x - x')\|_{l^2}, \quad \forall x, x' \in \Sigma_{\mathbf{s},\mathbf{M}},$$

where $C = C(\rho, \gamma) > 0$ is a constant depending only on ρ and γ (*SI Appendix*). This ensures that if $\|x - x'\|_{\ell^2} \gg 2\epsilon$, then we cannot, simultaneously, have that $\|A(x - x')\| < \epsilon$, causing the instability in Eq. **7**. Below, we give natural examples of sampling in compressed imaging where such a property holds, for known ρ and γ, with large probability. We can now state a simplified version of our result (the full version with explicit constants is given and proved in *SI Appendix*):

Theorem 3. *There exists an algorithm such that for any input sparsity parameters $(\mathbf{s}, \mathbf{M})$, weights $\{w_i\}_{i=1}^N$, $A \in \mathbb{C}^{m \times N}$ (with the input A given by $\{A_l\}$) satisfying the wrNSPL with constants $0 < \rho < 1$ and $\gamma > 0$ (also input), and input parameters $n \in \mathbb{N}$ and $\{\delta, b_1, b_2\} \subset \mathbb{Q}_{>0}$, the algorithm outputs a neural network ϕ_n with $\mathcal{O}(n)$ hidden layers and $\mathcal{O}(N)$ width with the following property: For any $x \in \mathbb{C}^N$, $y \in \mathbb{C}^m$ with*

$$\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + \|Ax - y\|_{l^2} \lesssim \delta, \quad \|x\|_{l^2} \lesssim b_1, \quad \|y\|_{l^2} \lesssim b_2,$$

we have $\|\phi_n(y) - x\|_{l^2} \lesssim \delta + e^{-n}$.

Hence, up to the small error term $\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1}$, as $n \to \infty$ (with exponential convergence), we recover x stably with an error proportional to the measurement error $\|Ax - y\|_{l^2}$. The explicit constant in front of the $\|Ax - y\|_{l^2}$ term can be thought of as

an asymptotic local Lipschitz constant for the NNs as $n \to \infty$ and thus measures stability of inexact input y. The error of order $\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1}$ measures how close the vector x is from the model class of sparse in levels vectors. In the full version of Theorem 3, we also bound the error when we only approximately apply the nonlinear maps of the NNs and show that these errors can only accumulate slowly as n increases. In other words, we also gain a form of numerical stability of the forward pass of the NN. We call our NNs FIRENETs.

Remark 5 (unrolling does not in general yield an algorithm producing an accurate network). Unrolling iterative methods has a rich history in DL (9, 56). Note, however, that Theorem 2 demonstrates that despite the existence of an accurate neural network, there are scenarios where no algorithm exists that can compute it. Thus, unrolling optimization methods can work only under certain assumptions. Our results are related to the work of Ben-Tal and Nemirovski (57) (see also ref. 58), which shows how key assumptions such as the robust nullspace property help bound the error of the approximation to a minimizer in terms of error bounds on the approximation to the objective function. This is related to robust optimization (59).

In the case that we do not know ρ or γ (the constants in the definition of wrNSPL), we can perform a log-scale grid search for suitable parameters. By increasing the width of the NNs to $\mathcal{O}(N\log(n))$, we can still gain exponential convergence in n by choosing the parameters in the grid search that lead to the vector with minimal F_3^A [the objective function of (P_3)]. In other cases, such as Theorem 4 below, it is possible to prove probabilistic results where ρ and γ are known.

Examples in Image Recovery. As an application, we consider Fourier and Walsh (binary) sampling, using Haar wavelets as a sparsifying transform. Our results can also be generalized to infinite-dimensional settings via higher-order Daubechies wavelets. Theorem 3 is quite general and there are numerous other applications where problem-dependent results similar to Theorem 4 can be shown.

Let $K = 2^r$ for $r \in \mathbb{N}$, and set $N = K^d$ so that the objective is to recover a vectorized d-dimensional tensor $c \in \mathbb{C}^N$. Let $V \in \mathbb{C}^{N\times N}$ correspond to the d-dimensional discrete Fourier or Walsh transform (*SI Appendix*). Let $\mathcal{I} \subset \{1,\dots,N\}$ be a sampling pattern with cardinality $m = |\mathcal{I}|$ and let $D = \operatorname{diag}(d_1,\dots,d_m) \in \mathbb{C}^{m\times m}$ be a suitable diagonal scaling matrix, whose entries along the diagonal depend only on $\mathcal{I}$. We assume we can observe the subsampled, scaled and noisy measurements $y = DP_{\mathcal{I}}Vc + e \in \mathbb{C}^m$, where projection $P_{\mathcal{I}}$ is treated as an $m \times N$ matrix by ignoring the zero entries.

To recover a sparse representation of c, we consider Haar wavelet coefficients. Denote the discrete d-dimensional Haar wavelet transform by $\Psi \in \mathbb{C}^{N\times N}$ and note that $\Psi^* = \Psi^{-1}$ since Ψ is unitary. To recover the wavelet coefficients $x = \Psi c$ of c, we consider the matrix $A = DP_{\mathcal{I}}V\Psi^*$ and observe that $y = Ax + e = DP_{\mathcal{I}}Vc + e$. A key result in this work is that we can design a probabilistic sampling strategy (*SI Appendix*), for both Fourier and Walsh sampling in d dimensions, requiring no more than $m \gtrsim (s_1 + \dots + s_r)\cdot\mathcal{L}$ samples, that can ensure with high probability that A satisfies the wrNSPL with certain constants. The sparsity in levels structure (*Definition 1*) is chosen to correspond to the r wavelet levels. Here $\mathcal{L}$ is a logarithmic term in N, m, s, and $\epsilon_{\mathbb{P}}^{-1}$ [where $\epsilon_{\mathbb{P}} \in (0,1)$ is a probability]. This result is crucial, as it makes A kernel aware for vectors that are approximately $(\mathbf{s},\mathbf{M})$ -sparse and allows us (using Theorem 3) to design NNs that can stably and accurately recover approximately $(\mathbf{s},\mathbf{M})$ -sparse vectors. Moreover, due to the exponential convergence in Theorem 3, the depth of these NNs depends only logarithmically on the error δ. Below follows a simplified version of our result (the full precise version is given and proved in *SI Appendix*).

Theorem 4. *Consider the above setup of recovering wavelet coefficients $x = \Psi c$ of a tensor $c \in \mathbb{C}^{K^d}$ from subsampled, scaled and noisy Fourier or Walsh measurements $y = DP_{\mathcal{I}}Vc + e$. Let $A = DP_{\mathcal{I}}V\Psi^*$, $m = |\mathcal{I}|$, and $\epsilon_{\mathbb{P}} \in (0,1)$. We then have the following*:

1) *If $\mathcal{I} \subset \{1,\dots,N\}$ is a random sampling pattern drawn according to the strategy specified in SI Appendix, and*

$$m \gtrsim (s_1 + \dots + s_r)\cdot\mathcal{L},$$

then with probability $1-\epsilon_{\mathbb{P}}$, A satisfies the wrNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $(\rho,\gamma) = (1/2,\sqrt{2})$, $w_{(l)} = \sqrt{s/s_l}$, $s = s_1 + \dots + s_r$. Here $\mathcal{L}$ denotes a term logarithmic in $\epsilon_{\mathbb{P}}^{-1}, N, m$ and s.
2) *Suppose $\mathcal{I}$ is chosen as above. For any $\delta \in (0,1)$, let $\mathcal{J}(\delta,\mathbf{s},\mathbf{M},w)$ be the set of all $y = Ax + e \in \mathbb{C}^m$ where*

$$\|x\|_{l^2} \le 1, \quad \max\left\{\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1}, \|e\|_{l^2}\right\} \le \delta. \qquad [9]$$

We provide an algorithm that constructs a neural network ϕ with $\mathcal{O}(\log(\delta^{-1}))$ hidden layers [and width bounded by $2(N+m)$] such that with probability at least $1-\epsilon_{\mathbb{P}}$,

$$\|\phi(y) - c\|_{l^2} \lesssim \delta, \quad \forall y = Ax + e \in \mathcal{J}(\delta,\mathbf{s},\mathbf{M},w).$$

Balancing the Stability and Accuracy Trade-Off

Current DL methods for image reconstruction can be unstable in the sense that 1) a tiny perturbation, in either the image or the sampling domain, can cause severe artifacts in the reconstructed image (Fig. 2, *Top* row) and/or 2) a tiny detail in the image domain might be washed out in the reconstructed image (lack of accuracy), resulting in potential false negatives. Inevitably, there is a stability–accuracy trade-off for this type of linear inverse problem, making it impossible for any reconstruction method to become arbitrarily stable without sacrificing accuracy or vice versa. Here, we show that the NNs computed by our algorithm (FIRENETs) are stable with respect to adversarial perturbations and accurate for images that are sparse in wavelets (cf. Theorem 4). As most images are sparse in wavelets, these networks also show great generalization properties to unseen images.

Adversarial Perturbations for AUTOMAP and FIRENETs. Fig. 2 (*Top* row) shows the stability test, developed in ref. 19, applied to the automated transform by manifold approximation (AUTOMAP) (60) network used for MRI reconstruction with 60% subsampling. The stability test is run on the AUTOMAP network to find a sequence of perturbations $\|e_1\|_{l^2} < \|e_2\|_{l^2} < \|e_3\|_{l^2}$. As can be seen from Fig. 2, *Top* row, the network reconstruction completely deforms the image and the reconstruction is severely unstable (similar results for other networks are demonstrated in ref. 19).

In contrast, we have applied the stability test, but now for the FIRENETs reported in this paper. Fig. 2 (*Bottom* row) shows the results for the constructed FIRENETs, where we rename the perturbations $\tilde{e}_j$ to emphasize the fact that these perturbations are sought for the FIRENETs and have nothing to do with the adversarial perturbations for AUTOMAP. We now see that despite the search for adversarial perturbations, the reconstruction remains stable. The error in the reconstruction was also found to be at most of the same order of the perturbation (as expected from the stability in Theorem 3). In applying the test to FIRENETs, we tested/tuned the parameters in the gradient ascent algorithm considerably (much more so than was needed for applying the test to AUTOMAP, where finding instabilities was straightforward) to find the worst reconstruction results, yet the reconstruction remained stable. Note also that this is just one form of stability test and it is likely that there are many other tests for creating instabilities for NNs for inverse problems. This

highlights the importance of results such as Theorem 3, which guarantees stability regardless of the perturbation.

To demonstrate the generalization properties of our NNs, we show the stability test applied to FIRENETs for a range of images in *SI Appendix*. This shows stability across different types of images and highlights that conditions such as Definition 2 allow great generalization properties.

Stabilizing Unstable NNs with FIRENETs. Our NNs also act as a stabilizer. For example, Fig. 3 shows the adversarial example for AUTOMAP (taken from Fig. 2), but now shows what happens when we take the reconstruction from AUTOMAP as an input to our FIRENETs. Here we use the fact that we can view our networks as approximations of unrolled and restarted iterative methods, allowing us to use the output of AUTOMAP as an additional input for the reconstruction. We see that FIRENETs fix the output of AUTOMAP and stabilize the reconstruction. Moreover, the full concatenation itself of the networks remains stable to adversarial attacks.

The Stability vs. Accuracy Trade-Off and False Negatives. It is easy to produce a perfectly stable network: The zero network is the obvious candidate! However, this network would obviously have poor performance and produce many false negatives. The challenge is to simultaneously ensure performance and stability. Fig. 4 highlights this issue. Here we have trained two NNs to recover a set of ellipses images from noise-free and noisy Fourier measurements. The noise-free measurements are generated as $y = Ax$, where $A \in \mathbb{C}^{m\times N}$ is a subsampled discrete Fourier transform, with $m/N = 0.15$ and $N = 1,024^2$. The noisy measurements are generated as $y = Ax + ce$, where A is as before, and the real and imaginary components of $e \in \mathbb{C}^m$ are drawn from a zero mean and unit variance normal distribution $\mathcal{N}(0, 1)$, and $c \in \mathbb{R}$ is drawn from the uniform distribution $\mathrm{Unif}([0, 100])$. The noise $ce \in \mathbb{C}^m$ is generated on the fly during the training process.

The trained networks use a standard benchmarking architecture for image reconstruction and map $y \mapsto \phi(A^*y)$, where $\phi\colon \mathbb{C}^N \to \mathbb{R}^N$ is a trainable U-net NN (8, 61). Training networks with noisy measurements, using for example this architecture, have previously been used as an example of how to create NNs that are robust toward adversarial attacks (62). As we can see from Fig. 4 (*Bottom* row) this is the case, as it does indeed create a NN that is stable with respect to worst-case perturbations. However, a key issue is that it is also producing false negatives due to its inability to reconstruct details. Similarly, as reported in the 2019 FastMRI challenge, trained NNs that performed well in terms of standard image quality metrics were prone to false negatives: They failed to reconstruct small, but physically relevant image abnormalities (25). Pathologies, generalization, and AI-generated hallucinations were subsequently a focus of the 2020 challenge (26). FIRENET, on the other hand, has a guaranteed performance (on images approximately sparse in wavelet bases) and stability, given specific conditions on the sampling procedure. The challenge is to determine the optimal balance between accuracy and stability, a well-known problem in numerical analysis.

Concluding Remarks

1) (Algorithms may not exist—Smale's 18th problem) There are well-conditioned problems where accurate NNs exist, but no algorithm can compute them. Understanding this phenomenon is essential to addressing Smale's 18th problem on the limits of AI. Moreover, limitations established in this paper suggest a classification theory describing the conditions needed for the existence of algorithms that can compute stable and accurate NNs (remark 5).
2) (Classifications and Hilbert's program) The strong optimism regarding the abilities of AI is comparable to the optimism surrounding mathematics in the early 20th century, led by D. Hilbert. Hilbert believed that mathematics could prove or disprove any statement and, moreover, that there were no restrictions on which problems could be solved by algorithms. Gödel (36) and Turing (37) turned Hilbert's optimism upside down by their foundational contributions establishing impossibility results on what mathematics and digital computers can achieve.

   Hilbert's program on the foundations of mathematics led to a rich mathematical theory and modern logic and computer science, where substantial efforts were made to classify which problems can be computed. We have sketched a similar program for modern AI, where we provide certain sufficient conditions for the existence of algorithms to produce stable and accurate NNs. We believe that such a program on the foundations of AI is necessary and will act as an invaluable catalyst for the advancement of AI.
3) (Trade-off between stability and accuracy) For inverse problems there is an intrinsic trade-off between stability and accuracy. We demonstrated NNs that offer a blend of both stability and accuracy, for the sparsity in levels class. Balancing these two interests is crucial for applications and will no doubt require a myriad of future techniques to be developed.

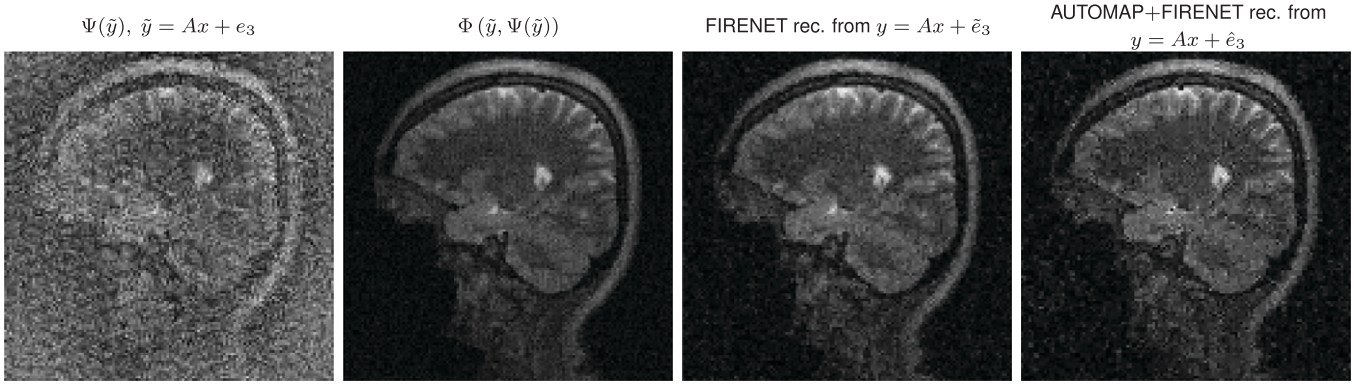

Fig. 3. Adding a few FIRENET layers at the end of AUTOMAP makes it stable. The FIRENET $\Phi\colon \mathbb{C}^m \times \mathbb{C}^N \to \mathbb{C}^N$ takes as input measurements $y \in \mathbb{C}^m$ and an initial guess for x, which we call $x_0 \in \mathbb{C}^N$. We now concatenate a 25-layer ($p = 5$, $n = 5$) FIRENET Φ and the AUTOMAP network $\Psi\colon \mathbb{C}^m \to \mathbb{C}^N$, by using the output from AUTOMAP as initial guess x_0; i.e., we consider the neural network mapping $y \mapsto \Phi(y, \Psi(y))$. In this experiment, we consider the image x from Fig. 2 and the perturbed measurements $\tilde{y} = Ax + e_3$ (here A is as in Fig. 2). *Left* shows the reconstruction of AUTOMAP from Fig. 2. *Center Left* shows the reconstruction of FIRENET with $x_0 = \Psi(\tilde{y})$. *Center Right* shows the reconstruction of FIRENET from Fig. 2. *Right* shows the reconstruction of the concatenated network with a worst-case perturbation $\hat{e}_3$ such that $\|\hat{e}_3\|_{l^2} \geq \|e_3\|_{l^2}$. In all other experiments we set $x_0 = 0$ and consider Φ as a mapping $\Phi\colon \mathbb{C}^m \to \mathbb{C}^N$.

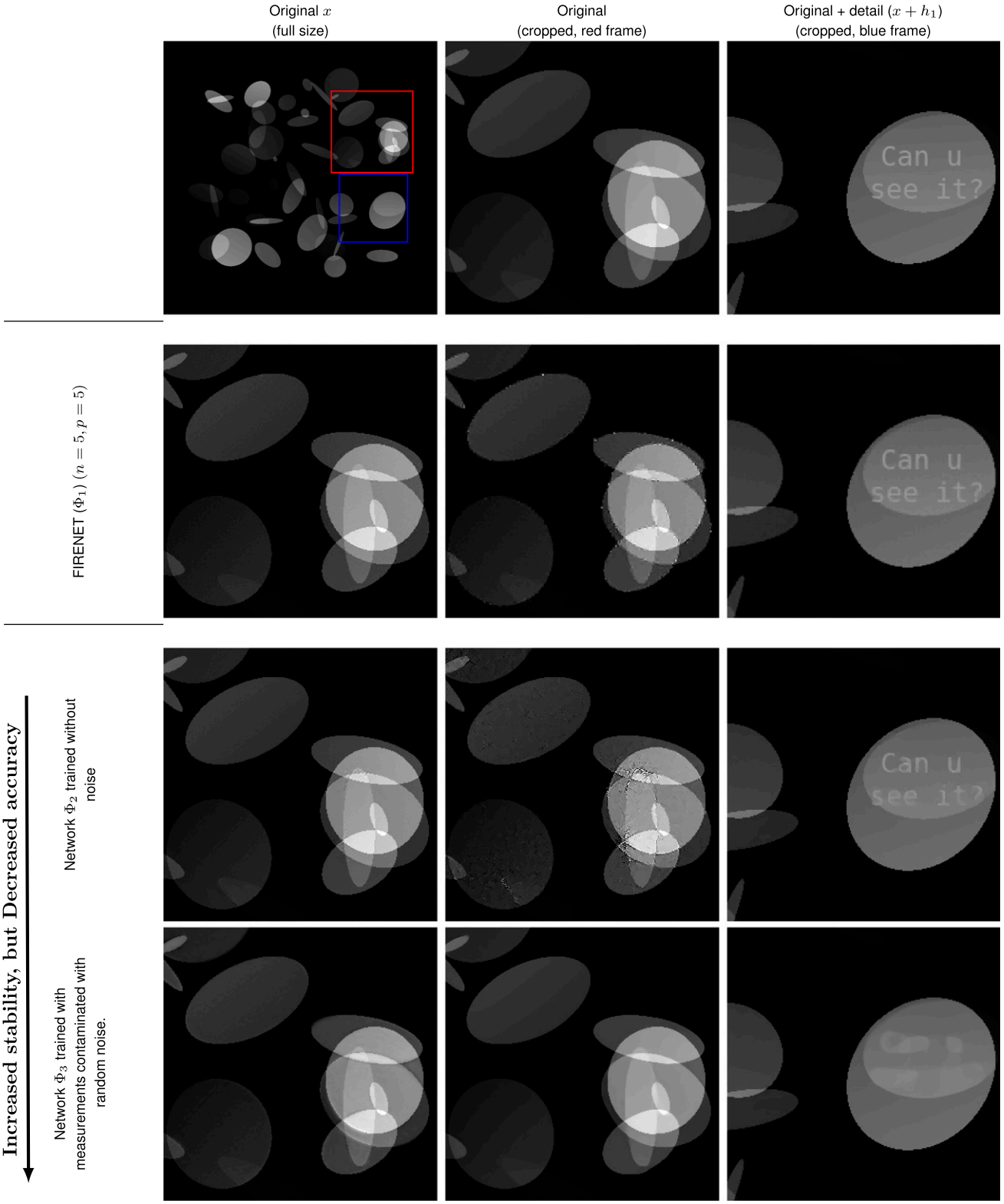

Fig. 4. Trained neural networks with limited performance can be stable. We examine the accuracy/stability trade-off for linear inverse problems by considering three reconstruction networks $\Phi_j \colon \mathbb{C}^m \to \mathbb{C}^N$, $j = 1, 2, 3$. Here Φ_1 is a FIRENET, whereas Φ_2 and Φ_3 are the U-nets mentioned in the main text, trained without and with noisy measurements, respectively. For each network, we compute a perturbation $w_j \in \mathbb{C}^N$ meant to simulate the worst-case effect, and we show a cropped version of the perturbed images $x + w_j$ in *Left* column (rows 2 to 4). In *Center* column (rows 2 to 4), we show the reconstructed images $\Phi_j(A(x + w_j))$ from each of the networks. In *Right* column (rows 2 to 4) we test the networks' ability to reconstruct a tiny detail h_1, in the form of the text "Can u see it?". As we see, the network trained on noisy measurements is stable to worst-case perturbations, but it is not accurate. Conversely, the network trained without noise is accurate but not stable. The FIRENET is balancing this trade-off and is accurate for images that are sparse in wavelets and stable to worst-case perturbations.

Tracing out the optimal stability vs. accuracy trade-off remains largely an open problem and depends on several factors such as the model class one wishes to recover, the error tolerance of the application, and the error metric used. We have shown stability and accuracy results in the l^2 norm, since it is common in the literature to measure noise via this norm. We expect a program quantifying the stability and accuracy trade-off to be of particular relevance in the increasing number of real-world implementations of machine learning in inverse problems.

4) (Inverse problems vs. classification problems) The mathematical techniques used in this paper are applied to inverse problems. However, the mathematical framework of ref. 33 can be used to produce similar impossibility results for computing NNs in classification problems (63).
5) (Future work—Which NNs can be computed?) There is an enormous literature (29, 30, 64–66) on the existence of NNs with great approximation qualities. However, Theorem 2 shows that only certain accuracy may be computationally achievable. Our results are just the beginning of a mathematical theory studying which NNs can be computed by algorithms. This opens up for a theory covering other sufficient (and potentially necessary) conditions guaranteeing stability and accuracy and extensions to other inverse problems such as phase retrieval (67, 68). One can also prove similar computational barriers in other settings via the tools developed in this paper.

Methods: The Solvability Complexity Index Hierarchy

Our proof techniques for fundamental barriers in Theorem 2 stem from the mathematics behind the solvability complexity index (SCI) hierarchy (33, 69–78). The SCI hierarchy generalizes the fundamental problems of Smale (79, 80) on existence of algorithms and work by McMullen (81) and Doyle and McMullen (82). We extend and refine these techniques, in particular those of ref. 33, and generalize the mathematics behind the extended Smale's ninth problem (33, 34)—which also builds on the SCI hierarchy. More precisely, to prove our results we develop the concept of sequential general algorithms. General algorithms are a key tool in the mathematics of the SCI hierarchy. Sequential general algorithms extend this concept and capture the notion of adaptive and/or probabilistic choice of training data. The architectures of the NNs in Theorem 3 are based on unrolled primal–dual iterations for (P_3). In addition to providing stability, the wrNSPL allows us to prove exponential convergence through a careful restarting and reweighting scheme. Full theoretical derivations are given in *SI Appendix*.

Data Availability. All the code and data used to produce the figures in this paper are available from GitHub, https://www.github.com/Comp-Foundations-and-Barriers-of-AI/firenet.

ACKNOWLEDGMENTS. We are grateful to Kristian Haug for allowing us to use parts of code for unrolled primal–dual algorithms in TensorFlow. This work was supported by a Research Fellowship at Trinity College, Cambridge University (to M.J.C.), as well as by a Leverhulme Prize and a Royal Society University Research Fellowship (to A.C.H.).

1. A. Krizhevsky, I. Sutskever, G. E. Hinton, "Imagenet classification with deep convolutional neural networks " in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger, Eds. (Curran Associates, Inc., 2012), pp. 1097–1105.
2. K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 770–778.
3. R. Girshick, J. Donahue, T. Darrell, J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2014), pp. 580–587.
4. G. Hinton *et al.*, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97 (2012).
5. J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, V. Svetnik, Deep neural nets as a method for quantitative structure-activity relationships. *J. Chem. Inf. Model.* **55**, 263–274 (2015).
6. G. E. Dahl, D. Yu, L. Deng, A. Acero, Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Trans. Audio Speech Lang. Process.* **20**, 30–42 (2011).
7. U. S. Kamilov *et al.*, Learning approach to optical tomography. *Optica* **2**, 517–522 (2015).
8. K. H. Jin, M. T. McCann, E. Froustey, M. Unser, Deep convolutional neural network for inverse problems in imaging. *IEEE Trans. Image Process.* **26**, 4509–4522 (2017).
9. M. T. McCann, K. H. Jin, M. Unser, Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Process. Mag.* **34**, 85–95 (2017).
10. K. Hammernik *et al.*, Learning a variational network for reconstruction of accelerated MRI data. *Magn. Reson. Med.* **79**, 3055–3071 (2018).
11. S. Arridge, P. Maass, O. Öktem, C. B. Schönlieb, Solving inverse problems using data-driven models. *Acta Numer.* **28**, 1–174 (2019).
12. G. Ongie *et al.*, Deep learning techniques for inverse problems in imaging. *IEEE J. Sel. Areas Inf. Theory* **1**, 39–56 (2020).
13. C. Szegedy *et al.*, "Intriguing properties of neural networks" in *International Conference on Learning Representations* (2014). https://openreview.net/forum?id=kklr_MTHMRQjG. (Accessed 3 March 2022).
14. S. M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 2574–2582.
15. S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, "Universal adversarial perturbations" in *IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2017), pp. 86–94.
16. N. Akhtar, A. Mian, Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* **6**, 14410–14430 (2018).
17. N. Carlini, D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text" in *2018 IEEE Security and Privacy Workshops (SPW)* (IEEE, 2018), pp. 1–7.
18. Y. Huang *et al.*, "Some investigations on robustness of deep learning in limited angle tomography" in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, A. F. Frangi, J. A. Schnabel, C. Davatzikos, C. Alberola-López, G. Fichtinger, Eds. (Springer, 2018), pp. 145–153.
19. V. Antun, F. Renna, C. Poon, B. Adcock, A. C. Hansen, On instabilities of deep learning in image reconstruction and the potential costs of AI. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 30088–30095 (2020).
20. S. G. Finlayson *et al.*, Adversarial attacks on medical machine learning. *Science* **363**, 1287–1289 (2019).
21. I. Y. Tyukin, D. J. Higham, A. N. Gorban, "On adversarial examples and stealth attacks in artificial intelligence systems" in *2020 International Joint Conference on Neural Networks* (IEEE, 2020), pp. 1–6.
22. N. M. Gottschling, V. Antun, B. Adcock, A. C. Hansen, The troublesome kernel: Why deep learning for inverse problems is typically unstable. arXiv [Preprint] (2020). https://arxiv.org/abs/2001.01258 (Accessed 5 January 2020).
23. N. Baker *et al.*, "Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence" (Tech. Rep. 1478744, USDOE Office of Science, 2019).
24. R. Hamon, H. Junklewitz, I. Sanchez, *Robustness and Explainability of Artificial Intelligence - From Technical to Policy Solutions* (Publications Office of the European Union, 2020).
25. F. Knoll *et al.*, Advancing machine learning for MR image reconstruction with an open competition: Overview of the 2019 fastMRI challenge. *Magn. Reson. Med.* **84**, 3054–3070 (2020).
26. M. J. Muckley *et al.*, Results of the 2020 fastMRI Challenge for Machine Learning MR Image Reconstruction. *IEEE Trans. Med. Imaging* **40**, 2306–2317 (2021).
27. C. Belthangady, L. A. Royer, Applications, promises, and pitfalls of deep learning for fluorescence image reconstruction. *Nat. Methods* **16**, 1215–1225 (2019).
28. D. P. Hoffman, I. Slavitt, C. A. Fitzpatrick, The promise and peril of deep learning in microscopy. *Nat. Methods* **18**, 131–132 (2021).
29. A. Pinkus, Approximation theory of the MLP model in neural networks. *Acta Numer.* **8**, 143–195 (1999).
30. R. DeVore, B. Hanin, G. Petrova, Neural network approximation. *Acta Numer.* **30**, 327–444 (2021).
31. B. Adcock, N. Dexter, The gap between theory and practice in function approximation with deep neural networks. *SIAM J. Math. Data Sci.* **3**, 624–655 (2021).
32. C. F. Higham, D. J. Higham, Deep learning: An introduction for applied mathematicians. *SIAM Rev.* **61**, 860–891 (2019).
33. A. Bastounis, A. C. Hansen, V. Vlačić, The extended Smale's 9th problem – On computational barriers and paradoxes in estimation, regularisation, computer-assisted proofs and learning. arXiv [Preprint] (2021). https://arxiv.org/abs/2110.15734 (Accessed 29 October 2021).
34. S. Smale, Mathematical problems for the next century. *Math. Intell.* **20**, 7–15 (1998).
35. A. Turing, I.-Computing machinery and intelligence. *Mind* **LIX**, 433–460 (1950).
36. K Gödel, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatsh. Math. Phys.* **38**, 173–198 (1931).
37. A. Turing, On computable numbers, with an application to the Entscheidungsproblem. *Proc. Lond. Math. Soc.* **42**, 230–265 (1936).
38. S. Weinberger, *Computers, Rigidity, and Moduli: The Large-Scale Fractal Geometry of Riemannian Moduli Space* (Princeton University Press, Princeton, NJ, 2004).
39. P. Niyogi, S. Smale, S. Weinberger, A topological view of unsupervised learning from noisy data. *SIAM J. Comput.* **40**, 646–663 (2011).
40. X. Chen, J. Liu, Z. Wang, W. Yin, "Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Q. Weinberger, Eds. (Curran Associates, Inc., 2018), pp. 9061–9071.

41. A. Raj, Y. Bresler, B. Li, "Improving robustness of deep-learning-based image reconstruction" in *International Conference on Machine Learning*, H. Daumé III, A. Singh, Eds. (PMLR, 2020), pp. 7932–7942.
42. I. Goodfellow *et al.*, "Generative adversarial nets" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Q. Weinberger, Eds. (Curran Associates, Inc., 2014), pp. 2672–2680.
43. M. Arjovsky, S. Chintala, L. Bottou, "Wasserstein generative adversarial networks" in *International Conference on Machine Learning*, D. Precup, Y. W. Teh, Eds. (PMLR, 2017), vol. 70, pp. 214–223.
44. B. Adcock, A. C. Hansen, C. Poon, B. Roman, "Breaking the coherence barrier: A new theory for compressed sensing" in *Forum of Mathematics, Sigma* (Cambridge University Press, 2017), vol. 5.
45. A. Bastounis, A. C. Hansen, On the absence of uniform recovery in many real-world applications of compressed sensing and the restricted isometry property and nullspace property in levels. *SIAM J. Imaging Sci.* **10**, 335–371 (2017).
46. B. Adcock, A. C. Hansen, *Compressive Imaging: Structure, Sampling, Learning* (Cambridge University Press, 2021).
47. C. Boyer, J. Bigot, P. Weiss, Compressed sensing with structured sparsity and structured acquisition. *Appl. Comput. Harmon. Anal.* **46**, 312–350 (2019).
48. E. J. Candès, J. Romberg, T. Tao, Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theory* **52**, 489–509 (2006).
49. E. J. Candes, J. K. Romberg, T. Tao, Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pure Appl. Math.* **59**, 1207–1223 (2006).
50. D. L. Donoho, Compressed sensing. *IEEE Trans. Inf. Theory* **52**, 1289–1306 (2006).
51. D. L. Donoho, J. Tanner, Counting faces of randomly-projected polytopes when the projection radically lowers dimension. *J. Am. Math. Soc.* **22**, 1–53 (2009).
52. A. Cohen, W. Dahmen, R. DeVore, Compressed sensing and best k-term approximation. *J. Am. Math. Soc.* **22**, 211–231 (2009).
53. A. Jones, A. Tamtögl, I. Calvo-Almazán, A. Hansen, Continuous compressed sensing for surface dynamical processes with helium atom scattering. *Sci. Rep.* **6**, 27776 (2016).
54. R. A. DeVore, Nonlinear approximation. *Acta Numer.* **7**, 51–150 (1998).
55. M. P. Friedlander, H. Mansour, R. Saab, Ö. Yilmaz, Recovering compressively sampled signals using partial support information. *IEEE Trans. Inf. Theory* **58**, 1122–1134 (2012).
56. V. Monga, Y. Li, Y. C. Eldar, Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Process. Mag.* **38**, 18–44 (2021).
57. A Ben-Tal, A Nemirovski, Lectures on modern convex optimization (2020–2021). https://www2.isye.gatech.edu/ñemirovs/LMCO_LN.pdf. (Accessed 5 February 2022).
58. Y. E. Nesterov, A. Nemirovski, On first-order algorithms for l1/nuclear norm minimization. *Acta Numer.* **22**, 509–575 (2013).
59. A. Ben-Tal, L. El Ghaoui, A. Nemirovski, *Robust Optimization* (Princeton Series in Applied Mathematics, Princeton University Press, 2009).
60. B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, M. S. Rosen, Image reconstruction by domain-transform manifold learning. *Nature* **555**, 487–492 (2018).
61. J. Long, E. Shelhamer, T. Darrell, "Fully convolutional networks for semantic segmentation" in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE, 2015), pp. 3431–3440.
62. M. Genzel, J. Macdonald, M. März, "Solving inverse problems with deep neural networks–Robustness included?" in *Transactions on Pattern Analysis and Machine Intelligence*, 10.1109/TPAMI.2022.3148324 (2022).
63. A. Bastounis, A. C. Hansen, V. Vlacic, The mathematics of adversarial attacks in AI – Why deep learning is unstable despite the existence of stable neural networks. arXiv [Preprint] (2021). https://arxiv.org/abs/2109.06098 (Accessed 13 September 2021).
64. H. Bölcskei, P. Grohs, G. Kutyniok, P. Petersen, Optimal approximation with sparsely connected deep neural networks. *SIAM J. Math. Data Sci.* **1**, 8–45 (2019).
65. I. Daubechies, R. DeVore, S. Foucart, B. Hanin, G. Petrova, Nonlinear approximation and (deep) ReLU networks. *Constr. Approx.* **55**, 127–172 (2021).
66. W. E. S. Wojtowytsch, On the Banach spaces associated with multi-layer ReLU networks: Function representation, approximation theory and gradient descent dynamics. *CSIAM Transact. Appl. Math* **1**, 387–440 (2020).
67. E. J. Candes, T. Strohmer, V. Voroninski, Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *Commun. Pure Appl. Math.* **66**, 1241–1274 (2013).
68. A. Fannjiang, T. Strohmer, The numerics of phase retrieval. *Acta Numer.* **29**, 125–228 (2020).
69. A. C. Hansen, On the solvability complexity index, the n-pseudospectrum and approximations of spectra of operators. *J. Am. Math. Soc.* **24**, 81–124 (2011).
70. A. C. Hansen, O. Nevanlinna, Complexity issues in computing spectra, pseudospectra and resolvents. *Banach Cent. Publ.* **112**, 171–194 (2016).
71. J. Ben-Artzi, M. J. Colbrook, A. C. Hansen, O. Nevanlinna, M. Seidel, Computing spectra – On the solvability complexity index hierarchy and towers of algorithms. arXiv [Preprint] (2020). https://arxiv.org/abs/1508.03280 (Accessed 15 June 2020).
72. J. Ben-Artzi, A. C. Hansen, O. Nevanlinna, M. Seidel, New barriers in complexity theory: On the solvability complexity index and the towers of algorithms. *C. R. Math.* **353**, 931–936 (2015).
73. M. Colbrook, "The foundations of infinite-dimensional spectral computations," PhD thesis, University of Cambridge, Cambridge, UK (2020).
74. M. J. Colbrook, Computing spectral measures and spectral types. *Commun. Math. Phys.* **384**, 433–501 (2021).
75. M. Colbrook, A. Horning, A. Townsend, Computing spectral measures of self-adjoint operators. *SIAM Rev.* **63**, 489–524 (2021).
76. J. Ben-Artzi, M. Marletta, F. Rösler, Computing the sound of the sea in a seashell. *Found. Comput. Math.*, 10.1007/s10208-021-09509-9 (2021).
77. M. J. Colbrook, A. C. Hansen, The foundations of spectral computations via the solvability complexity index hierarchy. arXiv [Preprint] (2021). https://arxiv.org/abs/1908.09592 (Accessed 6 August 2020).
78. H. Boche, V. Pohl, "The solvability complexity index of sampling-based Hilbert transform approximations" in 2019 13th International Conference on Sampling Theory and Applications (SampTA) (IEEE, 2019), pp. 1–4.
79. S. Smale, The fundamental theorem of algebra and complexity theory. *Am. Math. Soc. Bull.* **4**, 1–36 (1981).
80. S. Smale, Complexity theory and numerical analysis. *Acta Numer.* **6**, 523–551 (1997).
81. C. McMullen, Families of rational maps and iterative root-finding algorithms. *Ann. Math.* **125**, 467–493 (1987).
82. P. Doyle, C. McMullen, Solving the quintic by iteration. *Acta Math.* **163**, 151–180 (1989).
83. R. Strack, Imaging: AI transforms image reconstruction. *Nat. Methods* **15**, 309 (2018).
84. Q. Fan *et al.*, MGH-USC Human Connectome Project datasets with ultra-high b-value diffusion MRI. *Neuroimage* **124** (Pt. B), 1108–1114 (2016).

PNAS
www.pnas.org

Supplementary Information for

The difficulty of computing stable and accurate neural networks: On the barriers of deep learning and Smale's 18th problem

Matthew J. Colbrook, Vegard Antun and Anders C. Hansen

E-mail: m.colbrook@damtp.cam.ac.uk, vegarant@math.uio.no, a.hansen@damtp.cam.ac.uk

This PDF file includes:

Supplementary text including theorems and their proofs.
Experimental details and further numerical experiments in addition to those in the main text.
Figs. S1 to S4
SI References

Here we provide statements of theorems from the main text, proofs of theorems, detailed explanations of the experimental setup and further numerical examples. We briefly collect some basic notation, and further notation will be introduced throughout where appropriate. We use $\mathcal{N}_{m,N}$ to denote the class of neural networks (NNs) from $\mathbb{C}^m$ to $\mathbb{C}^N$ (see §1.B.1 for the precise definition). Given a metric space $(\mathcal{M}, d)$, $x \in \mathcal{M}$ and $X \subset \mathcal{M}$, $d(x, X) = \text{dist}(x, X) = \inf_{y \in X} d(x, y)$. For a matrix $A \in \mathbb{C}^{m \times N}$, the norm $\|A\|$ refers to the operator norm of A when $\mathbb{C}^m$ and $\mathbb{C}^N$ are equipped with the standard l^2-norm. For $x \in \mathbb{C}^N$ and $p \in [1, \infty]$, $\|x\|_{l^p}$ refers to the l^p-norm of x. For a set of indices S and vector x, x_S is the vector defined by $(x_S)_j = x_j$ if $j \in S$ and $(x_S)_j = 0$ if $j \notin S$. Complex rationals $\mathbb{Q} + i\mathbb{Q}$ are denoted by $\mathbb{Q}[i]$. We use $\square$ to denote the end of a proof and $\boxtimes$ to denote the end of a remark.

Contents

1. Statement of Theorems and Results

We now state our main theorems. Proofs are given in §3 – §5. Recall that we study the canonical inverse problem of solving an underdetermined system of linear equations:

$$\text{Given noisy measurements } y = Ax + e \in \mathbb{C}^m \text{ of } x \in \mathbb{C}^N, \text{ recover } x. \tag{1.1}$$

Here $A \in \mathbb{C}^{m \times N}$ represents a model of typically undersampled sampling ($m < N$), such as a subsampled discrete Fourier transform as in Magnetic Resonance Imaging (MRI). Specific choices of A are discussed in §1.C. Problem (1.1) forms the basis for much of inverse problems and image analysis. The possibility of $y \neq Ax$ models noise or perturbations.

A. Existence of NNs is not enough, algorithms may not compute them sufficiently accurately. Here we present Theorems 1 and 2 of the main paper. Given a matrix $A \in \mathbb{C}^{m \times N}$ and a vector $y \in \mathbb{C}^m$, recall that we consider the following three minimisation problems:

$$(P_1) \quad \text{argmin}_{x \in \mathbb{C}^N} F_1^A(x) \coloneqq \|x\|_{l^1_w}, \text{ such that } \|Ax - y\|_{l^2} \leq \epsilon, \tag{1.2}$$

$$(P_2) \quad \text{argmin}_{x \in \mathbb{C}^N} F_2^A(x, y, \lambda) \coloneqq \lambda \|x\|_{l^1_w} + \|Ax - y\|^2_{l^2}, \tag{1.3}$$

$$(P_3) \quad \text{argmin}_{x \in \mathbb{C}^N} F_3^A(x, y, \lambda) \coloneqq \lambda \|x\|_{l^1_w} + \|Ax - y\|_{l^2}. \tag{1.4}$$

The parameters λ and ϵ are positive rational numbers, and the weighted l^1_w norm is given by $\|x\|_{l^1_w} \coloneqq \sum_{l=1}^N w_l |x_l|$, where each weight w_l is a positive rational. Throughout, we use the following notation:

$$\Xi(A, y) \text{ is the set of minimisers for } (P_j) \text{ given input } A \in \mathbb{C}^{m \times N}, y \in \mathbb{C}^m, \tag{1.5}$$

where, for notational convenience, we have suppressed the dependence on ϵ or λ (which are usually fixed parameters) and the index j. In certain cases, we will write Ξ_j to specify minimisers of problem (P_j). Let

$$A \in \mathbb{C}^{m\times N}, \quad \mathcal{S} = \{y_k\}_{k=1}^R \subset \mathbb{C}^m, \quad R < \infty.$$

In the main text we considered the following key question:

> *Given a collection Ω of such pairs $(A, \mathcal{S})$, does there exists a neural network approximating the mapping Ξ, and if so, can such an approximation be trained by an algorithm?*

To make this question precise, we first note that A and the elements in $\mathcal{S}$ will typically never be exact, but can be approximated to arbitrary precision. For example, this would be the case if A was a subsampled discrete cosine transform. Thus, we can access approximations $\{y_{k,n}\}_{k=1}^R \subset \mathbb{Q}[i]^m$ and $A_n \in \mathbb{Q}[i]^{m\times N}$ such that

$$\|y_{k,n} - y_k\| \leq 2^{-n}, \quad \|A_n - A\| \leq 2^{-n}, \quad \forall n \in \mathbb{N}. \tag{1.6}$$

We also assume access to $\{x_{k,n}\}_{k=1}^R \subset \mathbb{Q}[i]^N$ such that

$$\inf_{x^*\in\Xi(A_n,y_{k,n})} \|x_{k,n} - x^*\| \leq 2^{-n}, \quad \forall n \in \mathbb{N}. \tag{1.7}$$

Hence, the training set associated with $(A, \mathcal{S}) \in \Omega$ for training a suitable NN must be

$$\iota_{A,\mathcal{S}} \coloneqq \{(y_{k,n}, A_n, x_{k,n}) \,|\, k = 1, \dots, R, \text{ and } n \in \mathbb{N}\}. \tag{1.8}$$

Thus, given a collection of $(A, \mathcal{S})$, we denote the class of all such admissible training data by

$$\Omega_{\mathcal{T}} \coloneqq \{\iota_{A,\mathcal{S}} \text{ as in Eq. (1.8)} \,|\, (A, \mathcal{S}) \in \Omega, \text{ Eq. (1.6) and Eq. (1.7) hold}\}.$$

Precise statements addressing the above question are summarised in the following theorems, the first of which follows directly from standard universal approximation theorems.

Theorem 1 (**Neural networks exist for Ξ**). *Consider the problem (P_j) $(j = 1, 2, 3)$ for fixed dimensions $m < N$ and parameters λ or ϵ. Then, for any family Ω of such $(A, \mathcal{S})$ described above, there exists a mapping*

$$\mathcal{K}\colon \Omega_{\mathcal{T}} \to \mathcal{N}_{m,N}, \quad \mathcal{K}(\iota_{A,\mathcal{S}}) = \varphi_{A,\mathcal{S}}, \quad \text{such that} \quad \varphi_{A,\mathcal{S}}(y) \in \Xi(A, y), \quad \forall y \in \mathcal{S}.$$

In words, $\mathcal{K}$ maps the training data $\Omega_{\mathcal{T}}$ to NNs that solve the optimisation problem (P_j) for each $(A, \mathcal{S}) \in \Omega$.

Despite the existence of NNs guaranteed by Theorem 1, the problem of computing such a NN from training data is a most delicate issue, as described in the following theorem (proven in §3).

Theorem 2 (**Despite existence, neural networks may only be computed to a certain accuracy**). *For $j = 1, 2$ or 3, consider the optimisation problem (P_j) for fixed parameters $\lambda \in (0, 1]$ or $\epsilon \in (0, 1/2]$ and $w_l = 1$, where $N \geq 2$ and $m < N$. Let $K > 2$ be a positive integer and let $L \in \mathbb{N}$. Then there exists a class Ω of elements $(A, \mathcal{S})$ as in* Eq. (1.5)*, with the following properties. The class Ω is well-conditioned with condition numbers of the matrices AA^* and the solution maps Ξ, as well as the feasibility primal local condition number (see §3.A), all bounded by 1 independent of all parameters. However, the following hold:*

(i) There does not exist any algorithm that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$, produces a NN $\phi_{A,\mathcal{S}}$ with

$$\min_{y\in\mathcal{S}} \inf_{x^*\in\Xi(A,y)} \|\phi_{A,\mathcal{S}}(y) - x^*\|_{l^2} \leq 10^{-K}, \quad \forall\, (A, \mathcal{S}) \in \Omega. \tag{1.9}$$

Furthermore, for any $p > 1/2$, no probabilistic algorithm (BSS, Turing or any model of computation) can produce a NN $\phi_{A,\mathcal{S}}$ such that Eq. (1.9) *holds with probability at least p.*

(ii) There does exist a deterministic Turing machine that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$, produces a NN $\phi_{A,\mathcal{S}}$ with

$$\max_{y\in\mathcal{S}} \inf_{x^*\in\Xi(A,y)} \|\phi_{A,\mathcal{S}}(y) - x^*\|_{l^2} \leq 10^{-(K-1)}, \quad \forall\, (A, \mathcal{S}) \in \Omega. \tag{1.10}$$

However, for any probabilistic Turing machine $(\Gamma, \mathbb{P})$, $M \in \mathbb{N}$ and $p \in \left[0, \frac{N-m}{N+1-m}\right)$ that produces a NN $\phi_{A,\mathcal{S}}$, there exists a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$ such that for all $y \in \mathcal{S}$,

$$\mathbb{P}\Big(\inf_{x^*\in\Xi(A,y)} \|\phi_{A,\mathcal{S}}(y) - x^*\|_{l^2} > 10^{1-K} \text{ or the training data size needed to construct } \phi_{A,\mathcal{S}} > M\Big) > p. \tag{1.11}$$

(iii) There does exist a deterministic Turing machine that, given a training set $\iota_{A,\mathcal{S}} \in \Omega_{\mathcal{T}}$ and using only L training data from each $\iota_{A,\mathcal{S}}$, produces a NN $\phi_{A,\mathcal{S}}(y)$ such that

$$\max_{y\in\mathcal{S}} \inf_{x^*\in\Xi(A,y)} \|\phi_{A,\mathcal{S}}(y) - x^*\|_{l^2} \leq 10^{-(K-2)}, \quad \forall\, (A, \mathcal{S}) \in \Omega. \tag{1.12}$$

Remark 1.1 (Meaning of the notation $(\Gamma, \mathbb{P})$)**.** The notation $(\Gamma, \mathbb{P})$ in (ii) is used to denote a (possibly) randomised algorithm Γ and its law $\mathbb{P}$. This includes scenarios such as stochastic gradient descent, random selection of training data, random computation with training data etc. The precise setup is detailed in §3.A. ⊠

Remark 1.2 (Generalisations of Theorem 2)**.** For simplicity, we have stated Theorem 2 for errors measured in the l^2-norm and the case of unweighted l^1 regularisation (all the $w_l = 1$) in the problems (P_j). However, the proof can be adapted, and similar results hold for any norm replacing the l^2-norm, and any non-singular weighted l^1 regularisation. Moreover, result (i) in Theorem 2 holds regardless of the model of computation, even if we allowed real number arithmetic (see Definition 3.3). For further details on the precise setup, including the definition of condition numbers, which are standard in the literature, see §3.A. Finally, the theorem remains true if we restrict ourselves to real-valued matrices and vectors. ⊠

Further details on the experiment following this theorem (that was given in the main text) can be found in §3.D.

B. Computing stable and accurate neural networks.

B.1. Neural networks and notational conventions. To state our theorems, we need to be precise about the definition of a NN. For introductions to the field of DL and NNs, we refer the reader to (1, 2) and (3), respectively, and the references therein. To capture standard architectures used in practice such as skip connections, we consider the following definition of a NN. Without loss of generality and for ease of exposition, we also work with complex-valued NNs. Such networks can be realised by real-valued NNs by splitting into real and imaginary parts. A NN is a mapping $\phi\colon \mathbb{C}^m \to \mathbb{C}^N$ that can be written as a composition

$$\phi(y) = V_T(\rho_{T-1}(...\rho_1(V_1(y)))), \quad \text{where:}$$

- Each V_j is an affine map $\mathbb{C}^{N_{j-1}} \to \mathbb{C}^{N_j}$ given by $V_j(x) = W_j x + b_j(y)$ where $W_j \in \mathbb{C}^{N_j \times N_{j-1}}$ and the $b_j(y) = R_j y + c_j \in \mathbb{C}^{N_j}$ are affine functions of the input y.
- Each $\rho_j\colon \mathbb{C}^{N_j} \to \mathbb{C}^{N_j}$ is one of two forms:

  (i) There exists an index set $I_j \subset \{1, ..., N_j\}$ (possibly a strict subset) such that ρ_j applies a possibly non-linear function $f_j : \mathbb{C} \to \mathbb{C}$ element-wise on the input vector's components with indices in I_j:

$$\rho_j(x)_k = \begin{cases} f_j(x_k), & \text{if } k \in I_j \\ x_k, & \text{otherwise.} \end{cases}$$

  (ii) There exists a possibly non-linear function $f_j : \mathbb{C} \to \mathbb{C}$ such that, after decomposing the input vector x as $(x_0, X^\top, Y^\top)^\top$ ($\top$ denotes transpose) for scalar x_0 and $X \in \mathbb{C}^{m_j}$ ($Y \in \mathbb{C}^{N_j - 1 - m_j}$), we have

$$\rho_j : \begin{pmatrix} x_0 \\ X \\ Y \end{pmatrix} \to \begin{pmatrix} 0 \\ f_j(x_0)X \\ Y \end{pmatrix}. \tag{1.13}$$

The affine dependence of $b_j(y)$ on y allows skip connections from the input to the current level as in standard definitions of feed-forward NNs (4, p. 269), and the above architecture has become standard (5–7).

Remark 1.3 (On the use of multiplication)**.** The use of non-linear functions of the form (ii) may be re-expressed using the following element-wise squaring trick:

$$\begin{pmatrix} x_0 \\ X \\ Y \end{pmatrix} \to \begin{pmatrix} f_j(x_0) \\ X \\ Y \end{pmatrix} \to \begin{pmatrix} f_j(x_0)\mathbf{1} \\ X \\ f_j(x_0)\mathbf{1} + X \\ Y \end{pmatrix} \to \begin{pmatrix} f_j(x_0)^2\mathbf{1} \\ X^2 \\ [f_j(x_0)\mathbf{1} + X]^2 \\ Y \end{pmatrix} \to \begin{pmatrix} 0 \\ \frac{1}{2}\left[[f_j(x_0)\mathbf{1} + X]^2 - f_j(x_0)^2\mathbf{1} - X^2\right] \\ Y \end{pmatrix},$$

where $\mathbf{1}$ denotes a vector of ones of the same size as X (so that $f_j(x_0) \to f_j(x_0)\mathbf{1}$ is a linear map). However, this is not done in practice since the map in Eq. (1.13) is directly trainable via backpropagation. ⊠

Note that we do not allow the matrices W_j to depend on y. We denote the collection of all NNs of the above form by $\mathcal{N}_{\mathbf{D},T,q}$, where the vector $\mathbf{D} = (N_0 = m, N_1, ..., N_T = N)$ denotes the dimensions in each layer, T denotes the number of layers and q denotes the number of different non-linear functions applied (including the count of different I_j and m_j). In general, we will require that the layer sizes N_j do not grow with j so that the size of each layer is of the same order as the sampling matrix A.

We consider stable reconstruction from noisy undersampled measurements, as in Eq. (1.1), and NNs that can be constructed via algorithms. To make this precise, we assume that we have access to a sequence of matrices $A_l \in \mathbb{Q}[i]$ such that $\|A - A_l\| \leq q_l$ for some known null sequence $\{q_l\}$. This is consistent with the training set given by Eq. (1.8). To construct NNs via an algorithm, care must be taken with the non-linear activation functions. We assume that for $\theta \in \mathbb{Q}_{>0}$ we have access to a routine "sqrt_θ" such that $|\text{sqrt}_\theta(x) - \sqrt{x}| \leq \theta$ for all $x \in \mathbb{R}_{\geq 0}$. In what follows, the non-linear maps f_j used in the NNs are either arithmetic or constructed using arithmetic operations and sqrt_θ. We always ensure that sqrt_θ acts on non-negative real numbers and on rational inputs if the input to the NN is rational. We refer to the pair (ϕ, θ) as a NN.

Remark 1.4 (Approximating $\sqrt{\cdot}$ with neural networks)**.** On any bounded set (for bounded input our constructed NNs only require the routine sqrt_θ on a bounded set), we can construct an approximation to $\sqrt{\cdot}$ using standard non-linear activation functions such as ReLU (more efficient approximations may be achieved by using other activation functions such as rational maps (8)). The choice of the square root function is somewhat arbitrary, but simplifies our proof of Theorem 3. Similar results hold for other activation functions. ⊠

Remark 1.5 (An interpretation of θ)**.** As well as being necessary from a foundations point of view, an important interpretation of θ is numerical stability, or accumulation of errors, of the forward pass of the NN. Larger values of θ show greater stability when applying the NN in finite precision. We prove results of the form

$$\|\phi_n(y) - x\|_{l^2} \leq \varepsilon + c_1(A,x)\|Ax - y\|_{l^2} + c_2(A,x)\upsilon^n, \quad \forall x \in S \subset \mathbb{C}^N, y \in \mathbb{C}^m, \tag{1.14}$$

where (ϕ_n, θ_n) is a (sequence of) NN(s) with $\mathcal{O}(n)$ layers that is computed by an algorithm, $\upsilon \in (0,1)$ describes the exponential rate of convergence in the number of layers, $\varepsilon > 0$ (in our results ε will be related to the distance to vectors that are sparse in levels: $\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}$ in Definition 1.6), and $\theta_n^{-1} = \theta^{-1}$ is bounded independent of n. Up to the error tolerance ε, the constant $c_1(A,x)$ can be thought of as an asymptotic *local Lipschitz constant* for the NNs as $n \to \infty$, and thus measures *stability of inexact input* y. In practice one would use floating point arithmetic to approximate square roots. Hence, the boundedness of θ_n^{-1} is a *numerical notion of stability* - the accuracy needed for approximating square roots (and the non-linear maps) does not become too great and errors do not accumulate as n increases. Moreover, in practice the value of θ^{-1} needed is well below what is achieved using standard floating-point formats. ⊠

B.2. The construction of stable and accurate neural networks. The main result of this subsection, Theorem 3, uses the concept of sparsity in levels and weighted robust null space property in levels defined in the main text. We repeat these definitions here for the convenience of the reader.

Definition 1.6 (Sparsity in levels)**.** *Let* $\mathbf{M} = (M_1, ..., M_r) \in \mathbb{N}^r$, $1 \leq M_1 < ... < M_r = N$, *and* $\mathbf{s} = (s_1, ..., s_r) \in \mathbb{N}_0^r$, *where* $s_l \leq M_l - M_{l-1}$ *for* $l = 1, ..., r$ *(*$M_0 = 0$*).* $x \in \mathbb{C}^N$ *is* $(\mathbf{s}, \mathbf{M})$*-sparse in levels if*

$$|\text{supp}(x) \cap \{M_{l-1} + 1, ..., M_l\}| \leq s_l, \quad l = 1, ..., r.$$

The total sparsity is $s = s_1 + ... + s_r$. *We denote the set of* $(\mathbf{s}, \mathbf{M})$*-sparse vectors by* $\Sigma_{\mathbf{s},\mathbf{M}}$. *We also define the following measure of distance of a vector* x *to* $\Sigma_{\mathbf{s},\mathbf{M}}$ *by*

$$\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w} = \inf\{\|x - z\|_{l^1_w} : z \in \Sigma_{\mathbf{s},\mathbf{M}}\}.$$

For simplicity, we assume throughout that each $s_l > 0$ and that

$$w_i = w_{(l)}, \quad \text{if } M_{l-1} + 1 \leq i \leq M_l. \tag{1.15}$$

Definition 1.7 (weighted rNSP in levels)**.** *Let* $(\mathbf{s}, \mathbf{M})$ *be local sparsities and sparsity levels respectively. For weights* $\{w_i\}_{i=1}^N$ $(w_i > 0)$*, we say that* $A \in \mathbb{C}^{m \times N}$ *satisfies the weighted robust null space property in levels (weighted rNSPL) of order* $(\mathbf{s}, \mathbf{M})$ *with constants* $0 < \rho < 1$ *and* $\gamma > 0$ *if for any* $(\mathbf{s}, \mathbf{M})$ *support set* Δ*,*

$$\|x_\Delta\|_{l^2} \leq \rho \|x_{\Delta^c}\|_{l^1_w} / \sqrt{\xi} + \gamma \|Ax\|_{l^2}, \qquad \text{for all } x \in \mathbb{C}^N.$$

We also define the following quantities:

$$\xi = \xi(\mathbf{s}, \mathbf{M}, w) \coloneqq \sum_{l=1}^r w_{(l)}^2 s_l, \quad \zeta = \zeta(\mathbf{s}, \mathbf{M}, w) \coloneqq \min_{l=1,...,r} w_{(l)}^2 s_l, \quad \kappa = \kappa(\mathbf{s}, \mathbf{M}, w) \coloneqq \xi / \zeta.$$

Unless there is ambiguity, we will drop the $(\mathbf{s}, \mathbf{M}, w)$ from the notation of these parameters. Recall the setup throughout this paper of a matrix $A \in \mathbb{C}^{m \times N}$ $(m < N)$, where we have access to an approximation sequence A_l such that $\|A - A_l\| \leq q_l$ with known $q_l \to 0$ as $l \to \infty$. In this regard, the following simple perturbation lemma is useful (whose proof is given in §4).

Lemma 1.8 (**The weighted rNSP in levels is preserved under perturbations or approximations**)**.** *Assume that* Eq. (1.15) *holds and that* A *satisfies the weighted rNSPL of order* $(\mathbf{s}, \mathbf{M})$ *with constants* $0 < \rho < 1$ *and* $\gamma > 0$. *Let* $\hat{A}$ *be an approximation of* A *such that* $\|\hat{A} - A\| < (1 - \rho)\gamma^{-1} \left(1 + \frac{\sqrt{\xi}}{\min_{l=1,...,r} w_{(l)}}\right)^{-1}$. *Then* $\hat{A}$ *satisfies the weighted rNSPL of order* $(\mathbf{s}, \mathbf{M})$ *with new constants*

$$\hat{\rho} = \frac{\rho + \gamma\sqrt{\xi}\|\hat{A} - A\| / \min_{l=1,...,r} w_{(l)}}{1 - \gamma\|\hat{A} - A\|}, \quad \hat{\gamma} = \frac{\gamma}{1 - \gamma\|\hat{A} - A\|}.$$

Lemma 1.8 says that if A satisfied the weighted rNSPL of order $(\mathbf{s}, \mathbf{M})$, then so does A_l for large enough l. Moreover, given the sequence $\{q_l\}$, we can compute how large l must be and the new constants. For ease of exposition, we drop the notational hats from these constants. We can now state our main result, proven in §4.

Theorem 3 (**Stable and accurate neural networks with uniform recovery guarantees can be constructed**). *There exists an algorithm such that for any input sparsity parameters* $(\mathbf{s},\mathbf{M})$*, weights* $\{w_i\}_{i=1}^N$*,* $A\in\mathbb{C}^{m\times N}$ *(with the input* A *given by* $\{A_l\}$*) satisfying the rNSPL with constants* $0<\rho<1$ *and* $\gamma>0$ *(also input), and input parameters* $n\in\mathbb{N}$*,* $\{\delta,b_1,b_2\}\subset\mathbb{Q}_{>0}$ *and* $\upsilon\in(0,1)\cap\mathbb{Q}_{>0}$*, the algorithm outputs a neural network* ϕ_n *such that the following holds. For*

$$C_1=\left(\frac{1+\rho}{2}+(3+\rho)\frac{\kappa^{1/4}}{4}\right)\left(\frac{3+\rho}{1-\rho}\right)\sim\frac{\kappa^{1/4}}{1-\rho},\qquad C_2=2\left(\frac{3+\rho}{1-\rho}+\frac{7+\rho}{1-\rho}\frac{\kappa^{1/4}}{2}\right)\gamma\sim\frac{\kappa^{1/4}\gamma}{1-\rho},$$

1. *(Size)* $\phi_n\in\mathcal{N}_{\mathbf{D}_{(n,p)},3np+1,3}$ *with* $\mathbf{D}_{(n,p)}=(m,\underbrace{2N+m,2(N+m),2N+m+1}_{np\text{ times}},N)$*, where* $p\in\mathbb{N}$ *with the bound* $p\leq\left\lceil\frac{3C_2\|A\|}{\upsilon}\right\rceil$*. Moreover,* $\theta^{-1}\sim p^2(1+\|w\|_{l^2})\max\left\{1,\frac{\|w\|_{l^2}}{\|A\|\gamma\sqrt{\xi}}\right\}$.

2. *(Exponentially Convergent, Uniform and Stable Recovery) For any pair* $(x,y)\in\mathbb{C}^N\times\mathbb{C}^m$ *with*

$$\frac{2C_1}{C_2\sqrt{\xi}}\cdot\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2\|Ax-y\|_{l^2}\leq\delta,\quad\|x\|_{l^2}\leq b_1,\quad\|y\|_{l^2}\leq b_2,$$

we have the following exponentially convergent, uniform and stable recovery guarantees:

$$\|\phi_n(y)-x\|_{l^2}\leq\frac{2C_1}{\sqrt{\xi}}\cdot\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2C_2\cdot\|Ax-y\|_{l^2}+\left(\frac{1+\upsilon}{1-\upsilon}\right)C_2\cdot\delta+b_2C_2\cdot\upsilon^n,\tag{1.16}$$

$$\|\phi_n(y)-x\|_{l^1_w}\leq\left(\frac{3+\rho}{1-\rho}\right)\frac{\sqrt{\xi}}{C_1}\left(\frac{2C_1}{\sqrt{\xi}}\cdot\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2C_2\cdot\|Ax-y\|_{l^2}+\left(\frac{1+\upsilon}{1-\upsilon}\right)C_2\cdot\delta+b_2C_2\cdot\upsilon^n\right).\tag{1.17}$$

Remark 1.9 (The optimal choice of υ)**.** For a total budget of $T=3pn+1$ layers,

$$\upsilon^n=\exp\left(\frac{(T-1)}{3}\left\lceil\frac{3C_2\|A\|}{\upsilon}\right\rceil^{-1}\log(\upsilon)\right)$$

If we ignore the ceiling function, the optimal choice is $\upsilon=e^{-1}$ (strictly speaking Theorem 3 is only stated for rational υ, but we can easily approximate e^{-1}). This yields the error term $\upsilon^n=\exp\left(-\frac{(T-1)}{3}\lceil 3C_2e\|A\|\rceil^{-1}\right)$ and exponential convergence in the number of layers T. This is not optimal. For example, a study of the proof of Theorem 3 shows that we can replace $3C_2$ in the exponential by

$$2\left(\frac{1+\rho}{1-\rho}+\frac{3+\rho}{1-\rho}\frac{\kappa^{1/4}}{2}\right)\gamma+\epsilon$$

for arbitrary $\epsilon>0$. Suppose that we want $b_2C_2\cdot r^n\sim\delta$, then the number of layers required is proportional to $C_2\|A\|\log(b_2\delta^{-1})$, and only grows logarithmically with the precision δ^{-1}. This is made precise in Theorem 4, where we apply Theorem 3 to examples in compressive imaging. ⊠

The proof of Theorem 3 uses the optimisation problem (P_3) (defined in Eq. (1.2)), in the construction of ϕ_n. It is also possible to prove similar results using (P_1) and (P_2), but we do not provide the details. The NNs constructed are approximations of unrolled primal-dual iterations for (P_3), with a careful restart scheme to ensure exponential convergence in the number of layers. Further computational experiments beyond the main test are given in §2. The bounds in Eq. (1.16) and Eq. (1.17) are not quite optimal. If we were able to work in exact arithmetic (taking $\theta\to0$ and $A_l\to A$), we obtain slightly smaller constants, though these do not affect the asymptotic rates.

Remark 1.10 (What happens without restart or with unknown δ?)**.** Without the restart scheme, the convergence in the number of layers scales as $\mathcal{O}(n^{-1})$. However, one can get rid of the assumption $2C_1/(C_2\sqrt{\xi})\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2\|Ax-y\|_{l^2}\leq\delta$. (The assumption is to ensure that the reweighting of the restarts do not become too small - in practice, we found that this was not an issue and the assumption was not needed, with (up to small constants) δ replaced by $2C_1/(C_2\sqrt{\xi})\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2\|Ax-y\|_{l^2}$ in Eq. (1.16). See also the discussion in §2.) More precisely, the proof of Theorem 3 can be adapted to show the following. For an additional input $\beta\in\mathbb{Q}_{>0}$ (and without inputs b_2, δ and υ), there exists an algorithm that computes $\widehat{\phi}_n\in\mathcal{N}_{\mathbf{D}_n,3n+1,3}$ with

$$\mathbf{D}_n=(m+N,\underbrace{2N+m,2(N+m),2N+m+1}_{n\text{ times}},N),$$

such that for any $x,x_0\in\mathbb{C}^N$ with $\|x\|_{l^2}\leq b_1$ and all $y\in\mathbb{C}^m$, the following reconstruction guarantees hold:

$$\|\widehat{\phi}_n(y,x_0)-x\|_{l^2}\leq\frac{2C_1}{\sqrt{\xi}}\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2C_2\left[\|Ax-y\|_{l^2}+\frac{\|A\|}{n}\left(\frac{\|x-x_0\|_{l^2}^2}{\beta}+\beta\right)\right],\tag{1.18}$$

$$\|\widehat{\phi}_n(y,x_0)-x\|_{l^1_w}\leq\left(\frac{3+\rho}{1-\rho}\right)\frac{\sqrt{\xi}}{C_1}\left(\frac{2C_1}{\sqrt{\xi}}\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+2C_2\left[\|Ax-y\|_{l^2}+\frac{\|A\|}{n}\left(\frac{\|x-x_0\|_{l^2}^2}{\beta}+\beta\right)\right]\right).\tag{1.19}$$

Here, x_0 should be interpreted as an initial guess (an arbitrary input to the NNs) and β should be interpreted as a scaling parameter, with optimal scaling $\beta\sim\|x-x_0\|_{l^2}$. A good choice for β is $\|x\|_{l^2}$, or, when this is unknown, $\|y\|_{l^2}/\|A\|$. For completeness, we provide a proof sketch of Eq. (1.18) and Eq. (1.19) at the end of §4.C. ⊠

Algorithm unrolling is particularly well-suited to scenarios where it is difficult to collect large training samples. However, training a finite fixed number of layers typically incurs the same stability and generalisation issues mentioned above. Moreover, learning the weights and biases usually prevents the convergence analysis of standard (unlearned) iterative methods from carrying over. In particular, there is no guarantee of objective function minimisation (let alone convergence of the iterated arguments) or any form of convergence as the number of layers increases. A subtle, yet fundamental, point regarding iterative methods, whether they are unrolled as a NN and supplemented with learned parameters or not, is the following. Theorem 2 states that, in general, the optimisation problems (P_1), (P_2), and (P_3) are non-computable. This is despite the fact that there are many results in the literature describing rates of convergence for iterative methods. The resolution of this apparent puzzle is that convergence results regarding iterative methods are typically given in terms of the *objective function* that is being minimised (see also Theorem 5, which we use to prove Theorem 3). As the proof of Theorem 3 shows, it is crucial to have conditions such as the rNSPL to convert these objective function bounds to the desired error bounds on the distance to the *minimisers* or vector x. Moreover, this property has the key effect of allowing exponential convergence through restarting and reweighting.

C. Examples in image recovery. As an example application of Theorem 3, we consider the case of Fourier and Walsh sampling, using the Haar wavelets as the sparsifying transform. Our results can be generalised to the infinite-dimensional setting with the use of higher-order Daubechies wavelets (though the results are more complicated to write down), and we refer the reader to (9) for compressed sensing in infinite dimensions. We first define the concept of multilevel random subsampling (10).

Definition 1.11 (Multilevel random subsampling)**.** *Let* $\mathbf{N} = (N_1, \dots, N_l) \in \mathbb{N}^l$, *where* $1 \leq N_1 < \cdots < N_l = N$ *and* $\mathbf{m} = (m_1, \dots, m_l) \in \mathbb{N}^l$ *with* $m_k \leq N_k - N_{k-1}$ *for* $k = 1, \dots, l$, *and* $N_0 = 0$. *For each* $k = 1, \dots, l$, *let* $\mathcal{I}_k = \{N_{k-1}+1, \dots, N_k\}$ *if* $m_k = N_k - N_{k-1}$ *and if not, let* $t_{k,1}, \dots, t_{k,m_k}$ *be chosen uniformly and independently from the set* $\{N_{k-1}+1, \dots, N_k\}$ *(with possible repeats), and set* $\mathcal{I}_k = \{t_{k,1}, \dots, t_{k,m_k}\}$. *If* $\mathcal{I} = \mathcal{I}_{\mathbf{N},\mathbf{m}} = \mathcal{I}_1 \cup \cdots \cup \mathcal{I}_l$ *we refer to* $\mathcal{I}$ *as an* $(\mathbf{N}, \mathbf{m})$*-multilevel subsampling scheme.*

Definition 1.12 (Multilevel subsampled unitary matrix)**.** *A matrix* $A \in \mathbb{C}^{m \times N}$ *is an* $(\mathbf{N}, \mathbf{m})$*-multilevel subsampled unitary matrix if* $A = P_{\mathcal{I}} D U$ *for a unitary matrix* $U \in \mathbb{C}^{N \times N}$ *and* $(\mathbf{N}, \mathbf{m})$*-multilevel subsampling scheme* $\mathcal{I}$. *Here* D *is a diagonal scaling matrix with diagonal entries*

$$D_{ii} = \sqrt{\frac{N_k - N_{k-1}}{m_k}}, \quad i = N_{k-1}+1, ..., N_k, \quad k = 1, ..., l$$

and $P_{\mathcal{I}}$ *denotes the projection onto the linear span of the subset of the canonical basis indexed by* $\mathcal{I}$.

Throughout this subsection, we let $K = 2^r$ for $r \in \mathbb{N}$, and consider vectors on $\mathbb{C}^K$ or d-dimensional tensors on $\mathbb{C}^{K \times \cdots \times K}$. To keep consistent notation with previous sections, we set $N = K^d$ so that the objective is to recover a vectorised $x \in \mathbb{C}^N$. The following can also be generalised to rectangles (i.e. $\mathbb{C}^{2^{r_1} \times \cdots \times 2^{r_d}}$ with possibly different $r_1, ..., r_d$) or dimensions that are not powers of two.

Let $V \in \mathbb{C}^{N \times N}$ be either the matrix $F^{(d)}$ or $W^{(d)}$, corresponding to the $d-$dimensional discrete Fourier or Walsh transform (see §A). In the Fourier case, we divide the different frequencies $\{-K/2+1, \dots, K/2\}^d$ into dyadic bands. For $d = 1$, we let $B_1 = \{0, 1\}$ and $B_k = \{-2^{k-1}+1, \dots, -2^{k-2}\} \cup \{2^{k-2}+1, \dots, 2^{k-1}\}$ for $k = 2, \dots, r$. In the Walsh case, we define the frequency bands $B_1 = \{0, 1\}$ and $B_k = \{2^{k-1}, \dots, 2^k - 1\}$ for $k = 2, \dots, r$ in the one-dimensional case. In the general d-dimensional case for Fourier or Walsh sampling, we set

$$B_{\mathbf{k}}^{(d)} = B_{k_1} \times \ldots \times B_{k_d}, \quad \mathbf{k} = (k_1, \dots, k_d) \in \mathbb{N}^d.$$

For a d-dimensional tensor $c \in \mathbb{C}^{K \times \cdots \times K}$, we assume we can observe subsampled measurements of $V\mathrm{vec}(c)$, where $\mathrm{vec}(c) \in \mathbb{C}^N$ is a vectorised version of c. To recover a sparse representation, we consider the Haar wavelet coefficients. We denote the discrete Haar Wavelet transform by $\Phi \in \mathbb{C}^{N \times N}$, and note that $\Psi^* = \Psi^{-1}$ since Ψ is unitary. In other words, we consider a multilevel subsampled unitary matrix (Definition 1.12), with $U = V\Psi^*$. Given $\{m_{\mathbf{k}=(k_1,\dots,k_d)}\}_{k_1,\dots,k_d=1}^r$, we use a multilevel random sampling such that $m_{\mathbf{k}}$ measurements are chosen from $B_{\mathbf{k}}^{(d)}$ according to Definition 1.11. This corresponds to $l = r^d$ and the N_i's can be chosen given a suitable ordering of the Fourier/Walsh basis. The sparsity in levels structure (Definition 1.6) is chosen to correspond to the r wavelet levels. A pictorial representation is given in Figure S1. Finally, we define

$$\mathcal{M}_{\mathcal{F}}(\mathbf{s}, \mathbf{k}) := \sum_{j=1}^{\|\mathbf{k}\|_{l^\infty}} s_j \prod_{i=1}^{d} 2^{-|k_i - j|} + \sum_{j=\|\mathbf{k}\|_{l^\infty}+1}^{r} s_j 2^{-2(j - \|\mathbf{k}\|_{l^\infty})} \prod_{i=1}^{d} 2^{-|k_i - j|} \quad [1.20]$$

$$\mathcal{M}_{\mathcal{W}}(\mathbf{s}, \mathbf{k}) := s_{\|\mathbf{k}\|_{l^\infty}} \prod_{i=1}^{d} 2^{-|k_i - \|\mathbf{k}\|_{l^\infty}|}. \quad [1.21]$$

For notational convenience, we also define $\mathcal{Z} = \max\left\{1, \frac{\max_{l=1,\dots,r} w_{(l)} \sqrt{(M_l - M_{l-1})}}{\sqrt{\xi(\mathbf{s}, \mathbf{M}, w)}}\right\}$.

We now state the main theorem of this subsection (proven in §5), which states how many samples are needed and the number of layers of the NN needed, which only depends logarithmically on the error δ, a consequence of the exponential convergence in Theorem 3. We discuss the sampling conditions below.

Theorem 4. *Consider the above setup of recovering a d-dimensional tensor $c \in \mathbb{C}^{K^d}$ ($N = K^d$) from subsampled Fourier or Walsh measurements Vc, such that A is a multilevel subsampled unitary matrix with respect to $U = V\Psi^*$. Let $\epsilon_{\mathbb{P}} \in (0,1)$ and $\mathcal{L} = d \cdot r^2 \cdot \log(2m) \cdot \log^2(s \cdot \kappa(\mathbf{s},\mathbf{M},w)) + \log(\epsilon_{\mathbb{P}}^{-1})$. Suppose that:*

- *(a) In the Fourier case*

$$m_{\mathbf{k}} \gtrsim \kappa(\mathbf{s},\mathbf{M},w) \cdot \mathcal{M}_{\mathcal{F}}(\mathbf{s},\mathbf{k}) \cdot \mathcal{L}. \tag{1.22}$$

- *(b) In the Walsh case*

$$m_{\mathbf{k}} \gtrsim \kappa(\mathbf{s},\mathbf{M},w) \cdot \mathcal{M}_{\mathcal{W}}(\mathbf{s},\mathbf{k}) \cdot \mathcal{L}. \tag{1.23}$$

Then with probability at least $1-\epsilon_{\mathbb{P}}$, A satisfies the weighted rNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $\rho = 1/2$ and $\gamma = \sqrt{2}$. The conclusion of Theorem 3 then holds for the uniform recovery of the Haar wavelet coefficients

$$x = \Psi c \in \mathbb{C}^N. \tag{1.24}$$

Moreover, for any $\delta \in (0,1)$, let $\mathcal{J}(\delta,\mathbf{s},\mathbf{M},w)$ be the collection of all $y \in \mathbb{C}^m$ with $y = P_{\mathcal{I}}DVc + e$ where

$$\|c\|_{l^2} \leq 1, \quad \max\left\{ \frac{\sigma_{\mathbf{s},\mathbf{M}}(\Psi c)_{l^1_w}}{\sqrt{\xi}}, \|e\|_{l^2} \right\} \leq \delta. \tag{1.25}$$

Then we construct via an algorithm, a neural network $\phi \in \mathcal{N}_{\mathbf{D},3n+1,3}$ such that with probability at least $1-\epsilon_{\mathbb{P}}$,

$$\|\phi(y) - c\|_{l^2} \lesssim \kappa^{1/4}\delta, \quad \forall y = P_{\mathcal{I}}DVc + e \in \mathcal{J}(\delta,\mathbf{s},\mathbf{M},w). \tag{1.26}$$

The network parameters are

$$\mathbf{D} = (m, \underbrace{2N+m, 2(N+m), 2N+m+1}_{n \text{ times}}, N), \quad n \leq \left\lceil \log\left(\delta^{-1}\mathcal{Z}\right)\kappa^{1/4}\mathcal{Z} \right\rceil. \tag{1.27}$$

The sampling conditions in Eq. (1.22) and Eq. (1.23) are optimised by minimising $\kappa(\mathbf{s},\mathbf{M},w)$. Up to a constant scale, this corresponds to the choice $w_{(j)} = \sqrt{s/s_j}$ and

$$n = \left\lceil \log\left(\delta^{-1} \max_{j=1,\dots,r} \sqrt{\max\left\{1, \frac{M_j - M_{j-1}}{r s_j}\right\}} \right) r^{1/4} \max_{j=1,\dots,r} \sqrt{\max\left\{1, \frac{M_j - M_{j-1}}{r s_j}\right\}} \right\rceil.$$

Up to log-factors, the measurement condition then becomes equivalent to the currently best-known oracle estimator (where one assumes apriori knowledge of the support of the vector) (11, Prop. 3.1). For Fourier measurements, we can interpret the condition as follows. For $d = 1$, this estimate yields the sampling estimates

$$m_k \gtrsim \left(\sum_{j=1}^{k} s_j 2^{-|k-j|} + \sum_{j=k+1}^{r} s_j 2^{-3|k-j|} \right) \cdot r \cdot \mathcal{L}.$$

In other words, up to logarithmic factors and exponentially small terms, s_j measurements are needed in each level. Furthermore, if $s_1 = \ldots = s_r = s_*$ and $d = 2$ then Eq. (1.22) holds if

$$m_{(k_1,k_2)} \gtrsim s_* 2^{-|k_1-k_2|} \cdot r \cdot \mathcal{L}. \tag{1.28}$$

Another interpretation is gained by considering

$$m_k = \sum_{\|\mathbf{k}\|_{l^\infty} = k} m_{\mathbf{k}}, \quad k = 1,\dots,r,$$

the number of samples per annular region. We then have

$$m_k \gtrsim 3^d d \left(s_k + \sum_{l=1}^{k-1} s_l 2^{-(k-l)} + \sum_{l=k+1}^{r} s_l 2^{-3(l-k)} \right) \cdot r \cdot \mathcal{L}, \tag{1.29}$$

which is the same estimate as the one-dimensional case for bounded d. Note that the number of samples required in each annular region is (up logarithmic factors) proportional to the corresponding sparsity s_k with additional exponentially decaying terms dependent on $s_l, l \neq k$. This leads to a measurement condition on the total number of measurements $m = m_1 + \cdots + m_r$, of the form

$$m \gtrsim 3^d d\,(s_1 + \cdots + s_r) \cdot r \cdot \mathcal{L}.$$

In the case of Walsh sampling, Eq. (1.28) remains the same whereas Eq. (1.29) becomes $m_k \gtrsim 2^d \cdot d \cdot r \cdot \mathcal{L} \cdot s_k$, with no terms from the sparsity levels $s_l, l \neq k$.

2. Further examples of FIRENET

A. Generalisation properties. To demonstrate the generalisation properties of our NNs, Figure S2 shows the stability test (see main text) applied to FIRENETs for a range of images. This shows stability across different types of images and highlights an important fact. Namely, methods based on conditions such as Definition 1.7 allow great generalisation properties and avoid time-consuming and expensive retraining of NNs for different classes of images. As well as being rigorously proven to be stable, FIRENETs are accurate for images that are sparse in wavelets. As most images are sparse in wavelets, these networks also show great generalisation properties to unseen images.

B. Exponential convergence. We now provide a computational experiment to demonstrate the convergence in the number of layers stated in Theorem 4 (and Theorem 3). Note that the matrix A and its adjoint can be implemented rapidly using the fast Fourier transform (or fast Walsh–Hadamard transform). We take the image shown in Figure S3, a subsampling rate of only 15%, and corrupt the measurements by adding 2% Gaussian noise. Figure S3 shows the reconstructions using Fourier and Walsh sampling and Haar wavelets. Similar results hold for other wavelets, such as Daubechies wavelets with a larger number of vanishing moments. In fact, the reconstruction results are better than those shown for the Haar wavelet system. We have chosen to show the Haar wavelet results because this is the system for which Theorem 4 is stated. For the reconstruction, we take $\lambda = 0.00025, \tau = \sigma = 1$, $p = 5$ and the weights as discussed in §1.C. In the spirit of no parameter tuning, the weights were selected based on a standard phantom image, and not the image we use to test the algorithm. These parameters are certainly not optimal, and instead were chosen simply to emphasise that we have deliberately avoided parameter tuning. Moreover, we found that the choice of δ in the algorithm was of little consequence, so have taken $\delta = 10^{-9}$.

Figure S4 shows the convergence in the number of inner iterations (or, equivalently, n - the total number of inner iterations is np, and hence we have not specified n, which is typically chosen to be 5). We show the error between the constructed image after j iterations (denoted by c_j) and the true image (denoted by c), as well as the convergence of the objective function which we denote by F in the figure caption. To compute the minimum of F, denoted F^*, we ran several thousand iterates of the non-restarted version of the algorithm so that the error in the value of F^* is at least an order of magnitude smaller than the shown values of $F(c_j) - F^*$. Whilst the objective function is guaranteed to converge to the minimum value when computing F^* this way, there is no guarantee that the vectors computed by the non-restarted version converge to a minimiser, as demonstrated by the non-computability results in Theorem 2. However, in this case, the non-restarted version converged to a vector c^* up to an error much smaller than $\|c - c^*\|_{l^2}$. Hence $\|c - c^*\|_{l^2}$ indicates the minimum error we can expect from using (P_3) to recover the image.

The figure shows the expected exponential convergence, as the number of inner iterations increases, of the objective function values as well as c_j to c until the error is of the order $\|c - c^*\|_{l^2}$. This corresponds to an initial phase of exponential convergence, where the υ^{-n} term (with $\upsilon = e^{-1}$) is dominant in Theorem 3, followed by a plateau to the minimal error $\|c - c^*\|_{l^2}$ (shown as the dotted line). This plateau occurs due to inexact measurements (the noise) and the fact that the image does not have exactly sparse wavelet coefficients. This corresponds to the robust null space property (in levels) only being able to bound the distance $\|c - c_j\|_{l^2}$ up to the same order as $\|c - c^*\|_{l^2}$. In other words, we can only accelerate convergence up to this error bound. The error plateau disappears in the limit of exactly sparse vectors and zero noise (in the limit $\delta \downarrow 0$ in Theorem 3), and one gains exponential convergence down to essentially machine precision. Finally, the acceleration is of great practical interest. Rather than the several hundreds (or even thousands) of iterations that are typically needed for solving compressed sensing optimisation problems with first-order iterative methods, we obtain optimal accuracy in under 20 iterations. This was found for a range of different images, subsampling rates etc. The fact that so few layers are needed, coupled with the fast transforms for implementing the affine maps in the NNs, makes the NNs very computationally efficient and competitive speed-wise with state-of-the-art DL.

3. Proof of Theorem 2 and tools from the Solvability Complexity Index (SCI) hierarchy

In this section, we prove Theorem 2. To do this, we rely on some of the mathematics behind the SCI hierarchy (12–25) and the extended Smale's 9th problem (26, 27) – a subset of the SCI program that will be presented below. However, we start with some analytical results regarding phase transitions of solutions of $(P_1), (P_2)$ and (P_3) which are given in §3.B. However, before analysing these phase transitions, we need some preliminary definitions regarding algorithms, inexact inputs, and condition numbers. There are two main reasons for this framework. First, because our definitions are general, they lead to stronger impossibility results than when restricted to specific models of computation. Second, our framework greatly simplifies the proofs and makes it clear what the key mechanisms behind the proofs are (§3.B describes this in terms of phase transitions of minimisers). The following discussions are self-contained.

A. Algorithmic preliminaries: a user-friendly guide. We begin with a definition of a computational problem, which is deliberately general in order to capture any computational problem.

Definition 3.1 (Computational problem)**.** *Let Ω be some set, which we call the domain, and Λ be a set of complex valued functions on Ω such that for $\iota_1, \iota_2 \in \Omega$, then $\iota_1 = \iota_2$ if and only if $f(\iota_1) = f(\iota_2)$ for all $f \in \Lambda$, called an evaluation set. Let $(\mathcal{M}, d)$ be a metric space, and finally let $\Xi : \Omega \to \mathcal{M}$ be a function which we call the problem function. We call the collection $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ a computational problem. When it is clear what $\mathcal{M}$ and Λ are, we write $\{\Xi, \Omega\}$ for brevity.*

Remark 3.2 (Multivalued problems)**.** In some cases, such as when considering the optimisation problems (P_j) that may have more than one solution, we consider $\Xi(\iota) \subset \mathcal{M}$. With an abuse of notation, we then set $d(x, \Xi(\iota)) = \text{dist}(x, \Xi(\iota)) = \inf_{y \in \Xi(\iota)} d(x, y)$ and this distinction will be made clear from context. ⊠

The set Ω is the set of objects that give rise to our computational problems. The problem function $\Xi : \Omega \to \mathcal{M}$ is what we are interested in computing. Finally, the set Λ is the collection of functions that provide us with the information we are allowed to read as input to an algorithm. For example, Ω could consist of a collection of matrices A and data y in Eq. (1.1), Λ could consist of the pointwise entries of the vectors and matrices in Ω, Ξ could represent the solution set (with the possibility of more than one solution as in Remark 3.2) of any of the problems (P_j) and $(\mathcal{M}, d)$ could be $\mathbb{C}^N$ with the usual Euclidean metric (or any other suitable metric).

Given the definition of a computational problem, we need the definition of a general algorithm, whose conditions hold for any reasonable notion of a deterministic algorithm. Throughout this paper, we deal with the case that $\Lambda = \{f_j\}_{j \in \beta}$, where β is some (at most) countable index set. Following (12, 14, 15, 26) we use the concept of a general algorithm.

Definition 3.3 (General Algorithm)**.** *Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, a general algorithm is a mapping $\Gamma : \Omega \to \mathcal{M}$ such that for each $\iota \in \Omega$*

(i) There exists a non-empty finite subset of evaluations $\Lambda_\Gamma(\iota) \subset \Lambda$,

(ii) The action of Γ on ι only depends on $\{\iota_f\}_{f \in \Lambda_\Gamma(\iota)}$ where $\iota_f \coloneqq f(\iota)$,

(iii) For every $\kappa \in \Omega$ such that $\kappa_f = \iota_f$ for every $f \in \Lambda_\Gamma(\iota)$, it holds that $\Lambda_\Gamma(\kappa) = \Lambda_\Gamma(\iota)$.

If, in addition, there exists a canonical ordering $\Lambda_\Gamma(\iota) = \{f^\Gamma_{\iota,1} = f_{k_1}, ..., f^\Gamma_{\iota,S_\Gamma(\iota)} = f_{k_{S_\Gamma(\iota)}}\}$, where $S_\Gamma(\iota) = |\Lambda_\Gamma(\iota)|$, such that if $\kappa \in \Omega$ and $f^\Gamma_{\iota,j}(\iota) = f^\Gamma_{\iota,j}(\kappa)$ for all $j \leq r < S_\Gamma(\iota)$, then $f^\Gamma_{\iota,j} = f^\Gamma_{\kappa,j}$ for all $j \leq r+1$, then we call Γ a Sequential General Algorithm. In this case, we use the notation $k_j(\Gamma, \iota)$ to denote the ordered indices corresponding to the evaluation functions that the algorithm reads.

The three properties of a general algorithm are the most basic natural properties we would expect any deterministic computational device to obey. The first condition says that the algorithm can only take a finite amount of information, though it is allowed adaptively to choose, depending on the input, the finite amount of information that it reads. The second condition ensures that the algorithm's output only depends on its input, or rather the information that it has accessed (or "read"). The final condition is very important and ensures that the algorithm produces outputs and accesses information consistently. In other words, if it sees the same information for two different inputs, then it cannot behave differently for those inputs. Note that the definition of a general algorithm is more general than the definition of a Turing machine (28) or a Blum–Shub–Smale (BSS) machine (29), which can be thought of as digital and analog computational devices respectively. In particular, a general algorithm has no restrictions on the operations allowed. The extra condition for a sequential general algorithm is satisfied by any algorithm defined by a computational machine with input of readable information (one should think of the ordered indices of the evaluation functions as corresponding to sequentially reading the tape which encodes the input information). Hence, a sequential general algorithm is still more general than a Turing or a BSS machine. Complete generality in Definition 3.3 is used for two primary reasons:

(i) *Strongest possible bounds:* Since Definition 3.3 is completely general, the lower bounds hold in any model of computation, such as a Turing machine or a BSS machine. On the other hand, the algorithms we construct in this paper are made to work using only arithmetic operations over the rationals. Hence, we obtain the strongest possible lower bounds and the strongest possible upper bounds.

(ii) *Simplified exposition:* Using the concept of a general algorithm considerably simplifies the proofs of lower bounds and allows us to see precisely the mechanisms behind the proofs.

Next, we consider the definition of a randomised general algorithm, which again is more general than a probabilistic Turing or probabilistic BSS machine. Randomised algorithms are widely used in practice in areas such as optimisation, algebraic computation, machine learning, and network routing. In the case of Turing machines, it is currently unknown, in the sense of polynomial runtime, whether randomisation is beneficial from a complexity class viewpoint (30, Ch. 7), however, rather intriguingly, this is not the case for BSS machines (29, Ch. 17) (some of the proofs in this reference are non-constructive - it is an open problem whether any probabilistic BSS machine can be simulated by a deterministic machine having the same machine constants and with only a polynomial slowdown). Nevertheless, randomisation is an extremely useful tool in practice. From a machine learning point of view, we also want to consider randomised algorithms to capture procedures such as stochastic gradient descent which are commonly used to train NNs. As developed in (26), the concept of a general algorithm can be extended to a randomised general algorithm. This concept allows for universal impossibility results regardless of the computational model.

Definition 3.4 (Randomised General Algorithm)**.** *Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, a Randomised General Algorithm (RGA) Γ^{ran} is a collection X of general algorithms $\Gamma : \Omega \to \mathcal{M}$, a sigma-algebra $\mathcal{F}$ on X and a family of probability measures $\{\mathbb{P}_\iota\}_{\iota \in \Omega}$ on $\mathcal{F}$ such that the following conditions hold:*

1. *For each $\iota \in \Omega$, the mapping $\Gamma_\iota^{\mathrm{ran}} : (X, \mathcal{F}) \to (\mathcal{M}, \mathcal{B})$ defined by $\Gamma_\iota^{\mathrm{ran}}(\Gamma) = \Gamma(\iota)$ is a random variable, where $\mathcal{B}$ is the Borel sigma-algebra on $\mathcal{M}$.*

2. *For each $n \in \mathbb{N}$ and $\iota \in \Omega$, the set $\{\Gamma \in X : \sup\{m \in \mathbb{N} : f_m \in \Lambda_\Gamma(\iota)\} \leq n\} \in \mathcal{F}$.*

3. *For each $\iota_1, \iota_2 \in \Omega$ and $E \in \mathcal{F}$, such that for every $\Gamma \in E$ we have $f(\iota_1) = f(\iota_2)$ for every $f \in \Lambda_\Gamma(\iota_1)$, then $\mathbb{P}_{\iota_1}(E) = \mathbb{P}_{\iota_2}(E)$.*

With slight abuse of notation, we denote the family of randomised general algorithms by RGA.

The first two conditions are measure theoretic to avoid pathological cases and ensure that "natural sets" one might define for a random algorithm (such as notions of stopping times) are measurable. These conditions hold for all standard probabilistic machines (such as a Turing or BSS machine). The third condition ensures consistency, namely, that in the case of identical evaluations, the laws of the output cannot change. Finally, we will use the standard definition of a probabilistic Turing machine (which is a particular case of Definition 3.4). However, to make sense of probabilistic Turing machines in our context (in particular, to restrict operations to the rationals which can be encoded by the natural numbers), we must define the notion of inexact input.

Suppose we are given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, and that $\Lambda = \{f_j\}_{j\in\beta}$, where we remind the reader that β is some index set that can be finite or countably infinite. However, obtaining f_j may be a computational task on its own, which is exactly the problem in most areas of computational mathematics. In particular, for $\iota \in \Omega$, $f_j(\iota)$ could be the number $e^{\frac{\pi}{j}i}$ for example. Hence, we cannot access or store $f_j(\iota)$ on a computer, but rather $f_{j,n}(\iota)$ where $f_{j,n}(\iota) \to f_j(\iota)$ as $n \to \infty$. This idea is formalised in the definition below, however, to put this in perspective it is worth mentioning the Solvability Complexity Index (SCI) hierarchy.

Remark 3.5 (The Solvability Complexity Index (SCI) hierarchy (12, 14, 15, 26))**.** The SCI of a computational problem is the smallest number of limits needed in order to compute the solution. The full hierarchy is described in (14), and the mainstay of the hierarchy are the Δ_k^α classes. The α denotes the model of computation. Informally, we have the following description. Given a collection $\mathcal{C}$ of computational problems, then

(i) Δ_0^α is the set of problems that can be computed in finite time, the SCI $= 0$.

(ii) Δ_1^α is the set of problems that can be computed using one limit (the SCI $= 1$) with control of the error, i.e. $\exists$ a sequence of algorithms $\{\Gamma_n\}$ such that $d(\Gamma_n(\iota), \Xi(\iota)) \leq 2^{-n}$, $\forall \iota \in \Omega$.

(iii) Δ_2^α is the set of problems that can be computed using one limit (the SCI $= 1$) without error control, i.e. $\exists$ a sequence of algorithms $\{\Gamma_n\}$ such that $\lim_{n\to\infty} \Gamma_n(\iota) = \Xi(\iota)$, $\forall \iota \in \Omega$.

(iv) Δ_{m+1}^α, for $m \in \mathbb{N}$, is the set of problems that can be computed by using m limits, (the SCI $\leq m$), i.e. $\exists$ a family of algorithms $\{\Gamma_{n_m,\dots,n_1}\}$ with $\lim_{n_m\to\infty} \dots \lim_{n_1\to\infty} \Gamma_{n_m,\dots,n_1}(\iota) = \Xi(\iota)$, $\forall \iota \in \Omega$. ⊠

The above hierarchy gives rise to the concept of 'Δ_1-information.' That is, in informal terms, the problem of obtaining the inexact input to the computational problem is a Δ_1 problem. One may think of an algorithm taking the number exp(1) or $\sqrt{2}$ as input. Indeed, one can never produce an exact version of these numbers to the algorithm, however, one can produce an approximation to an arbitrarily small error.

Definition 3.6 (Δ_1-information (14, 15, 26))**.** *Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem. We say that Λ has Δ_1-information if each $f_j \in \Lambda$ is not available, however, there are mappings $f_{j,n} : \Omega \to \mathbb{Q} + i\mathbb{Q}$ such that $|f_{j,n}(\iota) - f_j(\iota)| \leq 2^{-n}$ for all $\iota \in \Omega$. Finally, if $\widehat{\Lambda}$ is a collection of such functions described above such that Λ has Δ_1-information, we say that $\widehat{\Lambda}$ provides Δ_1-information for Λ. Moreover, we denote the family of all such $\widehat{\Lambda}$ by $\mathcal{L}^1(\Lambda)$.*

We want to have algorithms that can handle all computational problems $\{\Xi, \Omega, \mathcal{M}, \widehat{\Lambda}\}$ whenever $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$. In order to formalise this, we define what we mean by a computational problem with Δ_1-information.

Definition 3.7 (Computational problem with Δ_1-information)**.** *A computational problem where Λ has Δ_1-information is denoted by $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} := \{\widetilde{\Xi}, \widetilde{\Omega}, \mathcal{M}, \widetilde{\Lambda}\}$, where*

$$\widetilde{\Omega} = \left\{\widetilde{\iota} = \{f_{j,n}(\iota)\}_{j,n\in\beta\times\mathbb{N}} : \iota \in \Omega, \{f_j\}_{j\in\beta} = \Lambda, |f_{j,n}(\iota) - f_j(\iota)| \leq 2^{-n}\right\},$$

Moreover, if $\widetilde{\iota} = \{f_{j,n}(\iota)\}_{j,n\in\beta\times\mathbb{N}} \in \widetilde{\Omega}$ then we define $\widetilde{\Xi}(\widetilde{\iota}) = \Xi(\iota)$ and $\widetilde{f}_{j,n}(\widetilde{\iota}) = f_{j,n}(\iota)$. We also set $\widetilde{\Lambda} = \{\widetilde{f}_{j,n}\}_{j,n\in\beta\times\mathbb{N}}$. Note that $\widetilde{\Xi}$ is well-defined by Definition 3.1 of a computational problem and the definition of $\widetilde{\Omega}$ includes all possible instances of Δ_1-information $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$.

We can now define a probabilistic Turing machine for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where the algorithm Γ is executed by a Turing machine (28), that has an oracle tape consisting of $\{\widetilde{\iota}_f\}_{f\in\widetilde{\Lambda}}$. In what follows, we have deliberately not written down the (lengthy) definition of a Turing machine (found in any standard text (30)), which one should think of as an effective algorithm or computer programme (the famous Church–Turing thesis).

Definition 3.8. *Given the definition of a Turing machine, a probabilistic Turing machine for* $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ *is a Turing machine that has an oracle tape consisting of* $\{\widetilde{\iota}_f\}_{f\in\widetilde{\Lambda}}$ *(for* $\widetilde{\iota} \in \widetilde{\Omega}$*), with an additional read-only tape containing independent binary random numbers (*0 *or* 1 *with equal probability), and which halts with probability one and outputs a single element of* $\mathcal{M}$*. The law of such a machine will be denoted by* $\mathbb{P}$*. With an abuse of notation, we sometimes denote the probabilistic Turing machine by* $(\Gamma, \mathbb{P})$*.*

Remark 3.9 (Where does the output live?)**.** Strictly speaking, when we say that the output of a probabilistic Turing machine lies in $\mathcal{M}$, we mean that the output corresponds, via an encoding, to an element of a subset of $\mathcal{M}$ such as $(\mathbb{Q}+i\mathbb{Q})^N \subset \mathbb{C}^N$. However, we follow the usual convention of suppressing such encodings. ⊠

One should think of Definition 3.8 as an algorithm for the computational problem with inexact input, but with the additional ability to generate random numbers (corresponding to the binary input tape) and execute commands based on the sequence of random numbers that are generated. The reader should intuitively think of this as a computer program with a random number generator. For equivalent definitions and the basic properties of such machines, see (30). For simplicity, we have only considered probabilistic Turing machines that halt with probability one, though extensions can be made to non-halting machines. Note that Definition 3.8 is a special case of Definition 3.4, where $\mathbb{P}_\iota = \mathbb{P}$ is fixed across different ι. In particular, given a probabilistic Turing machine, the sigma-algebra and probability distribution generated by the standard product topology on $\{0,1\}^{\mathbb{N}}$ induce the relevant collection X of Turing machines and sigma-algebra $\mathcal{F}$, as well as $\mathbb{P}$.

Finally, we recall standard definitions of condition used in optimisation (29, 31). The classical condition number of an invertible matrix A is given by $\mathrm{Cond}(A) = \|A\|\|A^{-1}\|$. For different types of condition numbers related to a possibly multivalued (signified by the double arrow) mapping $\Xi : \Omega \subset \mathbb{C}^n \rightrightarrows \mathbb{C}^N$ we need to establish what types of perturbations we are interested in. For example, if Ω denotes the set of diagonal matrices (which we treat as elements of $\mathbb{C}^n$ for some n), we may not be interested in perturbations in the off-diagonal elements as they will always be zero. In particular, we may only be interested in perturbations in the coordinates that are varying in the set Ω. Thus, given $\Omega \subset \mathbb{C}^n$ we define the active coordinates of Ω to be $\mathrm{Act} = \mathrm{Act}\,(\Omega) = \{j \,:\, \exists\, x, y \in \Omega, x_j \neq y_j\}$. Moreover, for $\nu > 0$ (including the obvious extension to $\nu = \infty$),

$$\Omega_\nu = \{x \,:\, \exists\, y \in \Omega \text{ such that } \|x-y\|_{l^\infty} \leq \nu, x_{\mathrm{Act}^c} = y_{\mathrm{Act}^c}\}\,.$$

In other words, Ω_ν is the set of ν-perturbations along the non-constant coordinates of elements in Ω. We can now recall some of the classical condition numbers from the literature (29, 31).

(1) *Condition of a mapping*: Let $\Xi : \Omega \subset \mathbb{C}^n \rightrightarrows \mathbb{C}^m$ be a linear or non-linear mapping, and suppose that Ξ is also defined on Ω_ν for some $\nu > 0$. Then,

$$\mathrm{Cond}\,(\Xi, \Omega) = \sup_{x\in\Omega} \lim_{\epsilon\to 0^+} \sup_{\substack{x+z\in\Omega_\nu \\ 0<\|z\|_{l^2}\leq\epsilon}} \left\{ \frac{\mathrm{dist}(\Xi(x+z), \Xi(x))}{\|z\|_{l^2}} \right\},$$

where we allow for multivalued functions by defining $\mathrm{dist}(\Xi(x+z), \Xi(z)) = \inf_{w_1\in\Xi(x+z), w_2\in\Xi(z)} \|w_1 - w_2\|_{l^2}$ (see Remark 3.2). We will use this notion of condition number for (P_1), (P_2) and (P_3).

(2) *Distance to infeasibility - the Feasibility Primal condition number*: For the problem (P_1) of basis pursuit (for (P_2) and (P_3) the following condition number is always zero) we set

$$\nu(A,y) = \sup\left\{\epsilon \geq 0 \,:\, \|\hat{y}\|_{l^2}, \|\hat{A}\| \leq \epsilon, (A+\hat{A}, y+\hat{y}) \in \Omega_\infty \Rightarrow (A+\hat{A}, y+\hat{y}) \text{ are feasible inputs to } \Xi_{P_1}\right\},$$

and define the *Feasibility Primal* (FP) local condition number $C_{\mathrm{FP}}(A,y) \coloneqq \frac{\max\{\|y\|_{l^2}, \|A\|\}}{\nu(A,y)}$. We then define the FP global condition number via $C_{\mathrm{FP}}\,(\Xi_{P_1}, \Omega) \coloneqq \sup_{(A,y)\in\Omega} C_{\mathrm{FP}}(A,y)$.

B. Phase transitions. To prove Theorem 2, we use the following lemmas, which describe phase transitions of the minimisers of the respective optimisation problems (e_j correspond to the canonical basis of $\mathbb{C}^N$).

Lemma 3.10 (**Phase transition for basis pursuit**)**.** *Let* $N \geq 2$ *and consider the problem* (P_1) *for*

$$A = \begin{pmatrix} \frac{w_1}{\rho_1} & \frac{w_2}{\rho_2} & \cdots & \frac{w_N}{\rho_N} \end{pmatrix} \in \mathbb{C}^{1\times N}, \quad y = 1, \quad \epsilon \in [0,1),$$

where $\rho_j > 0$ *for* $j = 1, ..., N$*. Then the set of solutions is given by*

$$\sum_{j=1}^{N} \left[t_j(1-\epsilon)\frac{\rho_j}{w_j} \right] e_j, \quad s.t. \quad t_j \in [0,1], \sum_{j=1}^{N} t_j = 1 \quad \textit{and} \quad t_j = 0 \quad \textit{if} \quad \rho_j > \min_k \rho_k. \tag{3.1}$$

Proof. Let $\hat{x}_j = x_j w_j \rho_j^{-1}$, then the optimisation problem becomes

$$\mathrm{argmin}_{\hat{x}\in\mathbb{C}^N}\, f(\hat{x}) \coloneqq \sum_{j=1}^{N} \rho_j\, |\hat{x}_j| \;\text{ such that }\; \left| 1 - \sum_{j=1}^{N} \hat{x}_j \right| \leq \epsilon. \tag{3.2}$$

Since $\epsilon < 1$ and the $(\rho_1, \rho_2, ...\rho_N)$ weighted l^1 norm is convex, it follows that the solution must lie on the hypersurface segment $\hat{x}_1 + \hat{x}_2 + ... + \hat{x}_j = 1 - \epsilon$ for $\hat{x}_j \in \mathbb{R}_{\geq 0}$. We now claim that if $\hat{x}$ is a solution of Eq. (3.2), and $\rho_j > \min_k \rho_k$, then $\hat{x}_j = 0$. Suppose for a contradiction that there exists a solution $\hat{x}$ of Eq. (3.2) where $\hat{x}_j > 0$ and $\rho_j > \min_k \rho_k$. Pick any l such that $\rho_l = \min_k \rho_k$, then $\hat{x} + \hat{x}_j(e_l - e_j)$ is feasible with $f(\hat{x} + \hat{x}_j(e_l - e_j)) < f(\hat{x})$, a contradiction. Similarly, if x is of the form given in Eq. (3.1), then $f(\hat{x}) = (1 - \epsilon) \min_k \rho_k$. In particular, the objective function is constant over the set of all such vectors and the result follows. □

Lemma 3.11 (**Phase transition for LASSO**). *Let $N \geq 2$ and consider the problem (P_2) for*

$$A = \lambda \begin{pmatrix} \frac{w_1}{\rho_1} & \frac{w_2}{\rho_2} & \cdots & \frac{w_N}{\rho_N} \end{pmatrix} \in \mathbb{C}^{1 \times N}, \quad y = 1,$$

where $0 < \rho_j < 2$ for $j = 1, ..., N$. Then the set of solutions is given by

$$\left(1 - \frac{\min_k \rho_k}{2}\right) \sum_{j=1}^{N} \frac{\rho_j t_j}{\lambda w_j} e_j, \quad s.t. \quad t_j \in [0,1], \sum_{j=1}^{N} t_j = 1 \quad and \quad t_j = 0 \quad if \quad \rho_j > \min_k \rho_k. \tag{3.3}$$

Proof. Let $\hat{x}_j = x_j \lambda w_j \rho_j^{-1}$, then the optimisation problem becomes $\operatorname{argmin}_{\hat{x} \in \mathbb{C}^N} f(\hat{x}) := |1 - \sum_{j=1}^{N} \hat{x}_j|^2 + \sum_{j=1}^{N} \rho_j |\hat{x}_j|$. It is clear that any optimal solution must be real, and hence we restrict our argument to real $\hat{x}$. Define the 2^N quadrant subdomains $D_{k_1,...,k_N} = \{\hat{x}_j \cdot (-1)^{k_j} > 0\}$ for $k_j \in \{0, 1\}$, and notice that

$$\nabla f(\hat{x}) = \begin{bmatrix} -2(1 - \sum_{j=1}^{N} \hat{x}_j) + (-1)^{k_1} \rho_1 \\ \vdots \\ -2(1 - \sum_{j=1}^{N} \hat{x}_j) + (-1)^{k_N} \rho_N \end{bmatrix}, \quad \text{for } \hat{x} \in D_{k_1,...,k_N}.$$

We first look for stationary points of the objective function in the subdomains $D_{k_1,...,k_N}$. The condition for a stationary point in the interior of such a domain leads to the constraint that $k_1 = k_2 = ... = k_N$. If $k_1 = k_2 = ... = k_N = 1$, then $\nabla f = 0$ leads to the contradiction $\rho_j = 2(\hat{x}_1 + ... + \hat{x}_N) - 2 < 0$. Finally, in the case (and only in the case) of $\rho_1 = \rho_2 = ... = \rho_N$, there is a hypersurface segment of stationary points in $D_{0,0,...,0}$ given by $\hat{x}_1 + ... + \hat{x}_N = 1 - \rho_1/2$ (recall that we assumed $\rho_1 < 2$ so this segment exists).

First, consider the case that $\rho_1 = ... = \rho_N$. Then any optimal solution must either lie on the boundary of some $D_{k_1,...,k_N}$ or on the hypersurface segment $\hat{x}_1 + ... + \hat{x}_N = 1 - \rho_1/2$ in $D_{0,0,...,0}$. A simple case by case analysis now yields that the solutions $\hat{x}$ are given by convex combinations of $(1 - \rho_j/2)e_j$ for $j = 1, ..., N$. Now consider the case that not all of the ρ_j are equal. Then any optimal solution must lie on the boundary of some $D_{k_1,...,k_N}$. A simple case by case analysis now yields that the solutions $\hat{x}$ are given by convex combinations of $(1 - \rho_j/2)e_j$ for j such that $\rho_j = \min_k \rho_k$. Rescaling back to x gives the result. □

Lemma 3.12 (**Phase transition for square-root LASSO**). *Let $N \geq 2$ and consider the problem (P_3) for*

$$A = \lambda \begin{pmatrix} \frac{w_1}{\rho_1} & \frac{w_2}{\rho_2} & \cdots & \frac{w_N}{\rho_N} \end{pmatrix} \in \mathbb{C}^{1 \times N}, \quad y = 1,$$

where $0 < \rho_j < 1$ for $j = 1, ..., N$. Then the set of solutions is given by

$$\sum_{j=1}^{N} \frac{\rho_j t_j}{\lambda w_j} e_j, \quad s.t. \quad t_j \in [0,1], \sum_{j=1}^{N} t_j = 1 \quad and \quad t_j = 0 \quad if \quad \rho_j > \min_k \rho_k. \tag{3.4}$$

Proof. Let $\hat{x}_j = x_j \lambda w_j \rho_j^{-1}$, then the optimisation problem becomes $\operatorname{argmin}_{\hat{x} \in \mathbb{C}^N} f(\hat{x}) := |1 - \sum_{j=1}^{N} \hat{x}_j| + \sum_{j=1}^{N} \rho_j |\hat{x}_j|$. It is clear that any optimal solution must be real and hence we restrict our argument to real $\hat{x}$. The objective function is piecewise affine and since $\rho_j < 1$, the gradient of f is non-vanishing on the interior of any of the domains $D_{k_1,...,k_N} = \{\hat{x}_j \cdot (-1)^{k_j} > 0\}$ for $k_j \in \{0, 1\}$. It follows that the optimal solutions must lie on the boundaries of the domains $D_{k_1,...,k_N}$. A simple case by case analysis shows that the solutions $\hat{x}$ are given by convex combinations of e_j for j such that $\rho_j = \min_k \rho_k$. Rescaling back to x gives the result. □

We will also need the following propositions, which give useful criteria for impossibility results.

Proposition 3.13. *Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem. Suppose that there are two sequences $\{\iota_n^1\}_{n \in \mathbb{N}}$, $\{\iota_n^2\}_{n \in \mathbb{N}} \subset \Omega$ satisfying the following conditions:*

(a) There are sets $S^1, S^2 \subset \mathcal{M}$ and $\kappa > 0$ such that $\inf_{x_1 \in S^1, x_2 \in S^2} d(x_1, x_2) > \kappa$ and $\Xi(\iota_n^j) \subset S^j$ for $j = 1, 2$.

(b) For every $f \in \Lambda$ there is a $c_f \in \mathbb{C}$ such that $|f(\iota_n^j) - c_f| \leq 1/4^n$ for all $n \in \mathbb{N}$ and $j = 1, 2$.

Then, if we consider $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$, we have the following:

(i) For any sequential general algorithm Γ and $M \in \mathbb{N}$, there exists $\iota \in \Omega$ and $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that

$$\operatorname{dist}(\Gamma(\iota), \Xi(\iota)) > \kappa/2 \quad or \quad S_\Gamma(\iota) > M.$$

(ii) If there is an $\iota^0 \in \Omega$ such that for every $f \in \Lambda$ we have that (b) is satisfied with $c_f = f(\iota^0)$, then for any RGA Γ and $p \in [0, 1/2)$, there exists $\iota \in \Omega$ and $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that $\mathbb{P}_\iota(\text{dist}(\Gamma(\iota), \Xi(\iota)) \geq \kappa/2) > p$.

Proof. Without loss of generality, we assume that $\Omega = \{\iota_n^1\}_{n\in\mathbb{N}} \cup \{\iota_n^2\}_{n\in\mathbb{N}}$. Part (ii) follows immediately from a Proposition of (26), so we only prove part (i). Let Γ be a sequential general algorithm and $M \in \mathbb{N}$. We will construct the required $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$ inductively. By Definition 3.3 and the setup of Δ_1-information for Λ, there exists some $f_{k_1} \in \Lambda$ and $n_1 \in \mathbb{N}$ such that for all $\iota \in \Omega$ and for all $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$, we have $f^\Gamma_{\iota,1} = f_{k_1,n_1}$. We set $f_{k_1,n_1}(\iota_m^j) = c_{f_{k_1}}$ for all $m \geq n_1$ and choose $f_{k_1,n_1}(\iota_m^j)$ consistently for $m < n_1$. Again by Definition 3.3 and the setup of Δ_1-information for Λ, it follows that there exists $f_{k_2} \in \Lambda$ and $n_2 \in \mathbb{N}$ (which without loss of generality $\geq n_1$) such that for all $m \geq n_1$, either $\Lambda_\Gamma(\iota_m^j) = \{f_{k_1,n_1}\}$ or $f^\Gamma_{\iota_m^j,2} = f_{k_2,n_2}$. In the latter case, we set $f_{k_2,n_2}(\iota_m^j) = c_{f_{k_2}}$ for all $m \geq n_2$ and choose $f_{k_2,n_2}(\iota_m^j)$ consistently for $m < n_2$. We continue this process for a maximum of M steps up to $f^\Gamma_{\iota_m^j,\min\{M,|\Lambda_\Gamma(\iota_m^j)|\}}$ as follows. At the qth step after defining f_{k_q,n_q}, by Definition 3.3 and the setup of Δ_1-information for Λ, it follows that there exists $f_{k_{q+1}} \in \Lambda$ and $n_{q+1} \in \mathbb{N}$ (which without loss of generality $\geq n_q$) such that for all $m \geq n_q$, either $\Lambda_\Gamma(\iota_m^j) \subset \{f_{k_1,n_1}, ..., f_{k_q,n_q}\}$ or $f^\Gamma_{\iota_m^j,q+1} = f_{k_{q+1},n_{q+1}}$. In the latter case, we set $f_{k_{q+1},n_{q+1}}(\iota_m^j) = c_{f_{k_{q+1}}}$ for all $m \geq n_{q+1}$ and choose $f_{k_{q+1},n_{q+1}}(\iota_m^j)$ consistently for $m < n_{q+1}$. We can then choose the rest of the function values to obtain $\widehat{\Lambda}$.

Given this $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$, suppose for a contradiction that for any $\iota \in \Omega$, $\text{dist}(\Gamma(\iota), \Xi(\iota)) \leq \kappa/2$ and $S_\Gamma(\iota) \leq M$. Without loss of generality, we assume that the above construction is carried out for M steps. It follows that we must have $f(\iota^1_{n_M}) = f(\iota^2_{n_M})$ for all $f \in \widehat{\Lambda}_\Gamma(\iota^1_{n_M})$. By (ii) and (iii) of Definition 3.3, it follows that $\Gamma(\iota^1_{n_M}) = \Gamma(\iota^2_{n_M})$. Let $\epsilon > 0$ be arbitrary. Since $\text{dist}(\Gamma(\iota), \Xi(\iota)) \leq \kappa/2$ for all $\iota \in \Omega$, there exists $s_j \in S^j$ such that $d(\Gamma(\iota^j_{n_M}), s_j) < \kappa/2 + \epsilon$. It follows that

$$\inf_{x_1\in S^1, x_2 \in S^2} d(x_1, x_2) \leq d(s_1, s_2) \leq d(\Gamma(\iota^1_{n_M}), s_1) + d(\Gamma(\iota^2_{n_M}), s_2) < \kappa + 2\epsilon.$$

Since $\epsilon > 0$ was arbitrary, we have $\inf_{x_1\in S^1, x_2\in S^2} d(x_1, x_2) \leq \kappa$, the required contradiction. □

Proposition 3.14. *Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem and $u \geq 2$ be a positive integer. Suppose that there are u sequences $\{\iota_n^j\}_{n\in\mathbb{N}} \subset \Omega$, for $j = 1, ..., u$, satisfying the following conditions:*

(a) There are sets $S^j \subset \mathcal{M}$, for $j = 1, ..., u$, and $\kappa > 0$ such that $\inf_{x_j\in S^j, x_k\in S^k} d(x_j, x_k) > \kappa$ for any $j \neq k$ and $\Xi(\iota_n^j) \subset S^j$ for $j = 1, ..., u$.

(b) For every $f \in \Lambda$, there is a $c_f \in \mathbb{C}$ such that $|f(\iota_n^j) - c_f| \leq 1/4^n$ for all $n \in \mathbb{N}$ and $j = 1, ..., u$.

Then, if we consider $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$, for any halting probabilistic Turing machine $(\Gamma, \mathbb{P})$, $M \in \mathbb{N}$ and $p \in [0, \frac{u-1}{u})$, there exists $\iota \in \Omega$ and $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that $\mathbb{P}\Big(\text{dist}(\Gamma(\iota), \Xi(\iota)) > \kappa/2 \quad \text{or} \quad S_\Gamma(\iota) > M\Big) > p$.

Proof. Without loss of generality, we can assume that $\Omega = \cup_{j=1}^u \{\iota_n^j\}_{n\in\mathbb{N}}$. Let $(\Gamma, \mathbb{P})$ be a halting probabilistic Turing machine and $M \in \mathbb{N}$, $p \in [0, (u-1)/u)$. Suppose for a contradiction that for all $\iota \in \Omega$ and all $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$, we have $\mathbb{P}(\text{dist}(\Gamma(\iota), \Xi(\iota)) > \kappa/2 \text{ or } S_\Gamma(\iota) > M) \leq p$. We will construct the required $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$ inductively. Let $\beta \in (0, 1)$ be such that $(1-\beta)^M - p > 1/u$. Such a β exists since $p \in [0, (u-1)/u)$. Since the Turing machine must halt with probability one, there exists finite sets $K_1, N_1 \subset \mathbb{N}$ such that with probability (w.r.t. $\mathbb{P}$) at least $1-\beta$, for all $\iota \in \Omega$ and for all $\widehat{\Lambda} \in \mathcal{L}^1(\Lambda)$, it holds that $f^\Gamma_{\iota,1} = f_{k_1,n_1}$ for some $k_1 \in K_1$ and $n_1 \in N_1$. We set $f_{k_1,n_1}(\iota_m^j) = c_{f_{k_1}}$ for all $m \geq \max\{n_1 : n_1 \in N_1\}$ and choose $f_{k_1,n_1}(\iota_m^j)$ consistently otherwise.

We continue this process inductively for M steps up to $f^\Gamma_{\iota_m^j,\min\{M,|\Lambda_\Gamma(\iota_m^j)|\}}$ as follows. At the qth step after defining f_{k_q,n_q} for $k_q \in K_q$ and $n_q \in N_q$, it follows (since the Turing machine halts with probability one) that there exists finite sets $K_{q+1}, N_{q+1} \subset \mathbb{N}$ with the following property. Let E_{q+1} be the event that for all $\iota_m^j \in \Omega$ with $m \geq \max\{n : n \in N_1 \cup ... \cup N_q\}$, either $f^\Gamma_{\iota_m^j,q+1} = f_{k_{q+1},n_{q+1}}$, for some $k_{q+1} \in K_{q+1}$ and $n_{q+1} \in N_{q+1}$, or $|\Lambda(\iota_m^j)| \leq q$. Then $\mathbb{P}(E_{q+1} | \cap_{k\leq q} E_k) \geq 1-\beta$. We then set $f_{k_{q+1},n_{q+1}}(\iota_m^j) = c_{f_{k_{q+1}}}$ for all $m \geq \max\{n : n \in N_1 \cup ... \cup N_{q+1}\}$ and choose $f_{k_{q+1},n_{q+1}}(\iota_m^j)$ consistently otherwise. This ensures the existence of K_{q+2} and N_{q+2}. After the Mth step, we can choose the rest of the function values to obtain $\widehat{\Lambda}$.

It follows that for $m \geq \max\{n : n \in N_1 \cup ... \cup N_M\}$, the outputs $\Gamma(\iota_m^j)$ conditional on the event $E_1 \cap ... \cap E_M \cap \{S_\Gamma(\cdot) \leq M\}$ are equal for $j = 1, ..., u$. Since $\inf_{x_j\in S^j, x_k\in S^k} d(x_1, x_2) > \kappa$ for $j \neq k$, it follows that the events $F_j := \{\text{dist}(\Gamma(\iota_m^j), \Xi(\iota_m^j)) \leq \kappa/2\} \cap \{S_\Gamma(\iota_m^j) \leq M\} \cap E_1 \cap ... \cap E_M$, $j = 1, ..., u$, are disjoint. Moreover, using the fact that $\mathbb{P}(A \cap B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cup B)$,

$$\begin{aligned}\mathbb{P}(F_j) &\geq \mathbb{P}(\{\text{dist}(\Gamma(\iota_m^j), \Xi(\iota_m^j)) \leq \kappa/2\} \cap \{S_\Gamma(\iota_m^j) \leq M\}) + \mathbb{P}(E_1 \cap ... \cap E_M) - 1 \\ &\geq \mathbb{P}(\{\text{dist}(\Gamma(\iota_m^j), \Xi(\iota_m^j)) \leq \kappa/2\} \cap \{S_\Gamma(\iota_m^j) \leq M\}) + (1-\beta)^M - 1 \geq (1-\beta)^M - p > 1/u.\end{aligned}$$

But this contradicts the disjointness of the F_j's. □

C. Proof of Theorem 2.

Proof of Theorem 2. We will argue for $m = 1$ and construct such an Ω for this case. The general case of $m > 1$ follows by embedding our construction for $A \in \mathbb{C}^{1\times(N+1-m)}$ in the first row of matrices and vectors of the form

$$\hat{A} = \begin{pmatrix} A & 0 \\ 0 & \alpha I \end{pmatrix}, \quad \hat{y} = (y, 0)^\top, \quad (A, y) \in \Omega,$$

where $I \in \mathbb{C}^{(m-1)\times(m-1)}$ denotes the $(m-1)\times(m-1)$ identity matrix and $\alpha = \alpha(A)$ is chosen such that $\hat{A}\hat{A}^*$ is a multiple of the identity. In particular, such an embedding does not effect the relevant condition numbers (it is straightforward to see that the matrix norm, distance to infeasibility for (P_1) and condition numbers of the mappings are all unchanged). For the classes we consider, the setup of Theorem 2 coincides with the Δ_1-information model discussed in §3.A. In particular, we can use Lemmas 3.10, 3.11 and 3.12 to derive the relevant $x_{s,n}$'s in Eq. (1.7). This means that we can apply Propositions 3.13 and 3.14 with the metric corresponding to the l^2-norm. Recall that for this theorem, we assume that $w_1 = w_2 = ... = w_N = 1$.

Step 1: Proof for (P_1). First, consider the class defined by

$$\Omega_1 = \left\{(A(\gamma_1;\rho), y) : A(\gamma_1;\rho) \coloneqq \gamma_1 \begin{pmatrix} \frac{1}{\rho_1} & \frac{1}{\rho_2} & \cdots & \frac{1}{\rho_N} \end{pmatrix}, y = 1, \rho_j \in [1-2\delta, 1-\delta]\right\},$$

for fixed $\gamma_1 > 10$ and $\delta \in (0, 1/4)$. We choose γ_1 and δ such that

$$\frac{1-\epsilon}{\gamma_1} \cdot \sup_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k} \|\rho_j e_j - \rho_k e_k\|_{l^2} = 3 \cdot 10^{-K}, \tag{3.5}$$

$$\frac{1-\epsilon}{\gamma_1} \cdot \inf_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k} \|\rho_j e_j - \rho_k e_k\|_{l^2} > 2 \cdot 10^{-K}, \tag{3.6}$$

where the e_j denote the canonical basis of $\mathbb{C}^N$. Note that we can ensure $\gamma_1 > 10$ since $\epsilon \leq 1/2$ and $K > 2$. If $\rho_j \in [1-2\delta, 1-\delta)$, for $j = 1, 2$, then by (a simple rescale of) Lemma 3.10,

$$\Xi_{P_1}(A(\gamma_1;(\rho_1, 1-\delta, ..., 1-\delta)), 1) = \frac{(1-\epsilon)\rho_1}{\gamma_1} e_1, \Xi_{P_1}(A(\gamma_1;(1-\delta, \rho_2, 1-\delta, ..., 1-\delta)), 1) = \frac{(1-\epsilon)\rho_2}{\gamma_1} e_2.$$

Since Eq. (3.6) holds, it follows by selecting appropriate sequences ι_n^j for choices of $\rho_j = \rho_j^n \uparrow 1-\delta$ that the conditions of Proposition 3.13 hold for Ω_1 with

$$S^j = \left\{\frac{1-\epsilon}{\gamma_1}\rho e_j : \rho \in [1-2\delta, 1-\delta]\right\}, \quad \kappa = 2 \cdot 10^{-K}. \tag{3.7}$$

Moreover, the condition for part (ii) of Proposition 3.13 also holds with $\iota^0 = (A(\gamma_1;(1-\delta, 1-\delta, ..., 1-\delta)), 1)$.

Now suppose for a contradiction that there exists a (halting) RGA (with input $\iota_{A,\mathcal{S}}$) and $p > 1/2$ that produces a NN ϕ_A such that $\min_{y\in\mathcal{S}_A} \inf_{x^*\in\Xi_{P_1}(A,y)} \|\phi_A(y) - x^*\|_{l^2} \leq 10^{-K}$ holds with probability at least p for all $(A, y) \in \Omega_1$. Then there exists a (halting) RGA, Γ, taking $\iota_{A,\mathcal{S}}$ as input that computes a solution of (P_1) to K correct digits with probability at least p on each input in Ω_1. However, this contradicts Proposition 3.13 (ii). Next, consider the class defined by

$$\Omega_2 = \left\{(A(\gamma_2;\rho), y) : A(\gamma_2;\rho) = \gamma_2 \begin{pmatrix} \frac{1}{\rho_1} & \frac{1}{\rho_2} & \cdots & \frac{1}{\rho_N} \end{pmatrix}, y = 1, \rho_j \in [1-2\delta, 1-\delta], \rho_j \neq \rho_k \text{ if } j \neq k\right\},$$

where $\gamma_2 = \gamma_1/10 > 1$ so that

$$\sup_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k} \|\rho_j e_j - \rho_k e_k\|_{l^2} = 3 \cdot 10^{-K+1} \cdot \frac{\gamma_2}{1-\epsilon}, \tag{3.8}$$

$$\inf_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k} \|\rho_j e_j - \rho_k e_k\|_{l^2} > 2 \cdot 10^{-K+1} \cdot \frac{\gamma_2}{1-\epsilon}. \tag{3.9}$$

By extending the argument above to $u = N + 1 - m = N$ (recall without loss of generality that $m = 1$) sequences and sets S^j defined as in Eq. (3.7), the conditions of Proposition 3.14 hold with $\kappa = 2 \cdot 10^{-K+1}$. Now suppose that there exists a (halting) probabilistic Turing machine $(\Gamma, \mathbb{P})$, $M \in \mathbb{N}$ and $p \in \left[0, \frac{N-m}{N+1-m}\right)$, such that for any $(A, 1) \in \Omega_2$, Γ computes a NN ϕ_A with

$$\mathbb{P}\left(\inf_{x^*\in\Xi_{P_1}(A,y)} \|\phi_A(y) - x^*\|_{l^2} > 10^{1-K} \text{ or the sample size needed to construct } \phi_A > M\right) \leq p.$$

Then there exists a (halting) probabilistic Turing machine that computes a solution of (P_1) to $K-1$ correct digits on each input in Ω_2 with sample size at most M with probability at least $1-p$. However, this contradicts Proposition 3.14.

We now set $\Omega = \Omega_1 \cup \Omega_2$. Note that the negative statements of part (i) and (ii) follow from the above arguments by considering restrictions to Ω_1 and Ω_2 respectively. Hence, we are left with proving the condition number bounds, part (iii) and the positive part of part (ii). First, note that $\text{Cond}(AA^*) = 1$ for any $(A, y) \in \Omega$. For any $(A, y) \in \Omega$, we have $\nu(A, y) = \|A\| \geq 1 = \|y\|_{l^2}$ and hence $C_{\text{FP}}(\Xi_{P_1}, \Omega) \leq 1$. To bound the final condition number, first note that if $\rho_j, \rho_j' \leq 1$, then

$\|\rho-\rho'\|_{l^2} \leq \|\sum_{j=1}^N (\frac{1}{\rho_j}-\frac{1}{\rho'_j})e_j\|_{l^2}$. Let $(A(\gamma_1;\rho),1)\in\Omega_1$, then if $(A(\gamma_1;\rho'),1)\in\Omega_1$ with $\Delta(\rho,\rho') \coloneqq \gamma_1\|\sum_{j=1}^N (\frac{1}{\rho_j}-\frac{1}{\rho'_j})e_j\|_{l^2}$ sufficiently small,

$$\operatorname{dist}(\Xi_{P_1}(A(\gamma_1;\rho'),1),\Xi_{P_1}(A(\gamma_1;\rho),1)) \leq \frac{1-\epsilon}{\gamma_1}\left\|\rho-\rho'\right\|_{l^2}.$$

It follows that

$$\lim_{\beta\downarrow 0}\sup_{\substack{(A(\gamma_1;\rho'),1)\in\Omega_1\\ \Delta(\rho,\rho')\leq\beta}} \frac{\operatorname{dist}(\Xi_{P_1}(A(\gamma_1;\rho'),1),\Xi_{P_1}(A(\gamma_1;\rho),1))}{\Delta(\rho,\rho')} \leq \frac{1-\epsilon}{\gamma_1^2} < 1.$$

A similar argument holds for $(A(\gamma_2;\rho),1)\in\Omega_2$, and hence $\operatorname{Cond}(\Xi_{P_1},\Omega)\leq 1$.

We now prove the positive parts of (ii) and (iii). We begin with (ii) and describe the algorithm informally, noting that the output of the algorithm, $\Gamma(A)$, yields a NN which maps $y=1$ to $\Gamma(A)\in\mathbb{C}^N$. Given an input $(A,y)\in\Omega$, the algorithm first tests the size of $A_{1,1}$ to determine whether $(A,y)\in\Omega_1$ or $(A,y)\in\Omega_2$. Explicitly, we note that $A_{1,1}$ is positive and bounded away from 0. Hence, with one sample from $\iota_{A,\mathcal{S}}$ we can determine $A_{1,1}$ to an accuracy of at least $0.01\cdot A_{1,1}$ and such that, simultaneously, the corresponding approximation of $A_{1,1}^{-1}$ is accurate to at least 10^{-K}. If $(A,y)\in\Omega_1$, then $A_{1,1}\in\gamma_1\cdot[1,2]$ whereas if $(A,y)\in\Omega_2$, then $A_{1,1}\in\gamma_2\cdot[1,2]$. Since $\gamma_2=\gamma_1/10$, this level of accuracy is enough to determine whether $(A,y)\in\Omega_1$ or $(A,y)\in\Omega_2$. Next, if the algorithm determines $(A,y)\in\Omega_1$, it outputs the corresponding approximation of $(1-\epsilon)A_{1,1}^{-1}e_1=\frac{1-\epsilon}{\gamma_1}\rho_1 e_1$ correct to 10^{-K} in the l^2-norm from the sample. Since

$$\sup_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k}\|\rho_j e_j-\rho_k e_k\|_{l^2} = 3\cdot 10^{-K}\cdot\frac{\gamma_1}{1-\epsilon},$$

it follows that $\inf_{x^*\in\Xi_{P_1}(A,y)}\|\Gamma(A)-x^*\|_{l^2}\leq 4\cdot 10^{-K}<10^{-K+1}$. On the other hand, if the algorithm determines $(A,y)=(A(\gamma_2;\rho),y)\in\Omega_2$, then we know that all of the ρ_j are distinct. The algorithm continues to sample $\iota_{A,\mathcal{S}}$ until we determine j such that $\rho_j=\min_k\rho_k$. It then outputs an approximation of $(1-\epsilon)A_{1,j}^{-1}e_j=\Xi_{P_1}(A,y)$ correct to 10^{-K} in the l^2-norm. Such as approximation can be computed using $\iota_{A,\mathcal{S}}$. It then follows that $\|\Gamma(A)-\Xi_{P_1}(A,y)\|_{l^2}\leq 10^{-K}<10^{-K+1}$ and this finishes the proof of (ii). Finally, to prove (iii), note that the arguments above show that, given an input $(A,y)\in\Omega$, we can use one sample ($L=1$) of $\iota_{A,\mathcal{S}}$ to compute an approximation of $A_{1,1}^{-1}$ with error bounded by 10^{-K}. We simply set $\Gamma(A)$ to be $(1-\epsilon)e_1$ multiplied by the approximation of $A_{1,1}^{-1}$. Using Eq. (3.5) and Eq. (3.8), it follows that

$$\inf_{x^*\in\Xi_{P_1}(A,y)}\|\Gamma(A)-x^*\|_{l^2}\leq 3\cdot 10^{-K+1}+10^{-K}<10^{-K+2}.$$

Step 2: Proof for (P_2). This is almost identical step 1 with replacing Lemma 3.10 with Lemma 3.11. The other changes are replacing ϵ with the suitable $\rho_i/2$ in the solution of each LASSO problem, including the additional scale λ in the definition of the matrices (see Lemma 3.11) and choosing $\gamma_1\lambda>10$ and $\delta\in(0,1/4)$ such that (recall that $\lambda\leq 1$)

$$\sup_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k}\frac{1}{\lambda\gamma_1}\cdot\|\rho_j(1-\rho_j/2)e_j-\rho_k(1-\rho_k/2)e_k\|_{l^2}=3\cdot 10^{-K} \tag{3.10}$$

$$\inf_{\rho_j,\rho_k\in[1-2\delta,1-\delta],j\neq k}\frac{1}{\lambda\gamma_1}\cdot\|\rho_j(1-\rho_j/2)e_j-\rho_k(1-\rho_k/2)e_k\|_{l^2}>2\cdot 10^{-K}. \tag{3.11}$$

Let $f(x)=(1-x/2)x$, then for $x\in[0,1]$, $|f'(x)|\leq 1$. It follows that

$$\left\|\sum_{j=1}^N\left(\rho_j(1-\rho_j/2)-\rho'_j(1-\rho'_j/2)\right)e_j\right\|_{l^2}\leq\left\|\sum_{j=1}^N\left(\frac{1}{\rho_j}-\frac{1}{\rho'_j}\right)e_j\right\|_{l^2}.$$

Hence, for sufficiently small δ, for any $(A(\gamma_1;\rho),1)\in\Omega_1$ (recall the additional factor of λ) and ρ' sufficiently close to ρ with $(A(\gamma_1;\rho'),1)\in\Omega_1$,

$$\frac{\operatorname{dist}(\Xi_{P_2}(A(\gamma_1;\rho'),1),\Xi_{P_2}(A(\gamma_1;\rho),1))}{\gamma_1\lambda\left\|\sum_{j=1}^N\left(\frac{1}{\rho_j}-\frac{1}{\rho'_j}\right)e_j\right\|_{l^2}}\leq\frac{1}{\gamma_1^2\lambda^2}<1,$$

with the same bound holding for Ω_2. It follows that $\operatorname{Cond}(\Xi_{P_2},\Omega)\leq 1$.

Step 3: Proof for (P_3). This is almost identical step 2 with replacing Lemma 3.11 with Lemma 3.12, and deleting the corresponding factors of $1-\rho_j/2$. □

D. Details on the numerical example following Theorem 2 of the main text. In this section, we elaborate on the numerical example following Theorem 2 of the main text. The example is a simplification of the arguments found in the proof of Theorem 2 that uses Lemma 3.12 extensively. In our experiment, we use $N_1=2$ and $\lambda=1$, but for full generality, we do not keep these parameters fixed in the discussion below. We assume throughout that $\lambda\in(0,1]$ and $N_1\geq 2$. The experiment is done for real matrices so that the LISTA network architecture can be used.

Let $\gamma > 0$, $\rho \in \mathbb{R}^{N_1} \setminus \{0\}$, and $D \in \mathbb{C}^{N_2+1 \times N_2+1}$ be a unitary discrete cosine transform matrix. Define

$$A(\gamma,\rho) \coloneqq D \begin{pmatrix} a(\gamma,\rho)^\top & 0 \\ 0 & \|a(\gamma,\rho)\|_{l^2} I \end{pmatrix}, \quad \text{where} \quad a(\gamma,\rho)^\top \coloneqq \gamma \begin{pmatrix} \frac{1}{\rho_1} & \cdots & \frac{1}{\rho_{N_1}} \end{pmatrix} \in \mathbb{R}^{1 \times N_1},$$

and $I \in \mathbb{R}^{N_2 \times N_2}$ is the identity matrix. Observe that $A(\gamma,\rho) \in \mathbb{R}^{m \times N}$, with $N = N_1 + N_2$ and $m = N_2 + 1$ and that A has irrational entries (hence only approximations can be used in real-life computations). Furthermore, let $\delta = 1/6$ and $\gamma_K = \frac{\sqrt{2}}{3\lambda}(1-\delta) \cdot 10^K$, where K is the parameter from Theorem 2. Also let

$$y(x^{(2)}) = D \begin{pmatrix} 1 & \|a(\gamma,\rho)\|_{l^2} x^{(2)} \end{pmatrix}^\top \in \mathbb{R}^{N_2+1} \quad \text{for} \quad x^{(2)} \in \mathbb{R}^{N_2} \tag{3.12}$$

and $\Omega_K = \left\{ \left(y(x^{(2)}), A(\gamma_K,\rho) \right) : \rho_j \in [1-2\delta, 1-\delta], x^{(2)} \in \mathbb{R}^{N_2} \right\}$. Next define

$$\rho' = \begin{pmatrix} \rho_1' & 1-\delta & \cdots & \cdots & 1-\delta \end{pmatrix} \quad \text{and} \quad \rho^\sharp = \begin{pmatrix} 1-\delta & \rho_2^\sharp & 1-\delta & \cdots & 1-\delta \end{pmatrix}$$

where $\rho_1', \rho_2^\sharp \in [1-2\delta, 1-\delta)$. We let e_i denote the i'th canonical basis vector for $\mathbb{R}^{N_1}$ and define $x' = \frac{\rho_1'}{\lambda\gamma_K} e_1$ and $x^\sharp = \frac{\rho_1^\sharp}{\lambda\gamma_K} e_2$. For this choice of parameters we have from Lemma 3.12 and the fact that D is unitary that $\left(x', x^{(2)}\right)^\top \in \Xi_3\left(A(\gamma_K,\rho'), y(x^{(2)})\right)$ and $\left(x^\sharp, x^{(2)}\right)^\top \in \Xi_3\left(A(\gamma_K,\rho^\sharp), y(x^{(2)})\right)$.

Observe that we can let $A(\gamma_K,\rho')$ and $A(\gamma_K,\rho^\sharp)$ become arbitrary close by letting $\rho_1', \rho_2^\sharp \uparrow 1-\delta$. We will let $\rho_1' = \rho_2^\sharp$, and notice that for this choice the data $y(x^{(2)})$ are the same for both inputs. However, regardless of the choice of $\rho_1', \rho_2^\sharp \in [1-2\delta, 1-\delta)$ the minimisers for the two problems are bounded away from each other. In particular, we have that

$$\inf_{\rho_1', \rho_2^\sharp \in [1-2\delta, 1-\delta]} \frac{1}{\lambda\gamma_K} \|\rho_1' e_1 - \rho_2^\sharp e_2\|_{l^2} > 2 \cdot 10^K \quad \text{and} \quad \sup_{\rho_1', \rho_2^\sharp \in [1-2\delta, 1-\delta]} \frac{1}{\lambda\gamma_K} \|\rho_1' e_1 - \rho_2^\sharp e_2\|_{l^2} = 3 \cdot 10^K$$

which implies that $10^K < \|(x', x^{(2)})^\top - (x^\sharp, x^{(2)})^\top\|_{l^2} < 10^{K+1}$.

In the numerical experiment, we take $A = A(\gamma_K,\rho')$ with $\rho_1' = 1 - \delta + 2^{-n-1}$, and approximate this matrix with the matrix $A_n = A(\gamma_K,\rho^\sharp)$, where the parameter $\rho_2^\sharp = 1 - \delta + 2^{-n-1}$. This ensures that $\|A - A_n\| \leq 2^n$. For the trained neural network, we used 8000 triples of the form

$$\iota_{A,\mathcal{S},n} = \left\{ \left(y_{k,n}(x_{k,n}^{(2)}), A_n, (x_n^\sharp, x_{k,n}^{(2)})^\top \right) : k = 1, \ldots, 8000, \text{ and } x^{(2)} \text{ is 5-sparse} \right\}. \tag{3.13}$$

for $n = 10, 20, 30$. Note that it is not necessary to make the $x^{(2)}$ component sparse. This is done merely to make the experiment more realistic, as the main usage of the problems (P_j) are for recovery of sparse vectors.

4. Proof of Theorem 3

A roadmap for the proof is as follows. We consider the problem (P_3) and unroll iterations of Chambolle and Pock's primal-dual algorithm (32, 33). These iterations are approximated by NNs in Theorem 5, where we obtain bounds on a rescaled version of the objective function in Eq. (4.9). The assumption of weighted rNSPL then allows us to relate the bounds proven in Theorem 5 to bounds on the distance of the output of the NN to the wanted vector and also, simultaneously, prove stability. This also allows the acceleration to exponential convergence through a restart scheme (with a reweighting at each restart). We begin with the proof of Lemma 1.8, which allows us to consider the approximation matrices A_l in the construction of the NNs. We also state some results from compressed sensing that are needed in our proofs. We then discuss preliminary results on unrolling iterative algorithms for (P_3), which are used in the proof of Theorem 3. When writing out NNs in the proofs, we will use $\xrightarrow{\text{NL}}$ arrows to denote the non-linear maps and $\xrightarrow{\text{L}}$ arrows to denote the affine maps.

A. Some results from compressed sensing.

Proof of Lemma 1.8. Let Δ be a $(\mathbf{s}, \mathbf{M})$ support set and $x \in \mathbb{C}^N$, then

$$\|x_\Delta\|_{l^2} \leq \frac{\rho\|x_{\Delta^c}\|_{l_w^1}}{\sqrt{\xi}} + \gamma\|Ax\|_{l^2} \leq \frac{\rho\|x_{\Delta^c}\|_{l_w^1}}{\sqrt{\xi}} + \gamma\|\hat{A}x\|_{l^2} + \gamma\|\hat{A} - A\|\|x\|_{l^2}. \tag{4.1}$$

Note that $\min_{k=1,\ldots,r} w_{(k)}^2 \sum_{j\in\Delta^c} |x_j|^2 \leq (\sum_{j\in\Delta^c} |x_j|\, w_j)^2 = \|x_{\Delta^c}\|_{l_w^1}^2$ and hence, Eq. (4.1) implies that

$$\|x_\Delta\|_{l^2} \leq \frac{\rho\|x_{\Delta^c}\|_{l_w^1}}{\sqrt{\xi}} + \gamma\|\hat{A}x\|_{l^2} + \gamma\|\hat{A} - A\| \left(\|x_\Delta\|_{l^2} + \frac{\|x_{\Delta^c}\|_{l_w^1}}{\min_{k=1,\ldots,r} w_{(k)}} \right).$$

Rearranging now gives the result. □

The following results are taken from the compressed sensing literature (34).

Lemma 4.1 (rNSPL implies l^1_w distance bound). *Suppose that A has the weighted rNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $0<\rho<1$ and $\gamma>0$. Let $x,z\in\mathbb{C}^N$, then*

$$\|z-x\|_{l^1_w}\leq\frac{1+\rho}{1-\rho}\left(2\sigma_{\mathbf{s},\mathbf{M}}(x)_{l^1_w}+\|z\|_{l^1_w}-\|x\|_{l^1_w}\right)+\frac{2\gamma}{1-\rho}\sqrt{\xi}\|A(z-x)\|_{l^2}. \tag{4.2}$$

Lemma 4.2 (rNSPL implies l^2 distance bound). *Suppose that A has the weighted rNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $0<\rho<1$ and $\gamma>0$. Let $x,z\in\mathbb{C}^N$, then*

$$\|z-x\|_{l^2}\leq\left(\rho+\frac{(1+\rho)\kappa^{1/4}}{2}\right)\frac{\|z-x\|_{l^1_w}}{\sqrt{\xi}}+\left(1+\frac{\kappa^{1/4}}{2}\right)\gamma\|A(z-x)\|_{l^2}. \tag{4.3}$$

B. Preliminary constructions of neural networks. When constructing NNs, we will make use of the following maps from $\mathbb{C}^M$ to $\mathbb{C}^M$, defined for various $M\in\mathbb{N}$ and $\beta\in\mathbb{Q}_{>0}$ by $\psi^0_\beta(x)=\max\{0,1-\beta/\|x\|_{l^2}\}x$, $\quad\psi^1(x)=\min\{1,\|x\|^{-1}_{l^2}\}x$.

Lemma 4.3. *Let $M\in\mathbb{N}$, $\beta\in\mathbb{Q}_{>0}$ and $\theta\in\mathbb{Q}_{>0}$. Then there exists neural networks $\phi^0_{\beta,\theta},\phi^1_\theta\in\mathcal{N}_{\mathbf{D},3,2}$ with $\mathbf{D}=(M,2M,M+1,M)$ such that $\|\phi^0_{\beta,\theta}(x)-\psi^0_\beta(x)\|_{l^2}\leq\theta$ and $\|\phi^1_\theta(x)-\psi^1(x)\|_{l^2}\leq\theta$ for all $x\in\mathbb{C}^M$, and the non-linear maps can be computed from sqrt_θ and finitely many arithmetic operations and comparisons.*

Proof. We deal only with the case of ψ^0_β since the case of ψ^1 is nearly identical. Consider the maps $\phi^0_{\beta,\theta}$:

$$x\xrightarrow{\mathrm{L}}\begin{pmatrix}x\\x\end{pmatrix}\xrightarrow{\mathrm{NL}}\begin{pmatrix}|x_1|^2\\|x_2|^2\\\vdots\\|x_M|^2\\x\end{pmatrix}\xrightarrow{\mathrm{L}}\begin{pmatrix}\sum_{j=1}^M|x_j|^2\\x\end{pmatrix}\xrightarrow{\mathrm{NL}}\begin{pmatrix}\max\left\{0,1-\frac{\beta}{\mathrm{sqrt}_\theta(\|x\|^2_{l^2})}\right\}x\end{pmatrix}\xrightarrow{\mathrm{L}}\max\left\{0,1-\frac{\beta}{\mathrm{sqrt}_\theta(\|x\|^2_{l^2})}\right\}x.$$

The first, third and final arrows are simple affine maps. The second arrow applies pointwise modulus squaring, which can be done using finitely many arithmetic operations. The penultimate arrow applies a non-linear map which can be computed from one application of sqrt_θ and finitely many arithmetic operations and comparisons. The bound $\|\psi^0_\beta(x)-\phi^0_{\beta,\theta}(x)\|_{l^2}\leq\theta$ follows from a simple case by case analysis. □

The final piece of machinery needed is a NN approximation of applying a pointwise version of ψ^0_β.

Lemma 4.4. *Let $s,\theta\in\mathbb{Q}_{>0}$, $w\in\mathbb{Q}^N_{>0}$ and for $\hat{x}\in\mathbb{C}^N$ consider the minimisation problem*

$$\mathrm{argmin}_{x\in\mathbb{C}^N}\ \|x\|_{l^1_w}+s\|x-\hat{x}\|^2_{l^2}. \tag{4.4}$$

Let $\tilde{x}_s(\hat{x})$ denote the solution of Eq. (4.4)*. Then, there exists $\phi_{s,\theta}\in\mathcal{N}_{\mathbf{D},2,1}$ such that*

$$\|\phi_{s,\theta}(\hat{x})-\tilde{x}_s(\hat{x})\|_{l^2}\leq\theta\|w\|_{l^2},\quad\forall\hat{x}\in\mathbb{C}^N \tag{4.5}$$

and $\mathbf{D}=(N,N,N)$. Each affine map in the NN is linear and is an arithmetic function of w. Moreover, the non-linear maps used can be computed from sqrt_θ and finitely many arithmetic operations and comparisons.

Proof. Let $B=\mathrm{diag}(w_1,...,w_N)\in\mathbb{Q}^{N\times N}$ and consider the function $F(y)=\|By\|_{l^1}/(2s)=\|y\|_{l^1_w}/(2s)$. We write the minimisation problem in Eq. (4.4) as $\mathrm{prox}_F(\hat{x})$. Given $y\in\mathbb{C}^N$, we identify $y=(y_1,y_2)^\top\in\mathbb{R}^{2N}$.

First, for $\beta>0$ and $x\in\mathbb{R}^n$ recall that the proximal operator of a multiple of the l^2-norm is

$$\mathrm{prox}_{\beta\|\cdot\|_{l^2}}(x)=\max\{0,1-\beta/\|x\|_{l^2}\}x. \tag{4.6}$$

Thus, for $\beta>0$ we define $\varphi_\beta(y)=(v(y,\beta)*y_1,v(y,\beta)*y_2)^\top$, where $*$ denotes pointwise multiplication and $v(y,\beta)_j=\max\{0,1-\beta/\sqrt{y_{1,j}^2+y_{2,j}^2}\}$ for $j=1,...,N$. The function φ_β simply corresponds to a proximity map of the l^2-norm applied component-wise to the complexified version of y. Using Eq. (4.6), we have

$$\begin{aligned}\mathrm{prox}_F(y)&=\mathrm{argmin}_{z\in\mathbb{C}^N}\ \frac{1}{2s}\|Bz\|_{l^1}+\frac{1}{2}\|z-y\|^2_{l^2}\\&=\mathrm{argmin}_{z\in\mathbb{C}^N}\sum_{j=1}^N\left(\frac{B_{jj}}{2s}\sqrt{z_{1,j}^2+z_{2,j}^2}+\frac{1}{2}\left((z_{1,j}-y_{1,j})^2+(z_{2,j}-y_{2,j})^2\right)\right)\end{aligned}$$

It follows that (in complex vector form) $[\mathrm{prox}_F(y)]_j=\{0,1-\frac{B_{jj}/(2s)}{|y_j|}\}y_j$, for $j=1,\ldots,N$. We can therefore write $\mathrm{prox}_F(y)=B\varphi_{(2s)^{-1}}(B^{-1}y)$. We unroll the computation of $\mathrm{prox}_F(\hat{x})$ via:

$$\hat{x}\xrightarrow{\mathrm{L}}B^{-1}\hat{x}\xrightarrow{\mathrm{NL}}\varphi_{(2s)^{-1}}(B^{-1}\hat{x})\xrightarrow{\mathrm{L}}B\varphi_{(2s)^{-1}}(B^{-1}y).$$

The first arrow is a simple linear map, the second applies $\varphi_{(2s)^{-1}}$ and the third is a linear map. We approximate this by replacing $v(y,\beta)_j y_j$ with $\phi^0_{\beta,\theta}(y_{1,j}+y_{2,j}i)$ (denoting the replacement of $\varphi_{(2s)^{-1}}$ by $\varphi^\theta_{(2s)^{-1}}$) where $\phi^0_{\beta,\theta}$ is the NN from Lemma 4.3 with $M=1$. This clearly gives $\phi_{s,\theta}\in\mathcal{N}_{\mathbf{D},2,1}$, so we need to only bound the error. From Lemma 4.3 we have

$$\left\|\mathrm{prox}_F(\hat{x})-B\varphi^\theta_{(2s)^{-1}}(B^{-1}\hat{x})\right\|_{l^2}=\left\|B\left(\varphi_{(2s)^{-1}}(B^{-1}\hat{x})-\varphi^\theta_{(2s)^{-1}}(B^{-1}\hat{x})\right)\right\|_{l^2}\leq\theta\|w\|_{l^2}.$$

The bound in Eq. (4.5) now follows. □

The following theorem proves that one can construct NNs with objective function bounds. The proof constructs approximations of unrolled iterations of Chambolle and Pock's primal-dual algorithm (32, 33). We have used b to denote part of the inputs of the NNs, instead of y, to avoid a clash of notation with the usual notation for primal-dual iterations (y is used to denote a dual variable). The bounds in part 2 of Theorem 5 will be combined with results from §4.A to construct the families of NNs in Theorem 3.

Theorem 5. *Let $A\in\mathbb{Q}[i]^{m\times N}$ and $\theta\in\mathbb{Q}_{>0}$. Suppose also that $L_A\in\mathbb{Q}_{\geq 1}$ is an upper bound for $\|A\|$, and that $\tau,\sigma\in\mathbb{Q}_{>0}$ are such that $\tau\sigma L_A^2<1$. Let $\lambda\in\mathbb{Q}_{>0}$, $w\in\mathbb{Q}^N_{>0}$ and consider the resulting optimisation problem (P_3). Then there exists an algorithm that constructs a sequence of neural networks $\left\{\phi^A_{n,\lambda},\theta\right\}$ with the following properties:*

1. *(Size) Each $\phi^A_{n,\lambda}:\mathbb{C}^{m+N}\to\mathbb{C}^N$ takes as input data $b\in\mathbb{C}^m$ and an initial guess $x_0\in\mathbb{C}^N$, both of which are completely general. Also, $\phi^A_{n,\lambda}\in\mathcal{N}_{\mathbf{D}_n,3n+1,3}$ with*

$$\mathbf{D}_n=(m+N,\underbrace{2N+m,2(N+m),2N+m+1}_{\text{repeated } n \text{ times}},N).$$

2. *($\mathcal{O}(n^{-1}+n\theta)$ Error Control) Let*

$$C=(1+\|w\|_{l^2}+2\sigma\|A\|\|w\|_{l^2})\sqrt{\frac{\tau+\sigma}{1-\tau\sigma L_A^2}}\sqrt{\frac{\tau+\sigma}{\tau\sigma}}, \tag{4.7}$$

then for any inputs $b\in\mathbb{C}^m$ and $x_0\in\mathbb{C}^N$, there exists a vector $\psi_n(b,x_0)\in\mathbb{C}^N$ with

$$\left\|\psi_n(b,x_0)-\phi^A_{n,\lambda}(b,x_0)\right\|_{l^2}\leq n\theta C \tag{4.8}$$

such that for any $x\in\mathbb{C}^N$ and $\eta\in[0,1]$, it holds that

$$\lambda\|\psi_n(b,x_0)\|_{l^1_w}-\lambda\|x\|_{l^1_w}+\eta\|A\psi_n(b,x_0)-b\|_{l^2}-\|Ax-b\|_{l^2}\leq\frac{1}{n}\left(\frac{\|x-x_0\|^2_{l^2}}{\tau}+\frac{\eta^2}{\sigma}\right). \tag{4.9}$$

Proof. We use the notation $b\in\mathbb{C}^m$ to denote an input vector for our NNs throughout the proof and reserve y to denote dual vectors, consistent with the literature on primal-dual algorithms for saddle point problems.

Step 1: The first step is to consider an equivalent optimisation problem over $\mathbb{R}$ instead of $\mathbb{C}$, and rewrite the problem as a saddle point problem. For $x\in\mathbb{C}^N$, let $x_1=\mathrm{real}(x)$ and $x_2=\mathrm{imag}(x)$ and consider $x=(x_1,x_2)^\top$ as a vector in $\mathbb{R}^{2N}$ (and likewise for the dual variables). With an abuse of notation, we use the same notation for complex $x\in\mathbb{C}^N$ and the corresponding vector in $\mathbb{R}^{2N}$, though it will be clear from the context whether we refer to the complex or real case. We let $c=(\mathrm{real}(b),\mathrm{imag}(b))^\top$. Define the matrices

$$K_1=\begin{pmatrix}\mathrm{real}(A) & -\mathrm{imag}(A)\\ \mathrm{imag}(A) & \mathrm{real}(A)\end{pmatrix}\in\mathbb{R}^{2m\times 2N},\quad K_2=\begin{pmatrix}\mathrm{real}(B) & -\mathrm{imag}(B)\\ \mathrm{imag}(B) & \mathrm{real}(B)\end{pmatrix}\in\mathbb{R}^{2N\times 2N},$$

corresponding to multiplication by the matrices A and $B:=\mathrm{diag}(w_1,\ldots,w_N)$ respectively. Let $\widetilde{F}_1:\mathbb{R}^{2N}\to\mathbb{R}$ be defined by $\widetilde{F}_1(x)=\sum_{j=1}^N\sqrt{(K_2x)_j^2+(K_2x)_{j+N}^2}$ and $\widetilde{F}_3(x)=\lambda\widetilde{F}_1(x)$. Then (P_3) is equivalent to $\min_{x\in\mathbb{R}^{2N}}\ \widetilde{F}_3(x)+\|K_1x-c\|_{l^2}$ and L_A is an upper bound for $\|K_1\|$. The saddle point formulation of the problem is given by

$$\min_{x\in\mathbb{R}^{2N}}\max_{y\in\mathbb{R}^{2m}}\mathcal{L}(x,y):=\langle K_1x,y\rangle+\widetilde{F}_3(x)-f_3^*(y), \tag{4.10}$$

where $f_3^*(y)=\chi_{B_1(0)}(y)+\langle c,y\rangle$, and χ_S denotes the indicator function of a set S, taking the value 0 on S and $+\infty$ otherwise, and $B_1(0)$ denotes the closed l^2 unit ball.

Step 2: We will solve Eq. (4.10) by approximating Chambolle and Pock's primal-dual algorithm (32) (with a shift of updates considered in (33)) with a NN. We will write the iteration as an instance of the proximal point algorithm (35) and gain a non-expansive map in a norm which we relate to the standard Euclidean norm.

We start by setting $x^0 = x_0$ (one of the inputs of the NN) and $y^0 = 0$. Recall that for a convex function h, we have that $x = \text{prox}_h(z)$ if and only if $z \in x + \partial h(x)$, where ∂h denotes the subdifferential of h, see, for example, (36, Prop. B.23). Letting $g = \widetilde{F}_3$ and $f^* = f_3^*$, the exact iterates can be written as

$$\begin{aligned} x^{k+1} &= \text{argmin}_{x\in\mathbb{R}^{2N}}\, g(x) + \frac{1}{2\tau}\|x - (x^k - \tau K_1^* y^k)\|_{l^2}^2 = (I + \tau\partial g)^{-1}(x^k - \tau K_1^* y^k) \\ y^{k+1} &= \text{argmin}_{y\in\mathbb{R}^{2m}}\, f^*(y) + \frac{1}{2\sigma}\|y - (y^k + \sigma K_1(2x^{k+1} - x^k)\|_{l^2}^2 \\ &= (I + \sigma\partial f^*)^{-1}\left[y^k + \sigma K_1(2x^{k+1} - x^k)\right]. \end{aligned} \tag{4.11}$$

Note that the solutions of these proximal mappings are given by Lemmas 4.3 and 4.4 and their proofs, as we describe explicitly below in step 4. The function f^* also depends on the input data b.

Let $z = (x, y)^\top$ and define the matrix

$$M_{\tau\sigma} = \begin{pmatrix} \frac{1}{\tau}I & -K_1^* \\ -K_1 & \frac{1}{\sigma}I \end{pmatrix} \in \mathbb{R}^{2(m+N)\times 2(m+N)},$$

which is positive definite by the assumption $\tau\sigma L_A^2 < 1$ and hence induces a norm denoted by $\|\cdot\|_{\tau\sigma}$. We can write the iterations as (see, for example, (33, Sec. 3))

$$0 \in M_{\tau\sigma}^{-1}\begin{pmatrix} \partial g & K_1^* \\ -K_1 & \partial f^* \end{pmatrix} z^{k+1} + (z^{k+1} - z^k) \Rightarrow z^{k+1} = \left[I + M_{\tau\sigma}^{-1}\begin{pmatrix} \partial g & K_1^* \\ -K_1 & \partial f^* \end{pmatrix}\right]^{-1} z^k.$$

The multi-valued operator

$$M_{\tau\sigma}^{-1}\begin{pmatrix} \partial g & K_1^* \\ -K_1 & \partial f^* \end{pmatrix}$$

is maximal monotone with respect to the inner product induced by $M_{\tau\sigma}$ (35) and hence the iterates are non-expansive in the norm $\|\cdot\|_{\tau\sigma}$. We also have that

$$\|(x,y)^\top\|_{\tau\sigma}^2 \leq \frac{\|x\|_{l^2}^2}{\tau} + \frac{\|y\|_{l^2}^2}{\sigma} + 2L_A\|x\|_{l^2}\|y\|_{l^2} \leq \left(\frac{L_A}{\nu} + \tau^{-1}\right)\|x\|_{l^2}^2 + \left(L_A\nu + \sigma^{-1}\right)\|y\|_{l^2}^2,$$

for any $\nu > 0$ by the generalised AM–GM inequality. Choosing $\nu = \sigma L_A$ and using $\tau\sigma L_A^2 < 1$, we have that

$$\|(x,y)^\top\|_{\tau\sigma}^2 \leq (\tau^{-1} + \sigma^{-1})\|(x,y)^\top\|_{l^2}^2. \tag{4.12}$$

A similar calculation yields that

$$\|(x,y)^\top\|_{l^2}^2 \leq \frac{\tau + \sigma}{1 - \tau\sigma L_A^2}\|(x,y)^\top\|_{\tau\sigma}^2. \tag{4.13}$$

Step 3: Next, we use convergence guarantees proven in (33) to obtain inequalities that closely resemble Eq. (4.9). Define the ergodic averages $X^k = \frac{1}{k}\sum_{j=1}^k x^j, Y^k = \frac{1}{k}\sum_{j=1}^k y^j$. By convexity, the map from $(x^1, y^1)^\top$ to $(X^k, Y^k)^\top$ is also non-expansive in the norm $\|\cdot\|_{\tau\sigma}$. It also holds (see (33) Theorem 1 and remarks) that

$$\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k) \leq \frac{1}{k}\left(\frac{\|x - x_0\|_{l^2}^2}{\tau} + \frac{\|y\|_{l^2}^2}{\sigma}\right), \quad \forall x \in \mathbb{R}^{2N}, \forall y \in \mathbb{R}^{2m}. \tag{4.14}$$

Let y be parallel to $KX^k - c$ such that $\|y\|_{l^2} = \eta \leq 1$, and x be general in Eq. (4.14). This gives

$$\widetilde{F}_3(X^k) - \widetilde{F}_3(x) + \langle K_1X^k - c, y\rangle + \langle c - K_1x, Y^k\rangle \leq \frac{1}{k}\left(\frac{\|x - x_0\|_{l^2}^2}{\tau} + \frac{\eta^2}{\sigma}\right).$$

Since $\|Y^k\|_{l^2} \leq 1$ (otherwise we gain a contradiction in that the left-hand side of Eq. (4.14) is infinite), this implies

$$\widetilde{F}_3(X^k) - \widetilde{F}_3(x) + \eta\|K_1X^k - c\|_{l^2} - \|K_1x - c\|_{l^2} \leq \frac{1}{k}\left(\frac{\|x - x_0\|_{l^2}^2}{\tau} + \frac{\eta^2}{\sigma}\right). \tag{4.15}$$

Step 4: The next step is to unroll the iterations in Eq. (4.11) as (complex-valued) NNs that approximate the X^k. We unroll via the following steps:

$$\begin{pmatrix} X^k \\ x^k \\ y^k \end{pmatrix} \xrightarrow{\text{L}} \begin{pmatrix} X^k \\ x^k - \tau A^* y^k \\ y^k - \sigma A x^k \end{pmatrix} \xrightarrow{\text{NL}} \begin{pmatrix} X^k \\ x^{k+1} \\ y^k - \sigma A x^k \end{pmatrix} \xrightarrow{\text{L}} \begin{pmatrix} X^{k+1} \\ x^{k+1} \\ u^k \end{pmatrix} \xrightarrow{\text{NL}} \begin{pmatrix} X^{k+1} \\ x^{k+1} \\ y^{k+1} \end{pmatrix},$$

with $u^k = y^k + \sigma A(2x^{k+1} - x^k) - \sigma b$. The first arrow is a simple linear map, the second computes $x^{k+1} = \left(I + \tau\lambda\partial F_1^A\right)^{-1}(x^k - \tau A^* y^k)$. The third is an affine map and the final arrow applies ψ^1 to u^k. We now define the approximations $\tilde{Z}^k$ and $\tilde{z}^k$

(of $Z^k = (X^k, Y^k)^\top$ and $z^k = (x^k, y^k)^\top$ respectively) defined by replacing ψ^1 with ϕ_θ^1 (Lemma 4.3) and the computation of $(I + \tau\lambda\partial F_1^A)^{-1}(x^k - \tau A^* y^k)$ with $\phi_{(2\tau\lambda)^{-1},\theta}(\tilde{x}^k - \tau A^* \tilde{y}^k)$ (Lemma 4.4). We initialise the network with $\tilde{x}^0 = x_0$ and $y^0 = 0$. Since the composition of two affine maps is affine, it follows that the mapping from (b, x_0) to $\tilde{X}^n$ can be realised by $\phi_{n,\lambda}^A \in \mathcal{N}_{\mathbf{D}_n, 3n+1, 3}$ with

$$\mathbf{D}_n = (m+N, \underbrace{2N+m, 2(N+m), 2N+m+1}_{\text{repeated } n \text{ times}}, N).$$

Clearly, the sequence of NNs are NNs in the sense of §1.B.1 and can be constructed by an algorithm (see §3.A).

Step 5: Finally, we bound the difference between Z^k and $\tilde{Z}^k$ to deduce Eq. (4.8), and the error bound in the objective function using the inequalities in Step 3. We write $\tilde{x}^k = x^k + e_1^k$, $\tilde{y}^k = y^k + e_2^k$ and clearly have that $e_1^0 = 0$ and $e_2^0 = 0$. We can write $\tilde{x}^{k+1} = \phi_{(2\tau\lambda)^{-1}}(\tilde{x}^k - \tau A^* \tilde{y}^k) + e_3^{k+1}$, with $\|e_3^{k+1}\|_{l^2} \leq \theta \|w\|_{l^2}$ by Lemma 4.4. We also have that $\tilde{y}^{k+1} = \psi^1(\tilde{y}^k + \sigma A(2\tilde{x}^{k+1} - \tilde{x}^k) - \sigma b) + e_4^{k+1}$, with $\|e_4^{k+1}\|_{l^2} \leq \theta$ by Lemma 4.3. Since ψ^1 is non-expansive, it follows that $\tilde{y}^{k+1} = \psi^1(\tilde{y}^k + \sigma A(2(\tilde{x}^{k+1} - e_3^{k+1}) - \tilde{x}^k) - \sigma b) + e_5^{k+1}$, with $\|e_5^{k+1}\|_{l^2} \leq \theta(1 + 2\sigma \|A\| \|w\|_{l^2})$. We can then use the fact that the iterates applied with the exact proximal maps are non-expansive in the norm $\|\cdot\|_{\tau\sigma}$, along with Eq. (4.13) and Eq. (4.12), to conclude that

$$\begin{aligned}
\|X^n - \tilde{X}^n\|_{l^2} &\leq \sqrt{\frac{\tau+\sigma}{1-\tau\sigma L_A^2}} \|Z^n - \tilde{Z}^n\|_{\tau\sigma} \\
&\leq \sqrt{\frac{\tau+\sigma}{1-\tau\sigma L_A^2}} \left[\|Z^{n-1} - \tilde{Z}^{n-1}\|_{\tau\sigma} + \theta\sqrt{\frac{\tau+\sigma}{\tau\sigma}} \left(1 + \|w\|_{l^2} + 2\sigma\|A\|\|w\|_{l^2}\right) \right] \\
&\leq n\theta(1 + \|w\|_{l^2} + 2\sigma\|A\|\|w\|_{l^2}) \sqrt{\frac{\tau+\sigma}{1-\tau\sigma L_A^2}} \sqrt{\frac{\tau+\sigma}{\tau\sigma}}.
\end{aligned}$$

It follows that Eq. (4.8) holds with $\psi_n(b, x_0) = X^n$ and the complex version of Eq. (4.15) implies Eq. (4.9). □

C. Proof of Theorem 3. Step 1: The first step is to derive a bound on the distance between vectors using the square-root LASSO objective function and rNSPL. For any inputs A (the rational approximations $\{A_l\}$), ρ and γ described in the theorem, we can compute, using Lemma 1.8, a positive integer l in finitely many arithmetic operations and comparisons, such that $A_l \in \mathbb{Q}[i]^{m \times N}$ satisfies the rNSPL with constants $(1+\rho)/2 \in (0,1)$, $2\gamma > 0$. Lemmas 4.1 and 4.2 therefore imply that for any pair $z_1, z_2 \in \mathbb{C}^N$ we have

$$\|z_1 - z_2\|_{l_w^1} \leq \frac{3+\rho}{1-\rho} \left(2\sigma_{\mathbf{s},\mathbf{M}}(z_2)_{l_w^1} + \|z_1\|_{l_w^1} - \|z_2\|_{l_w^1}\right) + \frac{8\gamma\sqrt{\xi}}{1-\rho} \|A_l(z_1 - z_2)\|_{l^2}, \tag{4.16}$$

$$\|z_1 - z_2\|_{l^2} \leq \left(\frac{1+\rho}{2} + \frac{(3+\rho)\kappa^{1/4}}{4}\right) \frac{\|z_1 - z_2\|_{l_w^1}}{\sqrt{\xi}} + \left(2 + \kappa^{1/4}\right) \gamma \|A_l(z_1 - z_2)\|_{l^2}. \tag{4.17}$$

Combining these two inequalities, we obtain the bound

$$\begin{aligned}
\|z_1 - z_2\|_{l^2} &\leq \frac{2C_1}{\sqrt{\xi}} \sigma_{\mathbf{s},\mathbf{M}}(z_2)_{l_w^1} + \frac{C_1}{\sqrt{\xi}} \left(\|z_1\|_{l_w^1} - \|z_2\|_{l_w^1}\right) + C_2 \|A_l(z_1 - z_2)\|_{l^2} \\
&\leq \frac{2C_1}{\sqrt{\xi}} \sigma_{\mathbf{s},\mathbf{M}}(z_2)_{l_w^1} + 2C_2 \|A_l z_2 - y\|_{l^2} + \frac{C_1}{\lambda\sqrt{\xi}} \left(\lambda\|z_1\|_{l_w^1} - \lambda\|z_2\|_{l_w^1} + \|A_l z_1 - y\|_{l^2} - \|A_l z_2 - y\|_{l^2}\right),
\end{aligned} \tag{4.18}$$

where the second inequality follows from the fact that $\|A_l(z_1 - z_2)\|_{l^2} \leq \|A_l z_1 - y\|_{l^2} + \|A_l z_2 - y\|_{l^2}$ and we chose a positive rational $\lambda \leq C_1/(C_2\sqrt{\xi})$ (we will specify how small $|\lambda - C_1/(C_2\sqrt{\xi})|$ must be later, and always assume $\lambda \sim C_1/(C_2\sqrt{\xi})$). For notational convenience, we define

$$G(z_1, z_2, y) \coloneqq \lambda\|z_1\|_{l_w^1} - \lambda\|z_2\|_{l_w^1} + \|A_l z_1 - y\|_{l^2} - \|A_l z_2 - y\|_{l^2}, \tag{4.19}$$

the difference between the values of the objective function F_3^A for arguments z_1 and z_2. We also define

$$c(z, y) \coloneqq \frac{2C_1}{C_2\sqrt{\xi}} \cdot \sigma_{\mathbf{s},\mathbf{M}}(z)_{l_w^1} + 2\|A_l z - y\|_{l^2}. \tag{4.20}$$

It follows from Eq. (4.18) and $\lambda \leq C_1/(C_2\sqrt{\xi})$ that

$$\|z_1 - z_2\|_{l^2} \leq \frac{C_1}{\lambda\sqrt{\xi}} \left(c(z_2, y) + G(z_1, z_2, y)\right), \tag{4.21}$$

which also implies the bound $G(z_1, z_2, y) \geq -c(z_2, y)$. These bounds hold for general z_1, z_2 and y.

Step 2: We now apply Theorem 5 using a suitable scaling to define a family of parametrised NNs, which we iterate later in the proof (this corresponds to restarting primal-dual iterations with different parameters). Let $\sigma = \tau \in (4\|A_l\|^{-1}/5, 5\|A_l\|^{-1}/6)$ be positive rational numbers. We can compute such parameters by approximating $\|A_l\|$ via any standard algorithm that

approximates the largest singular value of a rectangular matrix using finitely many arithmetic operations and comparisons. We now use Theorem 5 (with θ specified below) with input $y/(p\beta)$ and $x_0/(p\beta)$ for a given $p \in \mathbb{N}$, and $\beta \in \mathbb{Q}_{>0}$ (which we explicitly define below). Given $\phi_{p,\lambda}^{A_l}(y/(p\beta), x_0/(p\beta))$, Theorem 5 ensures the existence of a vector $\psi_p = \psi_p(y/(p\beta), x_0/(p\beta))$ satisfying

$$\left\| \psi_p \left(\tfrac{y}{p\beta}, \tfrac{x_0}{p\beta} \right) - \phi_{p,\lambda}^{A} \left(\tfrac{y}{p\beta}, \tfrac{x_0}{p\beta} \right) \right\|_{l^2} \leq pC\theta$$

where C is given in Eq. (4.7) and

$$\lambda \|\psi_p\|_{l_w^1} - \lambda \left\| \frac{x}{p\beta} \right\|_{l_w^1} + \left\| A\psi_p - \frac{y}{p\beta} \right\|_{l^2} - \frac{1}{p\beta} \|Ax - y\|_{l^2} \leq \frac{1}{p} \left(\frac{\|x(p\beta)^{-1} - x_0(p\beta)^{-1}\|_{l^2}^2}{\tau} + \frac{1}{\sigma} \right) \tag{4.22}$$

for any $x \in \mathbb{C}^N$ (and we have taken $\eta = 1$ in Eq. (4.9)). Define the map $H_p^\beta : \mathbb{C}^m \times \mathbb{C}^N \to \mathbb{C}^N$ by

$$H_p^\beta(y, x_0) = p\beta \phi_{p,\lambda}^{A_l} \left(\frac{y}{p\beta}, \frac{x_0}{p\beta} \right).$$

The additional scaling factors can be incorporated so that $H_p^\beta \in \mathcal{N}_{\mathbf{D}_p, 3p+1, 3}$. Rescaling Eq. (4.22) yields the existence of a vector $\hat{\psi}_p(y, x_0) \in \mathbb{C}^N$ (where the $\hat{\cdot}$ denotes an appropriate rescaling by multiplying by $p\beta$) such that

$$G\left(\hat{\psi}_p(y, x_0), x, y\right) \leq \frac{5}{4} \left(\frac{\|A_l\|}{p^2\beta} \|x - x_0\|_{l^2}^2 + \|A_l\|\beta \right), \tag{4.23}$$

where we have used $\tau^{-1} = \sigma^{-1} \leq 5\|A_l\|/4$. Moreover, the constant C in Theorem 5 is bounded by

$$C = (1 + \|w\|_{l^2} + 2\sigma\|A_l\|\|w\|_{l^2}) \sqrt{\frac{\tau + \sigma}{1 - \tau\sigma L_A^2}} \sqrt{\frac{\tau + \sigma}{\tau\sigma}} \leq \hat{C}_1(1 + \|w\|_{l^2}), \tag{4.24}$$

for a constant $\hat{C}_1$ that we can explicitly compute. Hence, upon rescaling Eq. (4.8), we arrive at

$$\left\| \hat{\psi}_p(y, x_0) - H_p^\beta(y, x_0) \right\|_{l^2} \leq p^2\theta\beta\hat{C}_1(1 + \|w\|_{l^2}).$$

Using Hölder's inequality, this also implies that

$$\left\| \hat{\psi}_p(y, x_0) - H_p^\beta(y, x_0) \right\|_{l_w^1} \leq p^2\theta\beta\hat{C}_1(1 + \|w\|_{l^2})\|w\|_{l^2}.$$

It follows from the reverse triangle inequality that

$$G\left(H_p^\beta(y, x_0), x, y\right) \leq G\left(\hat{\psi}_p(y, x_0), x, y\right) + p^2\theta\beta\hat{C}_1(1 + \|w\|_{l^2}) \left(\|A_l\| + \lambda\|w\|_{l^2}\right). \tag{4.25}$$

Using this bound in Eq. (4.23), and the fact that $\lambda \lesssim (\gamma\sqrt{\xi})^{-1}$, we can choose $\theta \in \mathbb{Q}_{>0}$ such that

$$\theta^{-1} \lesssim p^2(1 + \|w\|_{l^2}) \max\left\{1, \frac{\lambda\|w\|_{l^2}}{\|A_l\|}\right\} \lesssim p^2(1 + \|w\|_{l^2}) \max\left\{1, \frac{\|w\|_{l^2}}{\|A\|\gamma\sqrt{\xi}}\right\},$$

and, simultaneously,

$$G\left(H_p^\beta(y, x_0), x, y\right) \leq \frac{4}{3} \left(\frac{\|A_l\|}{p^2\beta} \|x - x_0\|_{l^2}^2 + \|A_l\|\beta \right).$$

Combining this with Eq. (4.21), we obtain the key inequality

$$G\left(H_p^\beta(y, x_0), x, y\right) \leq \frac{4C_1^2\|A_l\|}{3p^2\beta\lambda^2\xi} \left[c(x, y) + G(x_0, x, y)\right]^2 + \frac{4}{3}\|A_l\|\beta. \tag{4.26}$$

Step 3: In this step, we specify the choice of p and β. So far, we have not used any information regarding the vectors x and y. Recall that for our recovery theorem, we restricted to pairs (x, y) such that

$$\frac{2C_1}{C_2\sqrt{\xi}} \cdot \sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + 2\|Ax - y\|_{l^2} \leq \delta, \quad \|x\|_{l^2} \leq b_1, \quad \|y\|_{l^2} \leq b_2.$$

Using this, we can choose l larger if necessary such that for any such (x, y), we have the bound

$$c(x, y) \leq \frac{2C_1}{C_2\sqrt{\xi}} \cdot \sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + 2\|Ax - y\|_{l^2} + 2\|A - A_l\|\|x\|_{l^2} \leq 2\delta.$$

The following lemma shows how to choose β and p to gain a decrease in G by a factor of $\upsilon \in (0, 1)$, up to small controllable error terms.

Lemma 4.5. *Let $\upsilon \in (0,1) \cap \mathbb{Q}_{>0}$, $\epsilon_0 \in \mathbb{Q}_{>0}$ and choose $\beta \in \mathbb{Q}_{>0}$ such that $8\|A_l\|\beta = 3\upsilon_0\upsilon(\epsilon_0 + 2\delta)$ for some $\upsilon_0 \in [1,2)$. Then for any x_0 with $G(x_0, x, y) \leq \epsilon_0$ and positive integer $p \geq \left\lceil \frac{8C_1\|A_l\|}{3\upsilon\lambda\sqrt{\xi}\sqrt{(2-\upsilon_0)\upsilon_0}} \right\rceil$ the following bound holds*

$$G\left(H_p^{\beta}(y, x_0), x, y\right) \leq \upsilon\left(2\delta + \epsilon_0\right). \tag{4.27}$$

Proof. The choice of β ensures that $\frac{4}{3}\|A_l\|\beta \leq \frac{\upsilon_0\upsilon}{2}(2\delta+\epsilon_0)$. Using Eq. (4.26), and the fact that $0 \leq c(x,y)+G(x_0,x,y) \leq 2\delta+\epsilon_0$, the bound in Eq. (4.27) therefore holds if

$$\frac{32C_1^2\|A_l\|^2}{9p^2\upsilon_0\upsilon\lambda^2\xi}\left(2\delta+\epsilon_0\right) \leq \frac{(2-\upsilon_0)\upsilon}{2}(2\delta+\epsilon_0).$$

Rearranging and taking the square root gives the result, where the ceiling function ensures p is an integer. □

We denote the choice of β in Lemma 4.5 by $\beta(\upsilon, \epsilon_0)$. Since $8/3 < 3$, we can, by taking l larger and by making λ closer to $C_1/(C_2\sqrt{\xi})$ if necessary, and through an appropriate choice of υ_0, ensure that we can compute (using finitely many arithmetic operations and comparisons) a choice $p(\upsilon) \leq \left\lceil \frac{3C_2\|A\|}{\upsilon} \right\rceil$ such that the conclusion of the lemma holds.

Step 4: We are now ready to construct our NNs. Note first that $G(0,x,y) \leq \|y\|_{l^2} \leq b_2$, for any y in our desired input. Given $n \in \mathbb{N}$, we set $\epsilon_0 = b_2$ and for $j = 2, ..., n$ set $\epsilon_j = \upsilon\left(2\delta + \epsilon_{j-1}\right)$. By summing a geometric series, this implies $\epsilon_n \leq \frac{2\upsilon\delta}{1-\upsilon} + \upsilon^n b_2$. We define $\phi_n(y)$ iteratively as follows. We set $\phi_1(y) = H_{p(\upsilon)}^{\beta(\upsilon,\epsilon_0)}(y, 0)$ and for $j = 2, ..., n$ we set $\phi_j(y) = H_{p(\upsilon)}^{\beta(\upsilon,\epsilon_{j-1})}(y, \phi_{j-1}(y))$. Clearly this algorithmically constructs a NN ϕ_n. We can concatenate (by combining affine maps) the NNs corresponding to the H_p^{β} maps to see that $\phi_n \in \mathcal{N}_{\mathbf{D}_{(n,p)},3np+1,3}$. Moreover, Lemma 4.5 implies the bound $G\left(\phi_n(y), x, y\right) \leq \epsilon_n \leq \frac{2\upsilon\delta}{1-\upsilon} + \upsilon^n b_2$. Combining this with Eq. (4.18),

$$\|\phi_n(y) - x\|_{l^2} \leq \frac{2C_1}{\sqrt{\xi}}\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + 2C_2\|Ax - y\|_{l^2} + 2C_2\|A - A_l\|_{l^2}b_1 + \frac{C_1}{\lambda\sqrt{\xi}}\left(\frac{2\upsilon\delta}{1-\upsilon} + \upsilon^n b_2\right), \tag{4.28}$$

Again, we can apriori choose l and λ to ensure that

$$2C_2\|A - A_l\|_{l^2}b_1 + \frac{C_1}{\lambda\sqrt{\xi}}\left(\frac{2\upsilon\delta}{1-\upsilon} + \upsilon^n b_2\right) \leq C_2\left(\frac{2\upsilon\delta}{1-\upsilon} + \delta + \upsilon^n b_2\right).$$

Applying this bound to Eq. (4.28) yields Eq. (1.16).

Finally, we argue for the error in the weighted l_w^1-norm. Note that since $\rho < 1$, the choice of λ ensures that $\frac{8\gamma\sqrt{\xi}}{1-\rho} < \frac{3+\rho}{1-\rho}\frac{1}{\lambda}$. It follows from Eq. (4.16), using the same argument for the l^2 case, that

$$\|\phi_n(y) - x\|_{l_w^1} \leq \frac{3+\rho}{1-\rho}\left(2\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + \frac{2}{\lambda}\|A_l x - y\|_{l^2} + \frac{1}{\lambda}G\left(\phi_n(y), x, y\right)\right), \tag{4.29}$$

Again, we can apriori adjust l and λ as necessary to obtain the bound

$$\|\phi_n(y) - x\|_{l_w^1} \leq \frac{3+\rho}{1-\rho}\left(2\sigma_{\mathbf{s},\mathbf{M}}(x)_{l_w^1} + \frac{2C_2\sqrt{\xi}}{C_1}\|Ax - y\|_{l^2} + \frac{C_2\sqrt{\xi}}{C_1}\left(\frac{2\upsilon\delta}{1-\upsilon} + \upsilon^n b_2\right) + \delta\frac{C_2\sqrt{\xi}}{C_1}\right),$$

where the final term in brackets corresponds to this final approximation. Simplifying this yields Eq. (1.17). □

To end this section, we provide a brief proof sketch of the bounds in Remark 1.10. The argument is similar to the proof of Theorem 3. We set $\widehat{\phi}_n(y, x_0) = \beta\phi_{n,\lambda}^{A_l}\left(\frac{y}{\beta}, \frac{x_0}{\beta}\right)$, and the arguments in Theorem 3 show that we can choose τ, σ, l and θ_n with $\theta_n^{-1} = \mathcal{O}(n^2)$ such that for any $x, x_0 \in \mathbb{C}^N$ and $y \in \mathbb{C}^m$,

$$G\left(\widehat{\phi}_n(y, x_0), x, y\right) \leq \frac{3}{2}\frac{\|A\|}{n}\left(\frac{\|x - x_0\|_{l^2}^2}{\beta} + \beta\right).$$

If $\|x\|_{l^2} \leq b_1$, then we can choose l such that $2\|A - A_l\|b_2 \leq \|A\|\beta/(2n)\min\{C_1/(C_2\lambda\sqrt{\xi}), 1\}$ and hence Eq. (1.18) follows from Eq. (4.18). Similarly, we can use the corresponding bound in Eq. (4.29) to show Eq. (1.19).

5. Proof of Theorem 4

For the benefit of the reader, we first recall the orthonormal bases used. We then provide coherence estimates which are used to obtain bounds on the number of samples needed, and end this section with the proof of Theorem 4. It will be convenient to sometimes enumerate the vector or tensor elements starting from 0, or negative numbers. That is for $x \in \mathbb{C}^N$ with $d = 1$ we might denote its elements as $x = (x(0), \ldots, x(N-1))$, or $x = (x(-N/2+1), \ldots, x(N/2))$ and for $d > 1$ its $\boldsymbol{k} = (k_1, \ldots, k_d)$'th element is written as $x(\boldsymbol{k})$. It will always be clear from the context, which range of indices we consider. Furthermore, recall from §1.C that we let $N = K^d$ and $K = 2^r$ for $r \in \mathbb{Z}_{\geq 0}$. This is assumed throughout this section.

A. Setup: the relevant orthonormal bases.

Discrete Fourier transform. For a d-dimensional signal $x = \{x(\boldsymbol{t})\}_{t_1,\dots,t_d=0}^{K-1} \in \mathbb{C}^{K\times\cdots\times K}$ we denote its Fourier transform by $[\mathcal{F}x](\boldsymbol{\omega}) = \frac{1}{N^{1/2}} \sum_{t_1,\dots,t_d=0}^{K-1} x(\boldsymbol{t}) \exp(\frac{2\pi i \boldsymbol{\omega}\cdot\boldsymbol{t}}{K}), \quad \boldsymbol{\omega} \in \mathbb{R}^d$. For discrete computations, it is customary to consider this transform at the integers $\boldsymbol{\omega} \in \{-K/2+1,\dots,K/2\}^d$ and let $F^{(d)} \in \mathbb{C}^{K^d\times K^d}$ denote the corresponding matrix so that $F^{(d)}\mathrm{vec}(x) = \{[\mathcal{F}x](\boldsymbol{\omega})\}_{\boldsymbol{\omega}\in\{-K/2+1,\dots,K/2\}^d}$ for a suitable vectorisation $\mathrm{vec}(x)$ of x and ordering of the $\boldsymbol{\omega}$'s. Let

$$\vartheta_{\boldsymbol{\omega}} = \left\{ N^{-1/2} \exp\left(-2\pi i K^{-1} \boldsymbol{\omega}\cdot\boldsymbol{t}\right) : \boldsymbol{t} \in \{0,\dots,K-1\}^d \right\} \subset \mathbb{C}^{K\times\cdots\times K}.$$

Then

$$\left\{ \mathrm{vec}(\vartheta_{\boldsymbol{\omega}}) : \boldsymbol{\omega} \in \{-K/2+1,\dots,K/2\}^d \right\} \tag{5.1}$$

is an orthonormal basis for $\mathbb{C}^{K^d} = \mathbb{C}^N$. Furthermore, recall from §1.C, that we divide the different frequencies into dyadic bands. For $d = 1$ we let $B_1 = \{0, 1\}$ and

$$B_k = \left\{-2^{k-1}+1, ..., -2^{k-2}\right\} \cup \left\{2^{k-2}+1, ..., 2^{k-1}\right\}, \quad k = 2, ..., r.$$

In the general d-dimensional case we set $B_{\mathbf{k}}^{(d)} = B_{k_1} \times ... \times B_{k_d}$ for $\mathbf{k} = (k_1, ..., k_d) \in \mathbb{N}^d$.

Walsh transform.

Definition 5.1. *The Walsh functions $v_n \colon [0,1) \to \{+1,-1\}$ are defined by*

$$v_\omega(z) = (-1)^{\sum_{j=1}^{\infty} (\omega^{(j)}+\omega^{(j+1)}) z^{(j)}}, \quad z \in [0,1), \quad \omega \in \mathbb{Z}_{\geq 0}, \tag{5.2}$$

where $(z^{(i)})_{i\in\mathbb{N}}$ denotes the binary expansion of z (terminating if z is a dyadic rational) and we write $\omega = \sum_{j=1}^{\infty} \omega^{(j)} 2^{j-1}$ for $\omega^{(j)} \in \{0,1\}$. For $\boldsymbol{z} \in [0,1)^d$ and $\boldsymbol{\omega} \in \mathbb{Z}_{\geq 0}^d$, we let $v_{\boldsymbol{\omega}}(\boldsymbol{z}) = v_{\omega_1}(z_1)\cdots v_{\omega_d}(z_d)$.

For $x \in \mathbb{C}^{K\times\cdots\times K}$ and $K = 2^r$ we let its d-dimensional Walsh transform be denoted by

$$[\mathcal{W}x](\boldsymbol{\omega}) = \frac{1}{N^{1/2}} \sum_{t_1,\dots,t_d=0}^{K-1} x(\boldsymbol{t}) v_{\boldsymbol{\omega}}(\boldsymbol{t}/K), \quad \boldsymbol{\omega} \in \{0,\dots,2^r-1\}^d.$$

As in the Fourier case, we let $W^{(d)} \in \mathbb{C}^{N\times N}$ so that $W^{(d)}\mathrm{vec}(x) = \{[\mathcal{W}x](\boldsymbol{\omega})\}_{\boldsymbol{\omega}\in\{0,\dots,K-1\}^d}$ for a suitable vectorisation of x and ordering of the $\boldsymbol{\omega}$'s. We let $\varrho_{\boldsymbol{\omega}} = \{N^{-1/2} v_{\boldsymbol{\omega}}(\boldsymbol{t}/K) : \boldsymbol{t} \in \{0,\dots,K-1\}^d\} \subset \mathbb{C}^{K\times\cdots\times K}$ and note that

$$\left\{ \mathrm{vec}(\varrho_{\boldsymbol{\omega}}) : \boldsymbol{\omega} \in \{0,\dots,K-1\}^d \right\} \tag{5.3}$$

is an orthonormal basis for $\mathbb{C}^N$. As in the Fourier case we recall the frequency bands introduced in §1.C. Let $B_1 = \{0,1\}$ and $B_k = \{2^{k-1},\dots,2^k-1\}$ for $k = 2,\dots,r$ in the one-dimensional case, and $B_{\mathbf{k}}^{(d)} = B_{k_1} \times ... \times B_{k_d}$, $\mathbf{k} = (k_1, ..., k_d) \in \mathbb{N}^d$. Whether the notation refers to the Walsh or Fourier frequency bands will always be clear from the context.

Haar-wavelet transform. On $\mathbb{C}^K$ the Haar wavelet vectors are defined as

$$\psi_{j,p}(i) = \begin{cases} 2^{\frac{j-r}{2}}, & p2^{r-j} \leq i < \left(p+\frac{1}{2}\right)2^{r-j} \\ -2^{\frac{j-r}{2}}, & \left(p+\frac{1}{2}\right)2^{r-j} \leq i < (p+1)2^{r-j} \\ 0, & \text{otherwise,} \end{cases}$$

for $j = 0, ..., r-1$ and $p = 0, ..., 2^j - 1$, and we can define the corresponding scaling vectors as $\varphi_{j,p}(i) = |\psi_{j,p}(i)|$. To simplify the notation we set $\psi_{j,k}^{(0)} = \varphi_{j,k}$ and $\psi_{j,k}^{(1)} = \psi_{j,k}$. For $d > 1$ and $\boldsymbol{q} = (q_1, ..., q_d) \in \{0,1\}^d$, $\boldsymbol{p} = (p_1, \dots, p_d) \in \mathbb{Z}_{\geq 0}^d$ define the tensor product $\psi_{j,\boldsymbol{p}}^{\boldsymbol{q}} = \psi_{j,p_1}^{(q_1)} \otimes ... \otimes \psi_{j,p_d}^{(q_d)}$. Splitting these tensors by scale

$$C_1 = \{\mathrm{vec}(\psi_{0,0}^{\boldsymbol{q}}) : \boldsymbol{q} \in \{0,1\}^d\}, \quad C_j = \{\mathrm{vec}(\psi_{j-1,\boldsymbol{p}}^{\boldsymbol{q}}) : \boldsymbol{q} \in \{0,1\}^d \backslash \{0\}, p_k = 0, ..., 2^{j-1}-1\},$$

for $j = 2, ..., r$, we get that $C_1 \cup \cdots \cup C_r$ is an orthonormal basis for $\mathbb{C}^N$. Next, let the vectors in $C_1 \cup \cdots \cup C_r$, form the rows of a matrix $\Phi \in \mathbb{C}^{N\times N}$. The matrix Ψ is called the *discrete wavelet transform* (DWT) matrix, and its inverse Ψ^{-1} is called the *inverse discrete wavelet transform* (IDWT) matrix. Notice that since $C_1 \cup \cdots \cup C_r$ is an orthonormal basis, we have the relation $\Psi^{-1} = \Psi^*$.

B. Uniform recovery guarantees and coherence estimates. We express $U = \left[U^{(\mathbf{k},j)}\right]_{\mathbf{k}=1,j=1}^{\|\mathbf{k}\|_{l^\infty}\leq r,r}$ in block form, where the entries in each $U^{\mathbf{k},j}$ consist of the inner products $\langle \varphi, \rho_{\boldsymbol{\omega}} \rangle$ for $\varphi \in C_j$ and where $\rho_{\boldsymbol{\omega}}$ is an element in either Eq. (5.1) or Eq. (5.3) with $\boldsymbol{\omega} \in B_{\mathbf{k}}^{(d)}$, depending on whether we consider Fourier or Walsh sampling. For this decomposition we define local coherence as follows.

Definition 5.2. *Let $U = \left[U^{(\mathbf{k},j)}\right]_{\mathbf{k}=1,j=1}^{\|\mathbf{k}\|_{l^\infty}\leq r,r}$ be defined as above. Then the $(\mathbf{k},j)$th local coherence of U is*

$$\mu(U^{\mathbf{k},j}) = \left|B_{\mathbf{k}}^{(d)}\right| \max_{p,q} |(U^{\mathbf{k},j})_{pq}|^2, \quad \text{where } \left|B_{\mathbf{k}}^{(d)}\right| \text{ is the cardinality of } B_{\mathbf{k}}^{(d)}.$$

Recall from Definition 1.6, that for an $(\mathbf{s},\mathbf{M})$-sparse vector, $s = s_1 + \ldots + s_r$ denotes the total sparsity. Furthermore, $m = \sum_{\mathbf{k}=1}^{\|\mathbf{k}\|_{l^\infty}\leq r} m_{\mathbf{k}}$ denotes the total number of samples in an $(\mathbf{N},\mathbf{m})$-multilevel subsampling scheme. The following shows that to use Theorem 3, we need to bound the local coherences of U.

Proposition 5.3 ((37, Thm. 13.12))**.** *Let $\epsilon_{\mathbb{P}} \in (0,1)$, $(\mathbf{s},\mathbf{M})$ be local sparsities and sparsity levels respectively with $2 \leq s \leq N$, and consider the $(\mathbf{N},\mathbf{m})$-multilevel subsampling scheme to form a subsampled unitary matrix A as in Definitions 1.11 and 1.12. Let*

$$t_j = \min\left\{ \left\lceil \frac{\xi(\mathbf{s},\mathbf{M},w)}{w_{(j)}^2} \right\rceil, M_j - M_{j-1} \right\}, \quad j = 1,...,r, \tag{5.4}$$

and suppose that

$$m_k \gtrsim \mathcal{L}' \cdot \sum_{j=1}^{r} t_j \mu(U^{k,j}), \quad k = 1,...,l \tag{5.5}$$

where $\mathcal{L}' = r \cdot \log(2m) \cdot \log^2(t) \cdot \log(N) + \log(\epsilon_{\mathbb{P}}^{-1})$. Then with probability at least $1-\epsilon_{\mathbb{P}}$, A satisfies the weighted rNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $\rho = 1/2$ and $\gamma = \sqrt{2}$.

The following bound the local coherences of U, with $\mathcal{M}_{\mathcal{F}}(\mathbf{s},\mathbf{k})$ and $\mathcal{M}_{\mathcal{W}}(\mathbf{s},\mathbf{k})$ defined in Eq. (1.20) and Eq. (1.21).

Lemma 5.4 (Coherence bound for Fourier case). *Consider the d-dimensional Fourier–Haar–wavelet matrix with blocks $U^{\mathbf{k},j}$, then the local coherences satisfy*

$$\mu(U^{\mathbf{k},j}) \lesssim 2^{-2(j-\|\boldsymbol{k}\|_{l^\infty})_+} \prod_{i=1}^{d} 2^{-|k_i - j|}, \tag{5.6}$$

where for $t \in \mathbb{R}$, $t_+ = \max\{0,t\}$. It follows that

$$\sum_{j=1}^{r} s_j \mu(U^{\mathbf{k},j}) \lesssim \sum_{j=1}^{\|\mathbf{k}\|_{l^\infty}} s_j \prod_{i=1}^{d} 2^{-|k_i-j|} + \sum_{j=\|\mathbf{k}\|_{l^\infty}+1}^{r} s_j 2^{-2(j-\|\mathbf{k}\|_{l^\infty})} \prod_{i=1}^{d} 2^{-|k_i-j|} = \mathcal{M}_{\mathcal{F}}(\mathbf{s},\mathbf{k}). \tag{5.7}$$

Proof. From the one-dimensional case treated in (38, See proof of Lem. 1), we have

$$\left| \left[\mathcal{F}\psi_{j,p}^{(1)}\right](\omega) \right|^2 \lesssim \begin{cases} 2^{-k}2^{-|k-j|}, & \text{if } j \leq k, \\ 2^{-k}2^{-3|k-j|}, & \text{otherwise} \end{cases},$$

We proceed by showing that $|[\mathcal{F}\psi_{j,p}^{(0)}](\omega)|^2 \lesssim 2^{-k}2^{-|k-j|}$ in the one-dimensional case, before considering d dimensions. Let $\omega \neq 0$ correspond to a frequency in B_k, $j \in \{0,...,r-1\}$ and $p \in \{0,...,2^j-1\}$. Then

$$\left[\mathcal{F}\psi_{j,p}^{(0)}\right](\omega) = 2^{\frac{j}{2}-r} e^{2\pi i \omega p 2^{-j}} \sum_{t=0}^{2^{r-j}-1} e^{2\pi i \omega t 2^{-r}} = 2^{\frac{j}{2}-r} e^{2^{1-j}\pi i \omega p} \frac{1 - e^{2\pi i \omega 2^{-j}}}{1 - e^{2\pi i \omega 2^{-r}}}.$$

A simple application of the double angle formula then yields

$$\left| \left[\mathcal{F}\psi_{j,p}^{(0)}\right](\omega) \right| \lesssim 2^{\frac{j}{2}-r} \frac{\left|\sin(\pi\omega 2^{-j})\right|}{\left|\sin(\pi\omega 2^{-r})\right|} = \frac{2^{\frac{j}{2}}}{|\omega|} \frac{\left|\omega 2^{-r}\right|}{\left|\sin(\pi\omega 2^{-r})\right|} \left|\sin(\pi\omega 2^{-j})\right| \lesssim 2^{\frac{j}{2}-k} \left|\sin(\pi\omega 2^{-j})\right|,$$

where the second inequality follows from $|\omega 2^{-r}| \leq 1/2$ and $2^k \lesssim |\omega|$. If $k > j$, this implies $|[\mathcal{F}\psi_{j,p}^{(0)}](\omega)|^2 \lesssim 2^{-k}2^{-|k-j|}$. If $k \leq j$, we use that $|\sin(\pi t)| \leq \pi|t|$, $\forall t \in \mathbb{R}$ to get $\left|\sin(\pi\omega 2^{-j})\right| \lesssim 2^{k-j}$. Hence,

$$\left| \left[\mathcal{F}\psi_{j,p}^{(0)}\right](\omega) \right|^2 \lesssim 2^{j-2k}2^{2k-2j} = 2^{-j} = 2^{-k}2^{-|k-j|}.$$

If $\omega = 0$ then by definition we have $|[\mathcal{F}\psi_{j,p}^{(0)}](\omega)|^2 \lesssim 2^{-j} = 2^{-k}2^{-|k-j|}$ and hence this bound still holds.

We now consider the general d-dimensional case. The above computations give that

$$\mu(U^{(\mathbf{k},1)}) \lesssim 2^{\sum_{i=1}^d k_i} \max_{\boldsymbol{q}\in\{0,1\}^d} \prod_{i=1}^d \max_{w\in B_{k_i}} \left| \left[\mathcal{F}\psi_{0,0}^{(q_i)}\right](\omega)\right|^2 \lesssim \prod_{i=1}^d 2^{-|k_i-1|}.$$

Similarly for $j>1$

$$\mu(U^{(\mathbf{k},j)}) \lesssim 2^{\sum_{i=1}^d k_i} \max_{\boldsymbol{q}\in\{0,1\}^d\setminus\{0\}} \prod_{i=1}^d \max_{w\in B_{k_i}} \max_{p_i\in\{0,\dots,2^{j-1}-1\}} \left|\left[\mathcal{F}\psi_{j-1,p_i}^{(q_i)}\right](\omega)\right|^2 \lesssim \max_{\boldsymbol{q}\in\{0,1\}^d\setminus\{0\}} \prod_{i=1}^d 2^{-|k_i-j|-2q_i(j-k_i)_+}.$$

The maximum value of this estimate is obtained when the non-zero component of $\boldsymbol{q}$ corresponds to the maximum value of k_i. This gives precisely Eq. (5.6). □

Before proceeding with the Walsh–Haar–wavelet case, we recall the following lemma (39).

Lemma 5.5. *Let ω and $j\geq 0$ be integers so that $2^j\leq\omega<2^{j+1}$ and let $\Delta_k^j=[k2^{-j},(k+1)2^{-j})$ for $k\in\mathbb{Z}_{\geq 0}$. Then v_ω is constant on each of the intervals Δ_k^{j+1}, $k\in\{0,\dots,2^{j+1}-1\}$. Each of the intervals Δ_k^j can be decomposed into the intervals Δ_{2k}^{j+1} and Δ_{2k+1}^{j+1}, where v_ω is equal to 1 on exactly one of them and equal to -1 on the other. When $\omega=0$, we have $v_\omega\equiv 1$.*

Lemma 5.6 (**Coherence bound for Walsh case**). *Consider the d-dimensional Walsh–Haar–wavelet matrix with blocks $U^{(\mathbf{k},j)}$, then the local coherences satisfy*

$$\mu(U^{(\mathbf{k},j)}) \lesssim \begin{cases} \displaystyle\prod_{i=1}^d 2^{-|k_i-j|} & \text{if } k_i\leq j \text{ for } i=1,\dots,d \text{ with at least one equality,} \\ 0 & \text{otherwise} \end{cases}. \tag{5.8}$$

It follows that

$$\sum_{j=1}^r s_j \mu(U^{(\mathbf{k},j)}) \lesssim s_{\|\mathbf{k}\|_{l^\infty}} \prod_{i=1}^d 2^{-|k_i-\|\mathbf{k}\|_{l^\infty}|} = \mathcal{M}_{\mathcal{W}}(\mathbf{s},\mathbf{k}). \tag{5.9}$$

Proof. We begin with some computations in the one-dimensional case. Let $I_{j,p}=\{p2^{r-j},\dots,(p+1)2^{r-j}-1\}$. We recall that that $\operatorname{supp}(\psi_{j,p}^{(0)})=\operatorname{supp}(\psi_{j,p}^{(1)})=I_{j,p}$. Using Lemma 5.5 it is clear that for $2^m\leq\omega<2^{m+1}$, ϱ_ω is constant on $I_{m+1,k}$, for $k\in\{0,\dots,2^{m+1}-1\}$ and that for any pair $I_{m+1,2t}$, $I_{m+1,2t+1}$, ϱ_ω changes sign. For $\omega=0$, we have that ϱ_ω is all constant. Keeping track of the supports gives the relations

$$\left|\left\langle\psi_{j,p}^{(0)},\varrho_\omega\right\rangle\right| = \begin{cases} 2^{-j/2} & \text{if } \omega<2^j \\ 0 & \text{otherwise} \end{cases}, \quad \text{and} \quad \left|\left\langle\psi_{j,p}^{(1)},\varrho_\omega\right\rangle\right| = \begin{cases} 2^{-j/2} & \text{if } 2^j\leq\omega<2^{j+1} \\ 0 & \text{otherwise} \end{cases}.$$

In particular, we can rewrite this as $|\langle\psi_{j,p}^{(0)},\varrho_\omega\rangle|=2^{-j/2}=2^{-k/2}2^{-(j-k)/2}$ if $\omega\in B_k$, $k\leq j$ and 0 otherwise, and note that $|\langle\psi_{j,p}^{(1)},\varrho_\omega\rangle|=2^{-j/2}$ if $\omega\in B_{j+1}$ and 0 otherwise.

Turning to the general d-dimensional case. The above computations immediately give that $\mu(U^{(\mathbf{k},1)})\lesssim\prod_{i=1}^d\delta_{k_i,1}$, where $\delta_{i,j}$ is the Kronecker-delta. Similarly for $j>1$

$$\begin{aligned}\mu(U^{(\mathbf{k},j)}) &= \left|B_{\mathbf{k}}^{(d)}\right| \max_{\boldsymbol{q}\in\{0,1\}^d\setminus\{0\}} \prod_{i=1}^d \max_{\omega_i\in B_{k_i}} \max_{p_i\in\{0,\dots,2^{j-1}-1\}} \left|\langle\varrho_{\omega_i},\psi_{j-1,p_i}^{(q_i)}\rangle\right|^2 \\ &\lesssim 2^{\sum_{i=1}^d k_i} \max_{\boldsymbol{q}\in\{0,1\}^d\setminus\{0\}} \prod_{i=1}^d \left(\delta_{q_i,0}\delta_{k_i<j}2^{-k_i-|k_i-j|}+\delta_{q_i,1}\delta_{k_i,j}2^{-k_i}\right) \\ &\lesssim \max_{\boldsymbol{q}\in\{0,1\}^d\setminus\{0\}} \prod_{i=1}^d \left(\delta_{q_i,0}\delta_{k_i<j}2^{-|k_i-j|}+\delta_{q_i,1}\delta_{k_i,j}\right).\end{aligned}$$

This estimate is zero unless $k_i\leq j$ and at least one of the k_i is equal to j. In this case the maximum corresponds to $q_i=1$ if $k_i=j$ and $q_i=0$ otherwise. This gives precisely Eq. (5.8). □

C. Proof of Theorem 4. For the benefit of the reader, we recall that $A = P_{\mathcal{I}} D V \Psi$. We apply Proposition 5.3, noting that the t_j in Eq. (5.4) satisfy

$$t_j \lesssim \frac{\xi(\mathbf{s},\mathbf{M},w)}{w_{(j)}^2} \leq s_j \cdot \kappa(\mathbf{s},\mathbf{M},w), \qquad t \lesssim s \cdot \kappa(\mathbf{s},\mathbf{M},w). \tag{5.10}$$

Therefore $\sum_{j=1}^r t_j \mu(U^{\mathbf{k},j}) \lesssim \kappa(\mathbf{s},\mathbf{M},w) \sum_{j=1}^r s_j \mu(U^{\mathbf{k},j})$. Combining with Eq. (5.10), note that Eq. (5.5) holds if

$$m_{\mathbf{k}} \gtrsim \kappa(\mathbf{s},\mathbf{M},w) \cdot \left(\sum_{j=1}^r s_j \mu(U^{\mathbf{k},j}) \right) \cdot \mathcal{L}, \quad \text{where} \tag{5.11}$$

$$\mathcal{L} = \frac{r \cdot \log(2m)}{\log(2)} \cdot \log^2\left(s \cdot \kappa(\mathbf{s},\mathbf{M},w)\right) \cdot \log(N) + \log(\epsilon_{\mathbb{P}}^{-1}) = d \cdot r^2 \cdot \log(2m) \cdot \log^2\left(s \cdot \kappa(\mathbf{s},\mathbf{M},w)\right) + \log(\epsilon_{\mathbb{P}}^{-1}),$$

since $N = 2^{r \cdot d}$. In the Fourier sampling case, by Lemma 5.4, Eq. (5.11) holds if Eq. (1.22) holds. Similarly, in the Walsh sampling case, by Lemma 5.6, Eq. (5.11) holds if Eq. (1.23) holds. By Proposition 5.3, with probability at least $1 - \epsilon_{\mathbb{P}}$, A satisfies the weighted rNSPL of order $(\mathbf{s},\mathbf{M})$ with constants $\rho = 1/2$ and $\gamma = \sqrt{2}$. The conclusion of Theorem 3 then holds for the uniform recovery of the Haar wavelet coefficients $x = \Psi c \in \mathbb{C}^N$.

For the final part, we use Theorem 3. The only difference is that we have to compose the NNs with (an approximation of) the matrix Ψ^* to recover approximations of c from approximations of $x = \Psi c$. Recall that

$$\mathcal{Z} = \max\left\{1, \frac{\max_{j=1,\dots,r} w_{(j)} \sqrt{(M_j - M_{j-1})}}{\sqrt{\xi(\mathbf{s},\mathbf{M},w)}}\right\}$$

and set $n_0 = \left\lceil \log\left(\delta^{-1}\mathcal{Z}\right) \kappa^{1/4} \mathcal{Z} \right\rceil$. Let p be as in Theorem 3 and let $n_1 \in \mathbb{Z}_{\geq 0}$ such that $n_0 = n_1 p + n_2$ for $n_1 \in \{0,\dots,p-1\}$ (the n from the statement of the theorem corresponds to $n_1 p$). Set $\phi(y) = \Psi^* \left[\phi_{n_1}(y,0)\right]$, where ϕ_{n_1} denotes the NN from Theorem 3 with $b_1 = 1$, $b_2 = \|A\| + \delta$ and $\upsilon = e^{-1}$. Strictly speaking, we need to approximate $\|A\|$ and e^{-1}, and also apply a rational approximation of the matrix Ψ^* instead of Ψ^*, but we have avoided this extra notational clutter (the associated approximation errors can be made smaller than $\kappa^{1/4}\delta$ since the vectors we apply the matrix to are uniformly bounded). Now suppose that $y = P_{\mathcal{I}} D V c + e \in \mathcal{J}(\delta,\mathbf{s},\mathbf{M},w)$, and notice that for $\|c\|_{l^2} \leq 1$ we have that $\|y\|_{l^2} \leq \|A\| + \|e\|_{l^2} \leq b_2$ since Ψ is an isometry. Then, since $C_1, C_2 \sim \kappa^{1/4}$ (using that $\kappa \geq 1$), Eq. (1.16) implies that

$$\|\phi(y) - c\|_{l^2} = \|\phi_{n_0}(y,0) - \Psi c\|_{l^2} \lesssim \kappa^{1/4}\delta + b_2 \kappa^{1/4} e^{-n_1}.$$

The theorem follows if we can prove that $b_2 e^{-n_1} \lesssim \delta$.

Let $\mathbf{t}$ be as in Eq. (5.4) and let $\Delta_1, \Delta_2, \dots$ be a partition of $\{1,\dots,N\}$ such that each support set is $(\mathbf{t},\mathbf{M})$-sparse. We can choose such as partition with at most

$$\max_{j=1,\dots,r} \left\lceil \frac{M_j - M_{j-1}}{t_j} \right\rceil \lesssim \max_{j=1,\dots,r} \left\lceil \frac{M_j - M_{j-1}}{\min\{\xi(\mathbf{s},\mathbf{M},w)/w_{(j)}^2, M_j - M_{j-1}\}} \right\rceil$$

sets. The proof of Proposition 5.3 shows that A satisfies the RIPL of order $(\mathbf{t},\mathbf{M})$ and hence for any $x \in \mathbb{C}^N$,

$$\|Ax\|_{l^2} \leq \sum_i \|A(x_{\Delta_i})\|_{l^2} \lesssim \sum_i \|x_{\Delta_i}\|_{l^2} \lesssim \max_{j=1,\dots,r} \sqrt{\left\lceil \frac{M_j - M_{j-1}}{\min\{\xi(\mathbf{s},\mathbf{M},w)/w_{(j)}^2, M_j - M_{j-1}\}} \right\rceil} \|x\|_{l^2},$$

where we have used Hölder's inequality in the last step. It follows that $\|A\| \lesssim \mathcal{Z}$ and hence that $p \lesssim \kappa^{1/4}\mathcal{Z}$ and $b_2 \lesssim \mathcal{Z}$. This implies that $n_1 \gtrsim \log\left(\delta^{-1}\mathcal{Z}\right)$ and $b_2 e^{-n_1} \lesssim \delta$, completing the proof. □

Fig. S1. The different sampling regions used for the sampling patterns for Fourier (left, $r = 3$) and Walsh (right, $r = 4$). The axis labels correspond to the frequencies in each band and the annular regions are shown as the shaded greyscale regions.

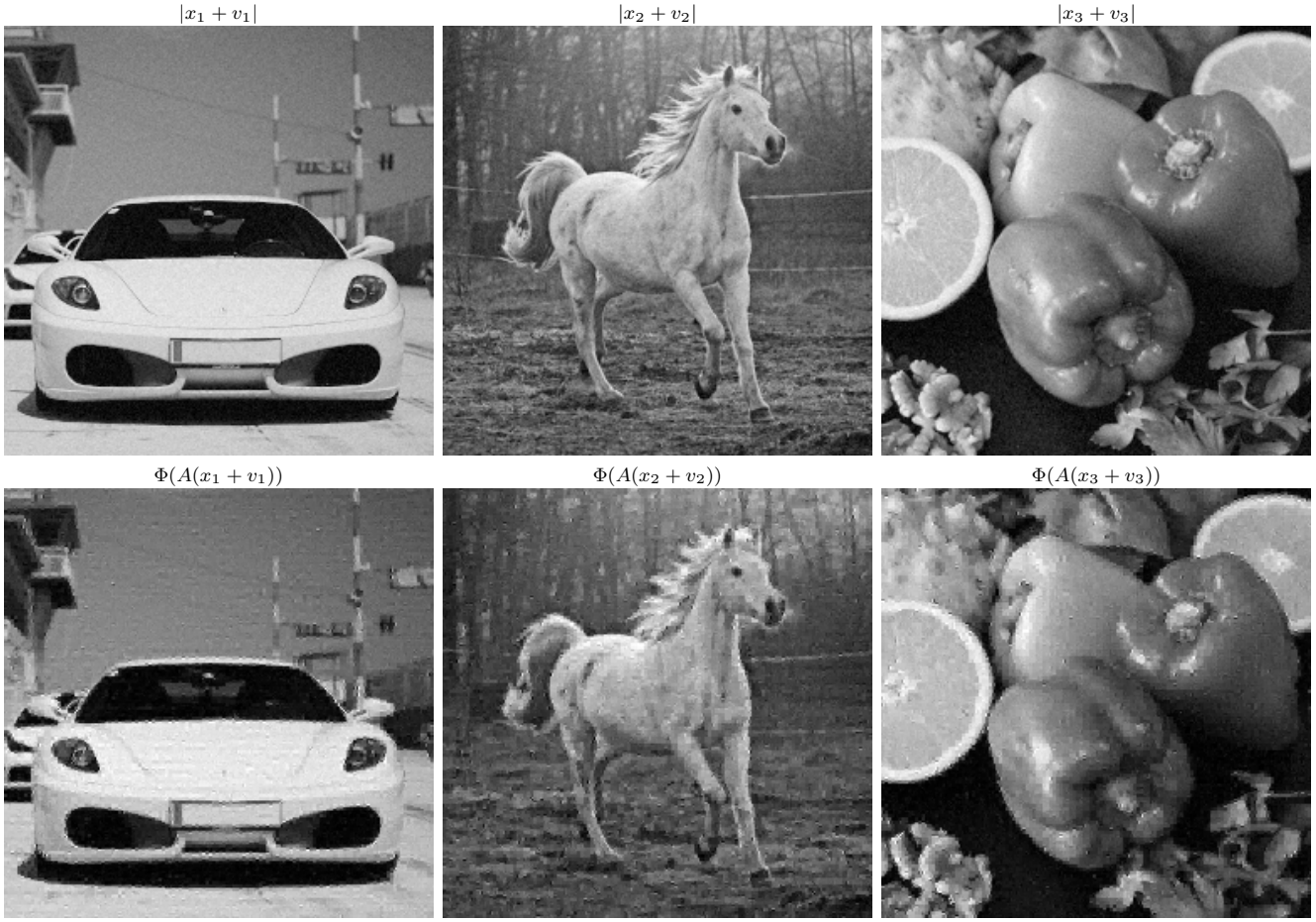

Fig. S2. (**FIRENET withstands worst-case perturbations and generalises well**). To show that FIRENET generalises well and is stable, we consider three different images $x_j, j = 1, 2, 3$. For each image x_j we compute a perturbation v_j meant to simulate worst-case effect for a FIRENET Φ with $n = 5$ and $p = 5$. The first row shows the perturbed images $x_j + v_j$, whereas the second row shows the FIRENET reconstructions from data $A(x_j + v_j)$. Here $A \in \mathbb{C}^{m \times N}$ is a subsampled discrete Fourier transform with $m/N = 0.25$ and $N = 256^2$. The perturbations v_j have magnitude $\|Av_j\|_{l^2} / \|Ax_j\|_{l^2} \geq 0.05$ in the measurement domain.

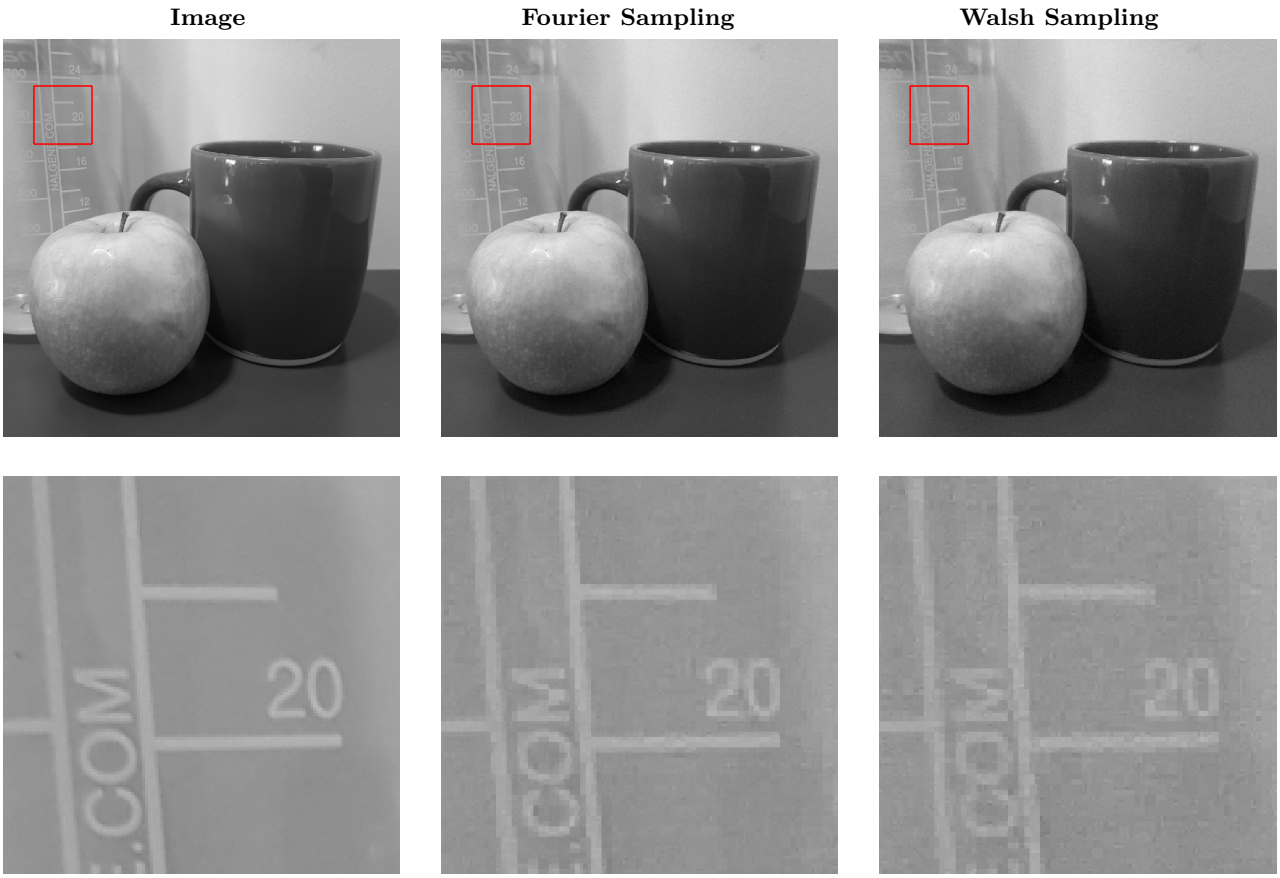

Fig. S3. Left: The true image. Middle: Reconstruction from noisy Fourier measurements. Right: Reconstruction from noisy Walsh measurements. Both images were reconstructed using only a 15% sampling rate according to the sampling patterns in Figure S1 and $n = p = 5$. The top row shows the full image and the bottom row shows a zoomed in section (corresponding to the red boxes in the top row).

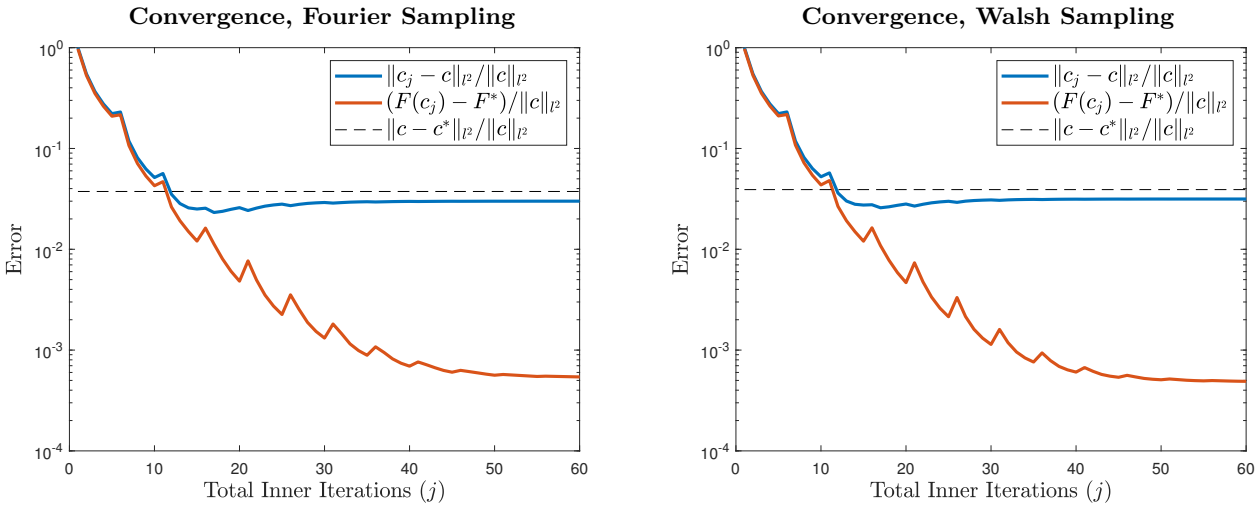

Fig. S4. The convergence of the algorithm in the number of inner iterations. The dashed line shows $\|c - c^*\|_{l^2}/\|c\|_{l^2}$. In both cases, the error between the reconstruction and the image decreases exponentially until this bound is reached. The objective function gap decreases exponentially slightly beyond this point, demonstrating that the robust null space property (in levels) controls the l^2-norm difference between vectors (locally around c^*) down to the error $\|c - c^*\|_{l^2}$ (see the bound in Eq. (4.18) in our proof).

References

1. Y LeCun, Y Bengio, G Hinton, Deep learning. *Nature* **521**, 436 EP – (2015).
2. CF Higham, DJ Higham, Deep learning: An introduction for applied mathematicians. *SIAM Rev.* **61**, 860–891 (2019).
3. A Pinkus, Approximation theory of the MLP model in neural networks. *Acta Numer.* **8**, 143–195 (1999).
4. S Shalev-Shwartz, S Ben-David, *Understanding machine learning: From theory to algorithms*. (Cambridge university press), (2014).
5. KH Jin, MT McCann, E Froustey, M Unser, Deep convolutional neural network for inverse problems in imaging. *IEEE Trans. Image Process.* **26**, 4509–4522 (2017).
6. K Hammernik, et al., Learning a variational network for reconstruction of accelerated MRI data. *Magn. Reson. Med.* **79**, 3055–3071 (2018).
7. R DeVore, B Hanin, G Petrova, Neural network approximation. *Acta Numer.* **30**, 327–444 (2021).
8. N Boullé, Y Nakatsukasa, A Townsend, Rational neural networks. *arXiv:2004.01902* (2020).
9. B Adcock, AC Hansen, Generalized sampling and infinite-dimensional compressed sensing. *Found. Comput. Math.* **16**, 1263–1323 (2016).
10. B Adcock, AC Hansen, C Poon, B Roman, Breaking the coherence barrier: A new theory for compressed sensing in *Forum Math., Sigma*. (CUP), Vol. 5, (2017).
11. B Adcock, C Boyer, S Brugiapaglia, On oracle-type local recovery guarantees in compressed sensing. *Inf. Inference* (2018).
12. AC Hansen, On the solvability complexity index, the n-pseudospectrum and approximations of spectra of operators. *J. Amer. Math. Soc.* **24**, 81–124 (2011).
13. AC Hansen, O Nevanlinna, Complexity issues in computing spectra, pseudospectra and resolvents. *Banach Cent. Publ.* **112**, 171–194 (2016).
14. J Ben-Artzi, MJ Colbrook, AC Hansen, O Nevanlinna, M Seidel, Computing spectra – On the solvability complexity index hierarchy and towers of algorithms. *arXiv:1508.03280* (2020).
15. MJ Colbrook, Computing spectral measures and spectral types. *Commun. Math. Phys.* **384**, 433–501 (2021).
16. M Webb, S Olver, Spectra of Jacobi operators via connection coefficient matrices. *Commun. Math. Phys.* **382**, 657–707 (2021).
17. M Colbrook, A Horning, A Townsend, Computing spectral measures of self-adjoint operators. *SIAM Rev.* **63**, 489–524 (2021).
18. J Ben-Artzi, M Marletta, F Rösler, Computing the sound of the sea in a seashell. *Found. Comput. Math.*, 1–35 (2021).
19. J Ben-Artzi, M Marletta, F Rösler, Computing scattering resonances. *arXiv preprint arXiv:2006.03368* (2020).
20. MJ Colbrook, AC Hansen, The foundations of spectral computations via the solvability complexity index hierarchy. *arXiv preprint arXiv:1908.09592* (2021).
21. MJ Colbrook, On the computation of geometric features of spectra of linear operators on Hilbert spaces. *arXiv preprint arXiv:1908.09598* (2021).
22. MJ Colbrook, B Roman, AC Hansen, How to compute spectra with error control. *Phys. Rev. Lett.* **122**, 250201 (2019).
23. C McMullen, Families of rational maps and iterative root-finding algorithms. *Annals Math. (2)* **125**, 467–493 (1987).
24. P Doyle, C McMullen, Solving the quintic by iteration. *Acta Math.* **163**, 151–180 (1989).
25. H Boche, V Pohl, The solvability complexity index of sampling-based Hilbert transform approximations in *2019 13th International conference on Sampling Theory and Applications (SampTA)*. (IEEE), pp. 1–4 (2019).
26. A Bastounis, AC Hansen, V Vlačić, The extended Smale's 9th problem – On computational barriers and paradoxes in estimation, regularisation, computer-assisted proofs and learning. *arXiv:2110.15734* (2021).
27. S Smale, Mathematical problems for the next century. *Math. Intell.* **20**, 7–15 (1998).
28. A Turing, On Computable Numbers, with an Application to the Entscheidungsproblem. *Proc. Lond. Math. Soc. (2)* **42**, 230–265 (1936).
29. L Blum, F Cucker, M Shub, S Smale, *Complexity and Real Computation*. (Springer-Verlag New York, Inc.), (1998).
30. S Arora, B Barak, *Computational complexity: a modern approach*. (Cambridge University Press), (2009).
31. P Bürgisser, F Cucker, *Condition : the geometry of numerical algorithms*, Grundlehren der mathematischen Wissenschaften. (Springer, Berlin, Heidelberg, New York), (2013).
32. A Chambolle, T Pock, A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vis.* **40**, 120–145 (2011).
33. A Chambolle, T Pock, On the ergodic convergence rates of a first-order primal–dual algorithm. *Math. Program.* **159**, 253–287 (2016).
34. B Adcock, V Antun, AC Hansen, Uniform recovery in infinite-dimensional compressed sensing and applications to structured binary sampling. *arXiv:1905.00126* (2019).
35. RT Rockafellar, Monotone operators and the proximal point algorithm. *SIAM J. Control. Optim.* **14**, 877–898 (1976).
36. S Foucart, H Rauhut, *A mathematical introduction to compressive sensing*. (Birkhäuser Basel), (2013).
37. B Adcock, AC Hansen, *Compressive Imaging: Structure, Sampling, Learning*. (Cambridge University Press), (2021).
38. B Adcock, AC Hansen, B Roman, A note on compressed sensing of structured sparse wavelet coefficients from subsampled Fourier measurements. *IEEE Signal Process. Lett.* **23**, 732–736 (2016).
39. V Antun, Coherence estimates between Hadamard matrices and Daubechies wavelets (2016) Master's thesis, *University of Oslo*.