THE EXTENDED SMALE'S 9TH PROBLEM

—

ON COMPUTATIONAL BARRIERS AND PARADOXES IN ESTIMATION, REGULARISATION, COMPUTER-ASSISTED PROOFS, AND LEARNING

A. BASTOUNIS, A. C. HANSEN, AND V. VLAČIĆ

1. INTRODUCTION

Linear and semidefinite programming (LP, SDP), regularisation through basis pursuit (BP) and Lasso, as well as neural networks in deep learning have seen great success in mathematics, statistics, data science, and computer-assisted proofs. The success and performance of LP is traditionally attributed to the fact that it is polynomially solvable (colloquially, "in P") for rational inputs. On the other hand, in his list of problems for the 21st century [1] S. Smale calls for "[Computational] models which process approximate inputs and which permit round-off computations". Indeed, since e.g. $\sqrt{\cdot}$ and $\exp(\cdot)$ do not have exact representations, inaccurate data input is a daily encounter. The relevance of such a model is further emphasised by the fact that even a rational number such as $1/3$ is stored only approximately when using floating-point arithmetic, a situation faced by most software. This model allowing inaccurate input of arbitrary precision, which we call *the extended model*, leads to extended versions of fundamental problems such as: "Are LP and other aforementioned problems in P?" The same question can be asked of an extended version of Smale's 9th problem [1] on the list of mathematical problems for the 21st century. Recall, Smale's 9th problem reads

> *Is there a polynomial time algorithm over the real numbers which decides the feasibility of the linear system of inequalities $Ax \geq y$, and if so, outputs such an x?*

One can thus pose this problem in the extended model where A and y are given as inexact inputs with arbitrary precision. Similarly, the optimisation problems BP, SDP, as well as (constrained and unconstrained) Lasso, where the task is to output a solution to a specified precision, can likewise be posed in the extended model. Given the widespread use of randomised algorithms, one can then ask questions on the existence of randomised algorithms providing an approximate solution with a certain probability. We will collectively refer to these problems as the *extended Smale's 9th problem* (see Problem 1 for the precise formulation), which we will consider in both the extended Turing and the extended Blum-Shub-Smale (BSS) model for real arithmetic.

We settle this problem in both the negative and the positive, revealing two surprises: (1) In mathematics, sparse regularisation, statistics, and learning, one successfully computes with non-computable problems (e.g., in compressed sensing, for which we provide a detailed account). The same happens also in computer-assisted proofs, for example in the famous proof of Kepler's conjecture (Hilbert's 18th problem) [2], [3]. (2) In order to mathematically characterise this phenomenon, one needs an intricate complexity theory for, seemingly paradoxically, non-computable problems.

Short summary of the main results on the extended Smale's 9th problem: Consider the task of computing a minimiser of LP, BP, or (constrained and unconstrained) Lasso in the extended model, and choose an appropriate norm to measure the error. Then, for any integer $K > 2$, there exists a class of feasible inputs Ω such that we have the following.

   (i) No algorithm, even randomised, can produce K correct digits of the true solution for all inputs in Ω (with probability exceeding p $> 1/2$ in the randomised case).

  (ii) If we allow randomised algorithms with non-zero probability of not halting (i.e., not producing an output), then no such algorithm can produce K correct digits for all inputs in Ω with probability exceeding p $> 2/3$. However, there does exist such an algorithm that produces K correct digits for all inputs in Ω with probability $2/3$.

 (iii) One cannot decide if a given algorithm taking inputs from Ω fails to produce K correct digits on a given input. This is in fact strictly harder than solving the original problem in the following sense. Even if given an oracle for solving, e.g., LP accurately, one cannot decide if the algorithm for solving LP successfully produces K correct digits of the true solution.

 (iv) There does exist an algorithm that provides $K - 1$ correct digits for all inputs in Ω. However, any such algorithm will need an arbitrarily long time to achieve this. Specifically, there is an $\Omega' \subset \Omega$ with inputs of fixed dimensions, such that, for any $T > 0$ and any algorithm Γ, there exists an input $\iota \in \Omega'$ so that either $\Gamma(\iota)$ does not approximate the true solution with $K - 1$ correct digits or the runtime of Γ on ι exceeds T. Moreover, for any randomised algorithm Γ^{ran} and p $\in (0, 1/2)$, there exists an input $\iota \in \Omega'$ such that

$$\mathbb{P}\big(\Gamma^{\mathrm{ran}}(\iota) \text{ does not approximate the true solution with } K - 1 \text{ correct digits}$$
$$\text{or the runtime of } \Gamma \text{ on } \iota \text{ exceeds } T\big) > \mathrm{p}.$$

  (v) The problem of producing $K - 2$ correct digits for all inputs in Ω is in P, i.e., can be solved in time polynomial in the number of variables n.

 (vi) If one only considers (i) - (iv), Ω can be chosen with any fixed dimensions $m < N$ with $m \geq 4$ (see §1.1 for the precise formulation of the problems) . Moreover, if one only considers (i) - (iii), then K can be chosen to be 1.

(vii) Furthermore, there are classes $\Omega^\sharp \neq \Omega$ of problems for which there are algorithms that can produce K correct digits of the true solution for any input in $\Omega^\sharp$ and any K, with runtime bounded by a polynomial in both n and K, as opposed to just n as above.

(viii) Similar results to the above hold for the extended LP feasibility decision problem.

In the above, we use the unqualified term *algorithm* to mean an algorithm that always provides an output, i.e., always halts. More precise and elaborate versions of these claims will follow in Theorem 3.4, Theorem 5.1, Theorem 6.1, and Theorem 7.1.

It may come as a surprise that the above results are independent of the many condition numbers in the literature (see §9.1). Indeed, bounded condition numbers generally do not imply the existence of successful algorithms, and, on the other hand, there are problem classes in P with infinite condition numbers. Moreover, the input can be bounded from above and below. The reader is invited to consult §4 for a demonstration of the failure of modern software consistent with the above results.

Our theory leading to the results above has implications for a variety of fields such as computer-assisted proofs, compressed sensing, and Smale's 18th problem in connection with deep learning that can be summarised briefly as follows.

Computer-assisted proofs: The recent computer-assisted proof, led by T. Hales [2], [3], of the long-standing Kepler's conjecture was possible despite relying on computing with non-computable problems. A crucial part of the proof consists of the numerical computation of linear programs with inexact input

as in the extended model described above. In view of (i) - (iii) above, this may seem paradoxical. However, non-computable problems can indeed be used in computer-assisted proofs, the Dirac-Schwinger conjecture proved by C. Fefferman and L. Seco in [4]–[12] serving as another example.

Compressed sensing: This widely applied scientific and engineering discipline is an example where all of (i)-(vii) apply. Indeed, as we show, for many compressed sensing problems one cannot compute an approximate solution with, say, five digits accuracy, however, one can compute a four digit approximation efficiently. In imaging problems this typically suffices due to limitations of the human eye.

Smale's 18th problem and deep learning: The 18th problem on Smale's list [1] echoes Turing's famous paper from 1950 [13] and asks about the limitations of artificial intelligence. Our theory sheds light on this problem in view of the following paradoxes in deep learning.

(i) (*Non-existence of algorithms*) The standard computational problem of constructing a neural network through risk minimisation and applying it to test data is in general non-computable.

(ii) (*Instability*) Neural networks based on deep learning may have great success rate but typically become unstable. Specifically, there are uncountably many classification problems for which the following happens. One may have 100% success rate on arbitrarily large training and validation data sets, and yet there are uncountably many points arbitrarily close to the training data for which the trained network will fail.

Contents

1.1. **The computational problems.** Linear and semidefinite programming, regularisation techniques such as basis pursuit and Lasso, as well as deep learning have become highly popular over the last decades. These approaches have in many areas of mathematics, statistics, learning, artificial intelligence and data science changed the state of the art from linear to non-linear approaches, typically via obtaining minimisers of both convex and non-convex problems [14]–[28]. The list of areas using these techniques is far reaching and their influence has been extensive. The key problems to compute are:

(i) Linear Programming (LP)

$$z \in \operatorname*{argmin}_{x} \langle x, c \rangle \text{ subject to } Ax = y, \quad x \geq 0, \qquad (1.1)$$

(ii) Basis Pursuit (BP)

$$z \in \operatorname*{argmin}_{x} \mathcal{J}(x) \text{ subject to } \|Ax - y\|_2 \leq \delta, \qquad \delta \in [0, 1], \qquad (1.2)$$

(iii) Unconstrained Lasso (UL)

$$z \in \operatorname*{argmin}_{x} \|Ax - y\|_2^2 + \lambda \mathcal{J}(x), \qquad \lambda \in (0, 1], \qquad (1.3)$$

(iv) Constrained Lasso (CL)

$$z \in \operatorname*{argmin}_{x} \|Ax - y\|_2 \text{ subject to } \|x\|_1 \leq \tau, \qquad \tau > 0, \qquad (1.4)$$

(v) Semidefinite Programming (SDP)

$$Z \in \operatorname*{argmin}_{X \in \mathbb{S}^n} \langle C, X \rangle_{\mathbb{S}^n} \text{ subject to } \langle A_k, X \rangle_{\mathbb{S}^n} = b_k, X \succeq 0, k = 1, \ldots, m \qquad (1.5)$$

(vi) Neural Networks (NN)

$$\{\phi(y^j)\}_{j=1}^s, \text{ where } \quad \phi \in \operatorname*{argmin}_{\tilde{\phi} \in \mathcal{NN}_{\mathbf{N},L,d}} \mathcal{R}\big(\{\tilde{\phi}(x^j)\}_{j=1}^r, \{f(x^j)\}_{j=1}^r\big), \qquad (1.6)$$

for $\mathcal{T} = \{x^1, \ldots, x^r\} \subset \mathbb{R}^d$ and $\mathcal{V} = \{y^1, \ldots, y^s\}$.

In the above notation we have

$$A \in \mathbb{R}^{m \times N}, y \in \mathbb{R}^m, c \in \mathbb{R}^N, \quad \mathcal{J}(x) = \|x\|_1 \text{ or } \mathcal{J}(x) = \|x\|_{\mathrm{TV}},$$

where the TV semi-norm is defined as $\|x\|_{\mathrm{TV}} = \sum_{j=1}^{N-1} |x_j - x_{j+1}|$. For SDP, the notation is

$$C, A_k \in \mathbb{S}^n \text{ (real } n \times n \text{ symmetric matrices)}, \quad b_k \in \mathbb{R}, \quad \langle C, X \rangle_{\mathbb{S}^n} = \operatorname{trace}(C^T X).$$

Finally, in the case of neural networks $\mathcal{T}$ denotes the training set, $\mathcal{V}$ the validation set, $f : \mathbb{R}^d \to \{0, 1\}^{N_L}$ is the classification function to be learnt, and $\mathcal{R} \in \mathcal{CF}$ is the cost function, where

$$\mathcal{CF} = \{\mathcal{R} : \mathbb{R}^{r \times N_L} \times \mathbb{R}^{r \times N_L} \to \mathbb{R}_+ \mid \mathcal{R}(v, w) = 0 \text{ iff } v = w\}.$$

Moreover, $\mathcal{NN}_{\mathbf{N},L,d}$, where $\mathbf{N} = (N_L, N_{L-1}, \ldots, N_1, N_0 = d)$, denotes the set of all L-layer neural networks, i.e., all mappings $\phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ of the form

$$\phi(x) = W^L(\rho(W^{L-1}(\rho(\cdots \rho(W^1(x)) \cdots)))), \quad x \in \mathbb{R}^d,$$

where $W^\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $1 \leq \ell \leq L$, are affine mappings and $\rho : \mathbb{R} \to \mathbb{R}$ is a function which acts on vectors componentwise. Throughout the paper we consider the ReLU nonlinearity, i.e., $\rho(x) = \max\{0, x\}$.

Note that all of the problems above may have multi-valued solutions in certain cases. Whenever this occurs, the computational problem of interest is to compute any of these solutions. We will throughout the paper use the notation

$$\Xi : \Omega \rightrightarrows \mathcal{M}, \qquad (1.7)$$

to denote the multivalued solution map, mapping an input $\iota \in \Omega$ to a metric space $(\mathcal{M}, d_{\mathcal{M}})$, allowing measurement of error. The metric space is typically $\mathbb{R}^N$ or $\mathbb{C}^N$ equipped with the $\|\cdot\|_2$ norm, however, any metric can be considered. Even though the solution map Ξ may be multivalued, in our theory the

output of an algorithm will always be single-valued. Thus, if $\Gamma : \Omega \to \mathcal{M}$ is an algorithm we measure the approximation error by

$$\text{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) = \inf_{\xi \in \Xi(\iota)} d_{\mathcal{M}}(\Gamma(\iota), \xi).$$

Remark 1.1 (Objective function vs minimisers)**.** In this paper we are primarily concerned with the problem of obtaining minimisers that are vectors and not the real-valued minimum value of the objective function. There is a very rich literature [29]–[33] on how to compute the objective function, and, in particular, the minimum value $f(x^*) = \min\{f(x) \,|\, x \in \mathcal{X}\}$, for some convex function $f : \mathbb{R}^d \to \mathbb{R}$, convex set $\mathcal{X} \subset \mathbb{R}^d$, and minimiser $x^* \in \mathcal{X}$. The traditional problem of interest is as follows. Given $\epsilon > 0$, compute an $x_\epsilon \in \mathbb{R}^d$ such that $f(x_\epsilon) - f(x^*) \le \epsilon$. Note, however, that $f(x_\epsilon) - f(x^*) \le \epsilon$ does not necessarily mean that

$$\|x_\epsilon - x^*\| \le \epsilon. \tag{1.8}$$

In this paper, however, it is exactly the problem of computing x_ϵ such that we have (1.8) that will be of prime interest. The motivation behind this is self-evident as there are vast areas of mathematics of information, regularisation, estimation, learning, compressed sensing and data sciences where the object of interest is the minimiser and not the minimum value.

2. The extended model and the extended Smale's 9th problem

The question: "is LP in P?" [34]–[36] was a fundamental problem whose solution reached the front page of The New York Times [37]. The affirmative answer has been refined several times and is now typically stated in the following form. One can solve LPs with rational inputs in runtime is bounded by

$$\mathcal{O}(n^{3.5} L^2 \cdot \log L \cdot \log \log L), \tag{2.1}$$

where n denotes the number of variables and L is the number of bits or digits required in the representation of the inputs [38], [39]. The problem, however, is that in an overwhelming number of problems in computational mathematics and scientific computing the input contains irrational numbers. Thus, if the number $\sqrt{2}$ occurs as an input in the LP, the question remains of what the cost of producing an accurate solution is.

2.1. **Inexact input and the extended model.** An example of an LP where the matrix A contains irrational numbers is when its rows derive from the discrete cosine transform or a discrete wavelet transform. Note that these are not contrived examples. In fact, in the fields of inverse problems, medical imaging, compressed sensing, etc. this is a common occurrence. Since A contains numbers that cannot be represented exactly as binary numbers, the bound (2.1) does not apply. Therefore, the classical model for asking "is LP in P?" does not address the common case of irrepresentable input. On the practical side, an overwhelming amount of the modern software used is based on floating-point arithmetic, and hence if the input is rational, there will be inexactness due to the floating-point representation. For example, $1/3$ can only be approximated in base 2, giving rise to round-off approximation. Indeed, the following quote explains the situation succinctly:

> *"But real number computations and algorithms which work only in exact arithmetic can offer only limited understanding. Models which process approximate inputs and which permit round-off computations are called for."*

> — S. Smale (from the list of mathematical problems for the 21st century [1])

This issue illustrates the classical dichotomy in mathematics between the discrete and continuous. Indeed, as mentioned above, classical complexity analysis for LP is done in a discrete model. Yet, LP

can also be naturally considered in the continuous world, which is the standard way in which Smale's 9th problem is stated. The following quote draws attention to this dichotomy:

> *" Perhaps the most successful tool in economics and operations research is linear programming, which lives on the boundary of discrete and continuous."*
>
> — L. Lovasz (from"Discrete and Continuous: Two sides of the same?"
> in "Visions in Mathematics", essays on mathematics entering the 21st century [40])

These issues call for an extension of the continuous model that accommodates inexact input, introduced next.

2.1.1. *The extended model - inexact input provided by an oracle.* Suppose that we are given an algorithm (a Turing or Blum-Shub-Smale (BSS) machine) intended to solve LP (or any of the other problems in §1.1), and furthermore assume that the algorithm is equipped with an oracle $\mathscr{O}$ that can acquire the true input to any accuracy ϵ. A natural assumption in this scenario is that the oracle completes its task in time polynomial in $|\log(\epsilon)|$ (see for example Lovász [41, p. 36]). More concretely, given a domain $\Omega \subset \mathbb{C}^n$ of inputs, the algorithm cannot access $\iota \in \Omega$, but rather, for any $k \in \mathbb{N}$, it can call the oracle $\mathscr{O}$ to obtain $\tilde{\iota} = \mathscr{O}(\iota, k) \in \mathbb{C}^n$ satisfying

$$\|\mathscr{O}(\iota, k) - \iota\|_\infty \leq 2^{-k}, \qquad \forall \iota \in \Omega, \forall k \in \mathbb{N}, \tag{2.2}$$

and the time cost of accessing $\mathscr{O}(\iota, k)$ is polynomial in k. Another key assumption when discussing the success of the algorithm is that it must be "oracle agnostic", i.e., it must work with any choice of the oracle $\mathscr{O}$ satisfying (2.2). In the Turing model the Turing machine accesses the oracle via an oracle tape and in the BSS model the BSS machine accesses the oracle through an oracle node.

2.1.2. *The extended Smale's 9th problem.* Given that the input is inexact, the output of an algorithm will come with an error as well. The model, both in the Turing and the BSS case, where one measures the computational cost of running the algorithm in terms of the number of variables n and the error (or the number of correct digits $K = |\log(\epsilon)|$, where ϵ is the error) is well established. See, for example Blum, Cucker, Shub and Smale [42, p. 29], Grötschel, Lovász and Schrijver [37, p. 34] and Valiant [43, p. 131]). We thus arrive at the following extension of Smale's 9th problem.

Problem 1 (The extended Smale's 9th problem). *Given any of the problems in* (1.1) - (1.4)*, represented by the solution map Ξ mapping a class of inputs Ω into a metric space $(\mathcal{M}, d_\mathcal{M})$, is there an algorithm which decides the feasibility of the problem, and if so, produces an output that is correct up to K digits (where the error is measured via $\mathrm{dist}_\mathcal{M}$) and whose computational cost is bounded by a polynomial in K and the number of variables n?*

This question can be asked both in the Turing model, where the computational cost can be expressed either in terms of the number of steps performed by the Turing machine, or alternatively in terms of the total number of arithmetic operations and comparisons as well as the space complexity. In the BSS model, the computational cost is given by the total number of arithmetic operations and comparisons executed by the BSS machine. We will consider all these cases.

Remark 2.1 (Weaker and randomised versions of the extended Smale's 9th problem). The question in Problem 1 can be weakened by asking if, for a fixed K, there is an algorithm polynomial in the number of variables that produces an output that is correct up to K digits. One can also weaken the statement by allowing randomised algorithms and asking whether an algorithm succeeds with probability $\mathrm{p} \in [0, 1]$. In the case of feasibility questions one can weaken the question by relaxing the constraints to only be

COMPUTATIONAL BARRIERS IN ESTIMATION, REGULARISATION AND LEARNING　　　　　　7

satisfied up to a certain accuracy. For example, given $K \in \mathbb{N} \cup \{\infty\}$ and $M \in \mathbb{R}$, one may ask to decide whether there is an $x \in \mathbb{R}^N$ such that

$$\langle x, c \rangle_K \leq M \text{ subject to } Ax = y, \quad x \geq 0,$$

where $\langle x, c \rangle_K = \lfloor 10^K \langle x, c \rangle \rfloor 10^{-K}$. We will discuss this particular problem later in §6.1 in connection with the computer-assisted proof of Kepler's conjecture.

We will as a general rule split the feasibility problem and the problem of producing an approximate solution into separate theorems.

3. MAIN THEOREM I – THE EXTENDED SMALE'S 9TH PROBLEM

The main results on the existence of successful polynomial cost algorithms for the extended Smale's 9th problem are summarised in Theorem 3.4, Theorem 6.1, and Theorem 7.1, whereas Theorem 5.1 deals with the decision problem of certifying the correctness of an algorithm (the "exit flag" problem). We now present each of these theorems.

3.1. **Universality of the results.** The statements in the theorems below are well-defined up to the definition of an algorithm, randomised algorithm and runtime. There are a myriad of different types of machines that can be used as the formal definition of an algorithm: the Turing machine, the BSS machine, the Von Neumann computer, the real RAM, etc. as well as their randomised versions. The different models are not equivalent when it comes to computability and runtime. Thus, to create universal impossibility results we use the concept of a general algorithm (defined in §9.2) and a randomised general algorithm (defined in §9.5) that encompasses any reasonable definition of a computational model in the way that they are more powerful than any standard machine, therefore making the impossibility results stronger. For example, these general algorithms can solve the halting problem, and hence, perhaps surprisingly, the impossibility results are not about recursiveness vs. non-recursiveness. In order to make our theorems reader-friendly, we state them before giving the precise definitions of algorithms and runtime. Moreover, in order to assure a streamlined exposition, some of the results are slightly weaker than what we actually prove. The fully formal statements of the theorems can be found in the propositions in §9 further below, and references to these follow each theorem.

3.2. **The paradox - A complexity theory for non-computable problems.** The first paradoxical result demonstrates the following intricate phenomenon. Key problems used in statistical estimation, sparse regularisation, compressed sensing, learning, and modern data science require a complexity theory for non-computable problems. The term "non-computable" here refers to the classical definition given by Turing in [44]. As it turns out, despite the classical result of LP being in P for rational inputs, in the extended model where inputs can be accessed only approximately even LP is non-computable, and yet we can discuss its complexity in a sense that will presently become apparent. In the following Ξ will denote the solution map (as in (1.7)) to any of the problems (1.1) - (1.4) with set of inputs Ω, so that

$$\Omega = \bigcup_{N > m \geq 4}^{\infty} \Omega_{m,N}, \quad \Xi : \Omega_{m,N} \rightrightarrows \mathcal{M}_N, \tag{3.1}$$

where $\Omega_{m,N}$ is a nonempty set of inputs for Ξ of fixed dimensions m and N, and $\mathcal{M}_N$ is $\mathbb{R}^N$ equipped with the $\| \cdot \|_p$ norm for some $p \in [1, \infty]$. The fact that the dimensions blow up is crucial in order to make sense of "in P"-type statements.

Remark 3.1 (Algorithms that may not halt). As already mentioned in the introduction, unless specified otherwise, the term *algorithm* will mean an algorithm that always provides an output, i.e., always

halts. Interestingly, in the randomised case, an algorithm that is allowed to not halt with non-zero probability may be more powerful than a randomised algorithm that is required to halt. An example of this phenomenon is provided in Theorem 3.4 parts (i) and (ii).

Remark 3.2 (Computing K correct digits). Having specified the p-norm $\| \cdot \|_p$ for measuring the error in $\mathcal{M}_N$, we frequently discuss whether an algorithm can "produce (or compute) K correct digits for Ξ". For an algorithm having access to the dimensions m, N and an oracle representation $\tilde{\iota}$ (according to 2.2) of an input $\iota \in \Omega_{m,N}$, this will mean the assertion that

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(m, N, \tilde{\iota}), \Xi(\iota)) = \inf_{\xi \in \Xi(\iota)} \|\Gamma(m, N, \tilde{\iota}) - \xi\|_p \leq 10^{-K}, \tag{3.2}$$

for all m, N and all possible oracle representations $\tilde{\iota}$ of all $\iota \in \Omega_{m,N}$. Moreover, in the randomised case, (3.2) should happen with a certain probability that will always be made explicit.

Remark 3.3 (Condition numbers). In the following we refer to several condition numbers common in the literature, namely the condition of a matrix, the feasibility-primal condition number C_{FP}, and the condition of a solution map. The precise definitions of these can be found in §9.1.

Theorem 3.4 (The extended Smale's 9th problem - computing solutions). *Let Ξ denote the solution map to any of the problems (1.1) - (1.4) with the regularisation parameters satisfying $\delta \in [0, 1]$, $\lambda \in (0, 1/3]$, and $\tau \in [1/2, 2]$ (and additionally being rational in the Turing case) and consider the $\| \cdot \|_p$-norm for measuring the error, for an arbitrary $p \in [1, \infty]$. Let $K > 2$ be an integer. There exists a class Ω of feasible inputs as in (3.1) so that we have the following.*

   (i) No algorithm can produce K correct digits on each input in Ω as in (3.2). Moreover, for any $\mathrm{p} > \frac{1}{2}$, no randomised algorithm can produce K correct digits with probability greater than or equal to p on each input in Ω.

   (ii) If we allow randomised algorithms with a non-zero probability of not halting (not producing an output), then, for any $\mathrm{p} > \frac{2}{3}$, no such algorithm can produce K correct digits with probability greater than or equal to p on each input in Ω. However, there does exist such an algorithm that can produce K correct digits on each input in Ω with probability $2/3$.

  (iii) There does exist an algorithm (a Turing or a BSS machine) that produces $K - 1$ correct digits for all inputs in Ω. However, any such algorithm will need an arbitrarily long time to achieve this. In particular, for any fixed dimensions m, N, any $T > 0$, and any algorithm Γ, there exists an input $\iota \in \Omega_{m,N}$ such that either Γ on input ι does not produce $K - 1$ correct digits for $\Xi(\iota)$ or the runtime of Γ on ι exceeds T. Moreover, for any randomised algorithm Γ^{ran} and $\mathrm{p} < 1/2$ there exists an input $\iota \in \Omega_{m,N}$ such that

$$\mathbb{P}\big(\Gamma^{\mathrm{ran}}(\iota) \text{ does not produce } K - 1 \text{ correct digits for } \Xi(\iota)$$

$$\text{or the runtime of } \Gamma \text{ on } \iota \text{ exceeds } T\big) > \mathrm{p}.$$

  (iv) There exists a polynomial $\mathrm{pol} : \mathbb{R} \to \mathbb{R}$, as well as a Turing machine and a BSS machine that both produce $K - 2$ correct digits for all inputs in Ω, so that the number of arithmetic operations for both machines is bounded by $\mathrm{pol}(n)$, where $n = m + mN$ is the number of variables, and the number of digits required from the oracle (2.2) is bounded by $\mathrm{pol}(\log(n))$. Moreover, the space complexity of the Turing machine is bounded by $\mathrm{pol}(n)$.

   (v) If one only considers (i) - (iii), Ω can be chosen with any fixed dimensions m and N provided that $m \geq 4$ and $N > m$. Moreover, if one only considers (i) then K can be chosen to be 1.

The statements (i) - (iii) above are true even when we require the input in each $\Omega_{m,N}$ to be well-conditioned and bounded from above. In particular, for any input $\iota = (y, A) \in \Omega_{m,N}$ ($\iota = (y, A, c)$ in the case of LP) we have $\mathrm{Cond}(AA^*) \leq 3.2$, $C_{\mathrm{FP}}(\iota) \leq 4$, $\mathrm{Cond}(\Xi) \leq 179$, $\|y\|_\infty \leq 2$, and $\|A\|_{\max} = 1$.

The precise and slightly stronger version of Theorem 3.4 is summarised in Proposition 9.32 and Proposition 9.33.

Remark 3.5 (Lower bound on the vector component of the inputs)**.** Note that the matrix component of every input $\iota = (y, A) \in \Omega$ in Theorem 3.4 is bounded from both below and above with universal bounds, specifically, $\|A\|_{\max} := \max_{i,j} |A_{i,j}| = 1$. The vector component y admits an upper bound but does not admit a lower bound uniform across Ω. This minor deficiency is an artefact of our construction of Ω and could be easily remedied by carrying out a more involved construction. This, however, would significantly increase the technical detail of the proof while not contributing to the theory nor any of the techniques developed, so we refrain from doing so.

Remark 3.6. The results of Theorem 3.4 also hold for the SDP problem (1.5), which can be shown easily by employing the standard argument to recast the LP problem (together with the associated class Ω) as an SDP. With this embedding the impossibility results hold immediately, however the parts referring to the existence of algorithms would need to be proven separately.

Remark 3.7 (Turing's definition of computability - should it be extended?)**.** There is traditionally a sharp divide in the literature between problems that are computable (according to Turing's definition [44]) and those that are not. Moreover, complexity theory is only considered for problems that are computable. Turing's definition of computability subsumes that one can compute an approximation to arbitrarily high accuracy. Although this is a natural definition in contexts where the phenomena described in Theorem 3.4 do not occur, it deems many problems of interest non-computable despite the fact that computing with them is commonplace. The apparent paradox is resolved by noting that these problems are only solved to an accuracy needed for the application of interest. Arbitrary accuracy, which by Theorem 3.4 may be impossible, is typically not needed. Standard examples include problems related to imaging, where the limitation of the human eye makes high precision superfluous. Indeed, as we show later in Theorem 7.1 on compressed sensing in the context of Smale's 9th problem, the phenomenon described in Theorem 3.4 is not due to exotic constructions, but also occurs naturally in key applications. Hence, in order to analyse and describe the above phenomenon, we need a complexity theory for non-computable problems. With this in mind, one could simply extend Turing's definition of computability and, for instance, call a problem K-computable if one can compute K correct digits, and replace the traditional concept of Turing computability with ∞-computability. In this setup, each class of K-computable problems would get its own complexity theory.

4. Failure of modern algorithms and computing the exit flag

A crucial topic in computational mathematics is the reliability of algorithms. This is a paramount issue in pure mathematics when computer-assisted proofs are used and in applications in the sciences in general. It is therefore natural to test whether the built-in algorithms in, for example, Matlab are reliable. We consider two concrete examples: the linear program

$$\min_{x \in \mathbb{R}^2} x_1 + x_2 \text{ subject to } x_1 + (1 - \delta)x_2 = 1, \qquad x_1, x_2 \geq 0, \tag{4.1}$$

where $\delta > 0$ is a parameter, and the centred and standardised (so that the columns of the design matrix are normalised) Lasso problem

$$\min_{x \in \mathbb{R}^N} \frac{1}{m}\|A_\delta D_\delta x - y\|_2^2 + \lambda\|x\|_1, \tag{4.2}$$

$+3$	The solution is feasible with respect to the relative `ConstraintTolerance` tolerance, but is not feasible with respect to the absolute tolerance.
$+1$	Function converged to a solution `x`.
0	Number of iterations exceeded `options.MaxIterations` or solution time in seconds exceeded `options.MaxTime`.
-2	No feasible point was found.
-3	Problem is unbounded.
-4	`NaN` value was encountered during execution of the algorithm.
-5	Both primal and dual problems are infeasible.
-7	Search direction became too small. No further progress could be made.
-9	Solver lost feasibility.

TABLE 1. The `EXITFLAG` is used to verify the correctness of the solution. Possible values for the `EXITFLAG` output of `linprog` as well as their corresponding interpretations are displayed in this table. Note that a value of 1 indicates the correctness of the solution, whereas other values indicate various types of failure.

	'dual-simplex'		'interior-point'		'interior-point-legacy'	
δ	Error	EXITFLAG	Error	EXITFLAG	Error	EXITFLAG
2^{-1}	0	1	0	1	$6.0 \cdot 10^{-12}$	1
2^{-15}	0	1	0	1	$3.0 \cdot 10^{-5}$	1
2^{-20}	0	1	0	1	$7.0 \cdot 10^{-7}$	1
2^{-24}	0	1	0	1	$7.1 \cdot 10^{-8}$	1
2^{-26}	1.4	1	1.4	1	$1.2 \cdot 10^{-1}$	1
2^{-28}	1.4	1	1.4	1	$4.6 \cdot 10^{-1}$	1
2^{-30}	1.4	1	1.4	1	$7.1 \cdot 10^{-1}$	1

TABLE 2. Testing the output of `linprog` applied to the problem in (4.1) for the three algorithms 'dual-simplex', 'interior-point' and 'interior-point-legacy'. The table shows the error $\|\hat{x} - \tilde{x}\|_{\ell^2}$ and the value of `EXITFLAG`, where $\hat{x}$ is the true minimiser of (4.1) and $\tilde{x}$ is the computed approximate minimiser. Note that machine epsilon is $\epsilon_{\mathrm{mach}} = 2^{-52}$.

where $m = 3, N = 2, \lambda \in (0, 1/\sqrt{3}]$,

$$A_\delta = \begin{pmatrix} \frac{1}{\sqrt{2}} - \delta & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} - \delta & -\frac{1}{\sqrt{2}} \\ 2\delta & 0 \end{pmatrix} \in \mathbb{R}^{3\times 2}, \quad y = \left(1/\sqrt{2} \quad -1/\sqrt{2} \quad 0\right)^T \in \mathbb{R}^3, \qquad (4.3)$$

and D_δ is the unique diagonal matrix such that each column of $A_\delta D_\delta$ has norm $\sqrt{m}$.

In order to compute a solution to (4.1), we consider Matlab's `linprog` command; a well-established and reliable optimisation algorithm for linear programs. This is a general purpose algorithm, which offers three different solvers: 'dual-simplex' (the default), 'interior-point', and 'interior-point-legacy'. Besides a minimiser, `linprog` also computes an additional output, `EXITFLAG`, which is an integer value corresponding to the reason for why the algorithm halted. These are listed in Table 1. Note that $+1$ indicates convergence to a minimiser, all other values indicate some form of failure. In Table 2 we apply the three `linprog` algorithms (with default settings) to the problem (4.1) with different values of δ. The results are fascinating. Not only does `linprog` completely fail to compute a minimiser accurately, it

	Default settings			'RelTol' = ϵ_{mach}			'RelTol' = ϵ_{mach} 'MaxIter' = $\epsilon_{\mathrm{mach}}^{-1}$		
δ	Error	Runtime	Warn	Error	Runtime	Warn	Error	Runtime	Warn
2^{-1}	$1 \cdot 10^{-16}$	$< 0.01s$	0	$1 \cdot 10^{-16}$	$< 0.01s$	0	$1 \cdot 10^{-16}$	$< 0.01s$	0
2^{-7}	0.68	$< 0.01s$	0	$2 \cdot 10^{-16}$	$0.02s$	0	$2 \cdot 10^{-16}$	$0.02s$	0
2^{-15}	1.17	$< 0.01s$	0	1.17	$0.33s$	1	$1 \cdot 10^{-11}$	$1381.5s$	0
2^{-20}	1.17	$< 0.01s$	0	1.17	$0.33s$	1	no output	$> 12h$	0
2^{-24}	1.17	$< 0.01s$	0	1.17	$0.34s$	1	no output	$> 12h$	0
2^{-26}	1.17	$< 0.01s$	0	1.17	$0.34s$	1	no output	$> 12h$	0
2^{-28}	1.17	$< 0.01s$	0	1.17	$< 0.01s$	0	1.17	$< 0.01s$	0
2^{-30}	1.17	$< 0.01s$	0	1.17	$< 0.01s$	0	1.17	$< 0.01s$	0

TABLE 3. Testing the output of `lasso` applied to the problem in (4.2) The output of `linprog` applied to (4.2) with inputs as in (4.3) and $\lambda = 0.1$. The table shows the error $\|\hat{x} - \tilde{x}\|_{\ell^2}$ (where $\hat{x}$ is the true minimiser and $\tilde{x}$ is the computed minimiser), the CPU runtime, and a boolean value indicating whether a `Warning` was issued.

also fails to recognise that the computed minimiser is incorrect: in all cases, the `EXITFLAG` returns the value $+1$ indicating a successful termination.

To compute a solution to (4.2), we consider Matlab's `lasso` command. We perform three tests: one with default settings, another with the tolerance parameter set to machine epsilon $\epsilon_{\mathrm{mach}} = 2^{-52}$, and finally one with the maximum number of iterations set to $\epsilon_{\mathrm{mach}}^{-1}$. The `lasso` routine does not have an 'exit flag', however, it provides a `Warning` if it considers the output to be untrustworthy. The results of this experiment are summarised in Table 3, where we display 1 under the `Warn` column if a `Warning` was issued, or 0 if no warning was issued. As is evident, the failure of `lasso` is similar to the failure of `linprog`, however, an interesting observation is that the `Warning` parameter is occasionally able to verify the wrong solution, yet, most of the time, no warning is issued despite completely inaccurate outputs. The failures of the `EXITFLAG` and the `Warning` suggest the following foundational question:

Problem 2 (Can the 'exit flag' be computed?). *Consider an algorithm designed to compute any of the problems* (1.1) - (1.4). *Suppose that the algorithm should produce K correct digits. Can we compute the 'exit flag' for this algorithm, i.e., the function taking on the value 1 if the algorithm succeeds in producing K correct digits, and 0 else?*

5. MAIN THEOREM II – THE EXTENDED SMALE'S 9TH PROBLEM AND THE EXIT FLAG

5.1. **The exit flag cannot be computed - even when given an exact solution oracle.** Let Ξ denote the solution map (as in (1.7)) to any of the problems (1.1) - (1.4), let $K \in \mathbb{N}$, and let Γ be an algorithm approximating Ξ. Suppose that Ω is a collection of inputs such that no algorithm can produce K correct digits for all inputs in Ω (recall that the existence of such an Ω is guaranteed by Theorem 3.4).

The exit flag problem is now as follows. Is it is possible to decide whether Γ fails or succeeds on a particular input? More precisely, can we find another algorithm that can compute the exit flag, meaning the function of the input taking on the value 1 when Γ succeeds in producing K correct digits for a particular input and 0 when it fails?

5.1.1. *Exact solution oracle.* To illustrate the difficulty of computing the exit flag, we add the assumption that we can also access the true solution, and show that even with this extra information the task is impossible. The most direct way to do this would be to assume that, in addition to accessing $\iota \in \Omega$, the

prospective algorithm for computing the exit flag can also access an exact solution $\rho \in \Xi(\iota)$. However, this assumption would be too restrictive, as we wish our theory to encompass Turing machines, which could not access ρ if it has irrational entries by their very nature. We thus assume instead that, for a fixed (but arbitrarily small) parameter $\omega > 0$, the algorithm can access a $\rho \in \mathcal{B}_\omega(\Xi(\iota))$, i.e., an element of $\mathcal{M}$ that is strictly within ω of a true solution. We will, nonetheless, consider an algorithm with access to such an oracle to be successful only if it works for any such ρ, which is a natural assumption that excludes from consideration "omniscient" algorithms that can distinguish between different solutions in $\Xi(\iota)$. Throughout the paper we will use the wording "with access to an exact solution oracle of precision ω" to mean the situation just described, and this concept will be formally treated in §9.11.

To avoid trivial examples we need another assumption. For instance, if Γ outputs values that do not belong to the range of the solution map Ξ, then one can easily construct examples where it becomes trivial to check if Γ fails. Concretely, if Γ produces correct output when $\Gamma(\iota) \in \Xi(\Omega)$ and wrong output when $\Gamma(\iota) = \xi \notin \Xi(\Omega)$, the exit flag is easy to compute: it is simply 0 when the output is ξ and 1 otherwise. A natural assumption precluding this situation could thus be that $\Gamma(\iota) \in \Xi(\Omega)$, for all $\iota \in \Omega$. This, however, would again create issues in the Turing model if $\Xi(\Omega)$ contained elements with irrational entries, and hence we introduce another precision parameter $\alpha > 0$ and make the following assumption:

$$\mathrm{dist}_\mathcal{M}(\Gamma(\iota), \Xi(\Omega)) < \alpha \ \text{ for all } \iota \in \Omega. \tag{5.1}$$

We are now ready to state the following result which shows that computing the exit flag is typically harder than the original problem.

Theorem 5.1 (Impossibility of computing the exit flag). *Let Ξ denote the solution map to any of the problems (1.1) - (1.4) with the regularisation parameters satisfying $\delta \in [0,1]$, $\lambda \in (0, 1/3]$, and $\tau \in [1/2, 2]$ (and additionally being rational in the Turing case) and consider the $\| \cdot \|_p$-norm for measuring the error, for an arbitrary $p \in [1, \infty]$. Let $K \in \mathbb{N}$ and fix real α and ω so that $0 < \alpha \leq \omega < 10^{-K}$. Then, for any fixed dimensions $N > m \geq 4$, there exists a class of inputs Ω for Ξ such that, if Γ is an algorithm satisfying (5.1) with parameter α for the computational problem of approximating Ξ with K correct digits, then we have the following.*

  (i) *No algorithm, even randomised with access to an exact solution oracle of precision ω, can compute the exit flag of Γ (with probability exceeding $\mathrm{p} > 1/2$ in the randomised case).*

 (ii) *If we allow randomised algorithms with non-zero probability of not halting (producing an output), then no such algorithm, even with access to an exact solution oracle of precision ω, can compute the exit flag of Γ with probability exceeding $\mathrm{p} > 1/2$.*

(iii) *The problem of computing the exit flag of Γ is strictly harder than computing K correct digits of Ξ in the following sense: if one is given the exit flag as an oracle then it is possible to construct an algorithm that computes K correct digits of Ξ. However, if one is instead given an oracle providing a K-digit approximation to Ξ, then it is still not possible to compute the exit flag of Γ.*

(iv) *For linear programming and basis pursuit, however, there exists a class of inputs $\Omega^\sharp \neq \Omega$ such that no algorithm, even randomised with non-zero probability of not halting, can compute the exit flag of Γ (with probability exceeding $\mathrm{p} > 1/2$ in the randomised case), yet one can compute the exit flag with a deterministic algorithm with access to an exact solution oracle of precision ω.*

The statements (i) and (ii) above are true even when we require the input in each Ω to be well-conditioned and bounded from above. In particular, for any input $\iota = (y, A) \in \Omega_{m,N}$ ($\iota = (y, A, c)$ in the case of LP) we have $\mathrm{Cond}(AA^*) \leq 3.2$, $C_{\mathrm{FP}}(\iota) \leq 4$, $\mathrm{Cond}(\Xi) \leq 179$, $\|y\|_\infty \leq 2$, and $\|A\|_{\max} = 1$.

The precise version of Theorem 5.1 can be found in Proposition 9.41.

Remark 5.2 (Halting vs non-halting algorithms). Note that (ii) in Theorem 5.1 directly implies (i) and is hence a stronger statement. However, we deliberately put both (i) and (ii) in the statement of the theorem to allow comparison with Theorem 3.4. Indeed, in Theorem 3.4 there is a difference in the lower bound for the probability between the case of a halting algorithm vs. a non-halting algorithm, which is not the case in Theorem 5.1.

6. Main Theorem III – The extended Smale's 9th problem and Kepler's conjecture

6.1. **Smale's 9th problem and the computational proof of Kepler's conjecture.** There is an important link between Smale's 9th problem and the recent proof of Kepler's conjecture [2], [3]. Kepler's conjecture, originally stated in 1611[45], hypothesised an optimality result for sphere packing in three dimensions. Continuing for nearly two decades, the program led by T. Hales [2], [3] concluded with a computer-assisted proof (the Flyspeck program) verifying Kepler's conjecture. The link to Smale's 9th problem is that Hales' program hinges on the numerical verification of more than fifty thousand decision problems of the following form: Given $K \in \mathbb{N} \cup \{\infty\}$ and $M \in \mathbb{R}$, does there exist an $x \in \mathbb{R}^N$ such that

$$\langle x, c \rangle_k \leq M \text{ subject to } Ax = y, \quad x \geq 0 \text{ coordinatewise,} \tag{6.1}$$

where $\langle x, c \rangle_K = \lfloor 10^K \langle x, c \rangle \rfloor 10^{-K}$? What makes this numerical verification delicate is that the inputs (y, A, c) are also approximated through numerical computation, and hence we find ourselves in the extended model. Informally, we could think of $\langle x, c \rangle_K$ as $\langle x, c \rangle$ computed to K correct digits (in the Flyspeck program (6.1) is computed with $K = 6$). The key question is now, is this problem in general decidable?

We will consider input sets of the form

$$\Omega = \bigcup_{m < N} \Omega_{m,N} \text{ so that } \quad \Xi_K : \Omega_{m,N} \to \{0, 1\} = \{\text{no, yes}\}, \tag{6.2}$$

where, for each m and N, $\Omega_{m,N}$ is a non-empty set of inputs of fixed dimensions m and N for the solution map Ξ_K of (6.1). Again, as in §3.2, we need to have inputs of arbitrarily large dimensions in order to be able to state "in P" results. We make the dependence of the solution map on K explicit, as we will consider (6.1) on the same input set and with the same M, but with different values of K.

Before discussing decidability, we note the following intricacy behind the choice of K. Suppose that the K-th digit behind the decimal point is 9. Then $\{x \in \mathbb{R}^N \mid \langle x, c \rangle_K \leq M\} = \{x \in \mathbb{R}^N \mid \langle x, c \rangle_{K-1} \leq M\}$, and hence $\Xi_K = \Xi_{K-1}$, i.e., the decision problem (6.1) is indistinguishable for K and $K - 1$. Now, writing $\mathcal{S}_k^9$ for the set of positive real numbers whose k-th digit after the decimal point is 9, we have the following theorem demonstrating the intricacy of the Smale's 9th problem in the extended model. This theorem shows that Smale's 9th problem in the extended model actually becomes a classification theory and testifies to the intricacy of the computational proof of Kepler's conjecture.

Theorem 6.1 (The extended Smale's 9th problem - deciding feasibility). *Let $M \geq 0$ be real and $K > 2$ an integer, and consider the decision problem (6.1) for this value of M and the corresponding solution maps Ξ_K, Ξ_{K-1}, and Ξ_{K-2}. Then there is a class of inputs Ω as in (6.2) such that we have the following:*

    (i) There does not exist any sequence of algorithms $\{\Gamma_j\}_{j \in \mathbb{N}}$ with output in $\{0, 1\}$ so that, for all $\iota \in \Omega$, we have $\lim_{j \to \infty} \Gamma_j(\iota) = \Xi_K(\iota)$ and, if $\Gamma_j(\iota) = 1$ for some $j \in \mathbb{N}$, then $\Xi_K(\iota) = \Gamma_j(\iota) = 1$.

    (ii) There does not exist a randomised algorithm Γ^{ran}, even with non-zero probability of not halting, such that, for all $\iota \in \Omega$, we have $\Gamma^{\mathrm{ran}}(\iota) = \Xi_K(\iota)$ with probability exceeding $\mathrm{p} > 1/2$.

Additionally, depending on whether M is in one (or both) of $\mathcal{S}_K^9$ or $\mathcal{S}_{K-1}^9$, one or both of the following hold as well:

(iii) *There does exist an algorithm Γ (Turing or BSS machine) such that, for all $\iota \in \Omega$, we have $\Gamma(\iota) = \Xi_{K-1}(\iota)$. However, any algorithm will need arbitrarily long time to compute Ξ_{K-1}. In particular, for any fixed dimensions m and N, any $T > 0$, and any algorithm Γ, there exists an input $\iota \in \Omega_{m,N}$ such that either $\Gamma(\iota) \neq \Xi_{K-1}(\iota)$ or the runtime of Γ on ι exceeds T. Moreover, for any randomised algorithm Γ^{ran} and $\mathrm{p} \in (0, 1/2)$, there exists an input $\iota \in \Omega_{m,N}$ such that*

$$\mathbb{P}\left(\Gamma^{\mathrm{ran}}(\iota) \text{ is wrong or the runtime exceeds } T\right) > \mathrm{p}.$$

(iv) *There exists a polynomial $\mathrm{pol} : \mathbb{R} \to \mathbb{R}$ as well as a Turing machine and a BSS machine that both compute Ξ_{K-2} for all inputs in Ω, so that the number of arithmetic operations for both machines is bounded by $\mathrm{pol}(n)$, where $n = m + mN$ is the number of variables, and the number of digits required from the oracle (2.2) is bounded by $\mathrm{pol}(\log(n))$. Moreover, the space complexity of the Turing machine is bounded by $\mathrm{pol}(n)$.*

Concretely, (iii) holds whenever $M \notin \mathcal{S}_K^9$, whereas (iv) holds whenever $M \notin \mathcal{S}_{K-1}^9$. Finally, if one only considers (i) - (iii), Ω can be chosen with any fixed dimensions $m < N$ with $N \geq 4$. Moreover, if one only considers (i) then K can be chosen to be 1.

The precise version of Theorem 6.1 can be found in Proposition 9.43.

6.2. **Undecidable problems in computer-assisted proofs: The proof of Kepler's conjecture.** The Flyspeck program is a stunning example of a successful computer-assisted proof of one of the few open questions spanning several centuries. However, Theorem 6.1 reveals that the computational proof is actually even more intricate than one might think. In fact, part (ii) of Theorem 6.1 demonstrates that the general decision problem is actually undecidable even for $K = 1$, and even in a randomised setting. Moreover, part (i) reveals that not only is it undecidable, but no sequence of algorithms can certify a positive answer to the decision problem. In the language of the Solvability Complexity Index (SCI) hierarchy (see §9.2), this means that the problem is not in Σ_1. This means that the problem can be viewed as strictly harder than the "halting problem", which can easily be seen to be in Σ_1. Given the results of Theorem 6.1, it may seem paradoxical that one could prove Kepler's conjecture through the Flyspeck program by trying to decide the decision problem Ξ_K with $K = 6$, which is done in the proof. There are other examples too of conjectures that are proven by means of computer-assisted proofs relying on problems that are non-computable in the general case, for instance the Dirac-Schwinger conjecture proved by C. Fefferman and L. Seco in [4]–[12] (see further discussion in [46]).

Theorem 6.1 shows that the Flyspeck program may have been impossible in several ways. Specifically, any of the following scenarios could have happened.

(1) Conditions of part (i) of Theorem 6.1 apply, i.e., the answer to all 50000 decision problems involved in the proof would have been *yes*, and hence Kepler's conjecture would have been true, yet the computations in the Flyspeck program would continue forever, regardless of the computing power, never producing the 50000 affirmative answers needed. Thus, it would never confirm Kepler's conjecture.

(2) Conditions of part (ii) of Theorem 6.1 apply instead of part (i). In particular, one might design a sequence of algorithms $\{\Gamma_j\}_{j\in\mathbb{N}}$ so that for any linear program input ι we have $\Gamma_j(\iota) \to \Xi_K(\iota)$ as $j \to \infty$ and, if $\Gamma_j(\iota) = 1$ for some $j \in \mathbb{N}$, then $\Gamma_j(\iota) = \Xi_K(\iota)$. Still, because of (ii), if one of the decision problems had a negative answer, the Flyspeck program would run forever and never conclude with an answer. Moreover, one could not even use randomised algorithms and conclude with a probability better than coin flipping.

(3) Conditions of part (iii) of Theorem 6.1 apply instead of parts (i) and (ii), i.e., there is an algorithm Γ such that for any linear program input ι we have $\Gamma(\iota) = \Xi_K(\iota)$. However, because of (iii) the

following could occur. All 50000 decision problems would have answer *yes*, yet it would take 10^{80} years for the Flyspeck program to conclude that Kepler's conjecture was true.

The scenario (1) can, in fact, always be avoided by considering different decision problems where one replaces $\langle x, c \rangle_K \leq M$ in (6.1) by $\langle x, c \rangle_K < M$. For this problem, a statement analogous to part (i) of Theorem 6.1 is no longer true. In fact, for every $K \in \mathbb{N}$, there does exist a sequence of algorithms $\{\Gamma_j\}_{j \in \mathbb{N}}$ such that for any linear program input ι we have $\Gamma_j(\iota) \to \Xi_K(\iota)$ as $j \to \infty$ and, if $\Gamma_j(\iota) = 1$ for some $j \in \mathbb{N}$, then $\Gamma_j(\iota) = \Xi_K(\iota)$. In the proof of Kepler's conjecture, all decision problems have the property that $\langle x, c \rangle_K < M$ for some feasible x (and the appropriate M). Moreover, these decision problems can be solved quickly, and one is in fact in conditions analogous to part (iv) of Theorem 6.1.

7. MAIN THEOREM IV – THE EXTENDED SMALE'S 9TH PROBLEM AND COMPRESSED SENSING

7.1. **The extended Smale's 9th problem and compressed sensing - Why things often work in practice.**
In view of Theorem 3.4, a natural question is the following: Under which conditions on the set of inputs may (1.2) in the extended model be in P? Obvious candidates for such conditions would be various standard assumptions on problems in compressed sensing and sparse regularisation, which we now recall. A fundamental concept of compressed sensing is sparsity, where we say that a vector x is s-sparse if it has at most s non-zero entries. Furthermore, a standard requirement is that the matrix A satisfy either the restricted isometry property or the robust nullspace property (RNP). We will only consider the latter. Concretely, a matrix $A \in \mathbb{C}^{m \times N}$ is said to satisfy the ℓ^2-*robust nullspace property of order s* with parameters $\rho \in (0, 1)$ and $\tau > 0$ if

$$\|v_S\|_2 \leq \frac{\rho}{\sqrt{s}}\|v_{S^c}\|_1 + \tau\|Av\|_2, \tag{7.1}$$

for all index sets $S \subseteq \{1, \ldots, N\}$ with $|S| \leq s$ (where $| \cdot |$ denotes the cardinality) and vectors $v \in \mathbb{C}^N$. The notation v_S means that the entries of v with indices in S^c are set to zero and the remaining entries are left unchanged. We can now make our question more specific.

Problem 3. *Given $K \in \mathbb{N}$, is the problem of computing K correct digits of a solution to the BP problem (1.2) in the extended model in P, assuming $y = Ax$, for some x guaranteed to be s-sparse, and A satisfies the ℓ^2-robust nullspace property of order s with parameters ρ and τ?*

As we will see below, this question is rather intricate since we again need a complexity theory for non-computable problems to provide a satisfactory answer. Namely, one cannot obtain $K = \lceil \log_{10}(2\delta^{-1}) \rceil$ correct digits for all inputs, yet the problem of getting $K = \lfloor \log_{10}\left((1 - \rho)(16\tau)^{-1}\delta^{-1}\right) \rfloor$ digits is in P. More precisely, we consider classes of inputs specified as follows. Fix real constants $\rho \in (1/3, 1)$, $\tau > 10$, $b_1 > 3$, and $b_2 > 6$, and, for $\epsilon \in [0, 1]$ and $s, m, N \in \mathbb{N}$ such that $m \leq N$ and $s \leq N$ define

$$\Omega_{s,m,N}^\epsilon = \{(y, A) \in \mathbb{R}^m \times \mathbb{R}^{m \times N} \,|\, (y, A) \text{ satisfies } (7.3)\} \tag{7.2}$$

where

$$A \text{ satisfies the RNP (7.1) of order } s \text{ with parameters } \rho \text{ and } \tau,$$

$$y = Ax \text{ for some } s\text{-sparse } x, \tag{7.3}$$

$$\|y\|_2 \leq b_1, \ \|A\|_2 \leq b_2\sqrt{N/m}.$$

Finally, set

$$\Omega^\epsilon = \bigcup_{\substack{s,m,N \in \mathbb{N} \\ m \leq N}} \Omega_{s,m,N}^\epsilon. \tag{7.4}$$

Then, defining $\log_{10}(0) = -\infty$ and recalling the definition of the C_{RCC} condition number (see §9.1), we have the following theorem.

Theorem 7.1. (The extended Smale's 9th problem and compressed sensing). *Let Ξ_{BP} and Ξ_{UL} denote the solution maps respectively to the ℓ^1-BP problem (1.2) and the ℓ^1-Lasso problem (1.3), and consider the $\|\cdot\|_2$-norm for measuring the error. If $\Omega_{s,m,N}^{\epsilon}$ and Ω^{ϵ} are as in (7.2) and (7.4) the following holds.*

(i) There exists a constant $C > 0$, independent of ρ, τ, b_1 and b_2, such that if we fix $s = 2^k$ for some $k \in \mathbb{N}$ and m, N satisfying $N > m \geq 7s$ and $m \geq Cs \log^2(2s) \log(N)$, then we have the following. For $\delta \in (0, 1]$ and

$$K \geq \left\lceil \log_{10}\left(2\delta^{-1}\right) \right\rceil,$$

there does not exist any algorithm (even randomised) that can produce K correct digits for Ξ_{BP} (with probability exceeding $\mathrm{p} > 1/2$ in the randomised case) for all inputs in $\Omega_{s,m,N}^0$.

(ii) Let s, m, and N be as in (i). Then, for $\lambda \in (0, 1/(2\sqrt{s})]$ and

$$K \geq \left\lceil \log_{10}\left(2s^{-1/2}\lambda^{-1}\right) \right\rceil,$$

the same impossibility result as in (i) holds for computing K correct digits for Ξ_{UL} for inputs in $\Omega_{s,m,N}^0$.

(iii) There exists a polynomial $\mathrm{pol} : \mathbb{R}^2 \to \mathbb{R}$ and, for every $\delta \in [0, (1-\rho)/(16\tau)]$, there exists an algorithm Γ_δ (either a Turing machine or a BSS machine) such that, for $K \in \mathbb{N}$ satisfying

$$K \leq \left\lfloor \log_{10}\left((1-\rho)(16\tau)^{-1}\delta^{-1}\right) \right\rfloor, \tag{7.5}$$

Γ_δ produces K correct digits for Ξ_{BP} for all inputs in Ω^δ. In the case that $\delta = 0$, (7.5) is to be interpreted as $K < \infty$. Moreover, the runtime of Γ_δ (steps performed by the Turing machine, arithmetic operations performed by the BSS machine) is bounded by $\mathrm{pol}(n, K)$, where $n = m + mN$ is the number of variables. In particular, for $\delta = 0$, the runtime is bounded by $\mathrm{pol}(n, K)$ for all $K \in \mathbb{N}$.

(iv) Consider the ℓ^1-BP problem (1.2) with $\delta = 0$. For any fixed $s \geq 3$, there are infinitely many pairs (m, N) and inputs $\iota = (Ax, A) \in \Omega^0$, where $x \in \mathbb{R}^m$ is s-sparse and $A \in \mathbb{R}^{m \times N}$ is a subsampled Hadamard, Bernoulli, or Hadamard-to-Haar matrix such that

$$C_{\mathrm{RCC}}(\iota) = \infty.$$

In particular, appreciating (iii) and (iv), there exist inputs in Ω with infinite RCC condition number, yet the problem is in P.

The precise version of Theorem 7.1 can be found in Proposition 9.36.

Theorem 7.1 can be viewed as the practical cousin of Theorem 3.4. Indeed, Theorem 7.1 demonstrates the facets of Theorem 3.4 in actual applications where the phenomenon described in Theorem 3.4 occur under usual conditions. However, it is important to emphasise that, while Theorem 7.1 and Theorem 3.4 explain why things fail, they also explain why things often work in practice. Specifically, the negative parts (i) and (ii) of Theorem 7.1 can be interpreted in conjunction with the positive parts (iii) and (iv) as follows: one may not be able to get, say, 5 digits of accuracy in certain cases, however, for sufficiently small values of δ, one may be able to get 4 digits, and that can be achieved quickly. As discussed in Remark 3.7, 4 digits of accuracy is enough for any application in imaging.

Part (iv) of Theorem 7.1 demonstrates how one can have tractable problems that are nonetheless ill-conditioned according to the RCC condition number. The fact that $C_{\mathrm{RCC}}(\iota) = \infty$ means that there are arbitrarily small perturbations that yield a problem with several minimisers. This is, however, not a problem when it comes to designing fast algorithms. Moreover, our examples are not contrived, but key examples from applications. Indeed, subsampled Hadamard matrices (where the rows are randomly sampled in a typical compressed sensing fashion) and Bernoulli matrices form the foundations of compressive imaging in microscopy, lensless camera, compressive video, etc. [47].

Remark 7.2. Note that (ii) in Theorem 7.1 only provides a lower bound for the Lasso problem. The "in P" statement and conditions on K are missing, which remains an open problem.

8. Smale's 18th problem and the paradoxes of deep learning

8.1. **Smale's 18th problem.** Smale's 18th problem, from the list of mathematical problems for the 21st century [1], echoes Turing's paper from 1950 [13] on the question of artificial intelligence. Turing asks if a computer can think, and suggests the imitation game as a test for his question about artificial intelligence. Smale takes the question even further and asks in his 18th problem:

What are the limits of intelligence, both artificial and human?

The question is followed by a discussion on the problem that ends as follows. *"Learning is a part of human intelligent activity. The corresponding mathematics is suggested by the theory of repeated games, neural nets and genetic algorithms."* Hence, given the unprecedented developments in deep learning and neural nets [28] over the last years and the impact these developments may have on artificial intelligence, it is timely to consider Smale's 18th problem in view of these advancements. Indeed, the latest deep learning results using neural networks on the image classification problem result in a failure rate below 3%, which is referred to as super human [48]. Moreover, methods based on deep learning provide by far the best results ever created for artificial intelligent image recognition, however, deep learning is empirically shown to be universally unstable. There is now an overwhelming literature documenting the universal instability phenomenon of deep learning. We will address Smale's 18th problem by demonstrating how instability becomes a key limitation in deep learning despite the existence of stable neural networks.

8.2. **Deep learning becomes unstable despite the existence of stable neural networks.** We consider the simplest deep learning problem of approximating a given classification function

$$f : D \subset \mathbb{R}^d \to \{0, 1\},$$

where D is some domain, by constructing a neural network from training data and testing it on a validation set. More precisely, given an input set Ω consisting of various $\iota = (\mathcal{T}, \mathcal{V})$, where the training set $\mathcal{T} = \{x^1, \ldots, x^r\}$ and validation set $\mathcal{V} = \{y^1, \ldots, y^s\}$ are finite subsets of $\mathbb{R}^d$, the solution map that we are asked to compute is given by

$$\Xi(\iota) = \left\{ \{\phi(y^j)\}_{j=1}^s \,|\, \phi \in \underset{\tilde{\phi} \in \mathcal{NN}_{\mathbf{N},L}}{\operatorname{argmin}} \mathcal{R}\left(\{\tilde{\phi}(x^j)\}_{j=1}^r, \{f(x^j)\}_{j=1}^r \right) \right\}, \qquad (8.1)$$

where $\iota = (\mathcal{T}, \mathcal{V})$, the set $\mathcal{NN}_{\mathbf{N},L}$ with $\mathbf{N} := (N_L = 1, N_{L-1}, \ldots, N_1, N_0 = d)$ denotes the set of all L-layer neural networks under the ReLU nonlinearity with N_ℓ neurons in the ℓ-th layer, and the cost function $\mathcal{R}$ is assumed to be an element of

$$\mathcal{CF}_r = \{\mathcal{R} : \mathbb{R}^r \times \mathbb{R}^r \to \mathbb{R}_+ \,|\, \mathcal{R}(v, w) = 0 \text{ iff } v = w\}. \qquad (8.2)$$

As we will discuss the stability of neural networks, we introduce the idea of *well-separated and stable sets* to exclude pathological examples whereby training and validation sets have elements that are arbitrarily close to each other in a way that could make the classification function jump subject to a small perturbation. Specifically, given a classification function $f : D \subset \mathbb{R}^d \to \{0, 1\}$, we define the family of well-separated and stable sets $\mathcal{S}_\delta^f$ with separation at least 2δ according to

$$\mathcal{S}_\delta^f = \{\{x^1, \ldots, x^r\} \subset D \,|\, \|x^j\|_\infty \le 1, \min_{x^i \ne x^j} \|x^i - x^j\|_\infty \ge 2\delta, f(x^j + y) = f(x^j) \text{ for } \|y\|_\infty < \delta\}.$$

Our first theorem addresses the instability of trained neural networks, and does not rely on the extended model (i.e., training with inexact input).

Theorem 8.1 (Instability of trained NNs despite existence of a stable NN). *There is an uncountable family of classification functions $f : \mathbb{R}^d \to \{0, 1\}$ with $d \geq 2$ and a constant $C > 0$ such that we have the following. For every $\epsilon > 0$, there is an uncountable family of distributions $\mathcal{D}$ on $[0, 1]^d$ so that for any neural network dimensions $\mathbf{N} = (N_L = 1, N_{L-1}, \ldots, N_1, N_0 = d)$ with $L \geq 2$, any $\mathrm{p} \in (0, 1)$, any positive integers s, r with*

$$r \geq \max \left\{ C \, \mathrm{p}^{-3}, C\big[(N_1 + 1) \cdots (N_{L-1} + 1)\big]^{3/2} \right\}, \tag{8.3}$$

and training and validation data $\mathcal{T} = \{x^1, \ldots, x^r\}$ and $\mathcal{V} = \{y^1, \ldots, y^s\}$, where the x^j and y^j are drawn independently at random from $\mathcal{D}$, the following happens with probability exceeding $1 - \mathrm{p}$.

We have $\mathcal{T}, \mathcal{V} \in \mathcal{S}^f_{\varepsilon((r \vee s)/\mathrm{p})}$, where $\varepsilon(n) = (Cn)^{-4}$, and, for every $\mathcal{R} \in \mathcal{CF}_r$, there exists a ϕ such that

$$\phi \in \underset{\tilde{\phi} \in \mathcal{NN}_{\mathbf{N},L}}{\mathrm{argmin}} \mathcal{R}\left(\{\tilde{\phi}(x^j)\}^r_{j=1}, \{f(x^j)\}^r_{j=1}\right)$$

and

$$\phi(x) = f(x) \quad \forall x \in \mathcal{T} \cup \mathcal{V}. \tag{8.4}$$

Yet, for any $\hat{\phi} \in \mathcal{NN}_{\mathbf{N},L}$ (and thus, in particular, for $\hat{\phi} = \phi$), there exist uncountably many $v \in \mathbb{R}^d$ such that

$$|\hat{\phi}(v) - f(v)| \geq 1/2 \quad and \quad \|v - x\|_\infty < \epsilon \qquad for \ some \ x \in \mathcal{T}. \tag{8.5}$$

However, there exists a (stable) neural network $\psi \notin \mathcal{NN}_{\mathbf{N},L}$ that satisfies $\psi(x) = f(x)$ for all $x \in \mathcal{B}^\infty_{\varepsilon((r \vee s)/\mathrm{p})}(\mathcal{T} \cup \mathcal{V})$.

Remark 8.2. The training and validation data $\mathcal{T}$ and $\mathcal{V}$ in Theorem 8.1 are technically not sets, but randomised multisets, as some of the samples x^j or y^j may be repeated.

The second theorem says that even validating a trained neural network on the training set itself is in general uncomputable in the extended model. Concretely, for a fixed training set $\mathcal{T} = \{x^1, \ldots, x^r\} \subset \mathbb{R}^d$, define the input domain

$$\Omega_{\mathcal{T},\nu} = \left\{ \iota = \left(\{\hat{x}^j\}^r_{j=1}, \{\hat{x}^j\}^r_{j=1}\right) \mid \hat{x}^j \in \mathcal{B}^\infty_\nu(x^j), \, \forall x^j \in \mathcal{T} \right\}, \tag{8.6}$$

where $\mathcal{B}^\infty_\nu(x^j)$ denotes the open ball of radius $\nu > 0$ in the ℓ^∞ norm centred at x^j.

Theorem 8.3 (Non-existence of algorithms for deep learning). *There is an uncountable family of classification functions $f : \mathbb{R}^d \to \{0, 1\}$ with $d \geq 2$ so that, for any neural network dimensions $\mathbf{N} = (N_L = 1, N_{L-1}, \ldots, N_1, N_0 = d)$ with $L \geq 2$ and any $r \geq 3(N_1 + 1) \cdots (N_{L-1} + 1)$, there exist uncountably many training sets $\mathcal{T} = \{x^1, \ldots, x^r\} \in \mathcal{S}^f_{\varepsilon'(r)}$ of size r (where $\varepsilon'(n) := [(4n + 3)(4n + 4)]^{-1}$) such that we have the following. Given any cost function $\mathcal{R} \in \mathcal{CF}_r$ and $\nu < \varepsilon(r)$, there does not exist any algorithm, even randomised with non-zero probability of not halting, that can compute*

$$\{\phi(x^j)\}^r_{j=1} \quad for \ some \quad \phi \in \underset{\tilde{\phi} \in \mathcal{NN}_{\mathbf{N},L}}{\mathrm{argmin}} \mathcal{R}\left(\{\tilde{\phi}(x^j)\}^r_{j=1}, \{f(x^j)\}^r_{j=1}\right) \tag{8.7}$$

to one digit accuracy (with error measured in the ℓ^∞ norm) for all inputs in $\Omega_{\mathcal{T},\nu}$ (as given in (8.6)) with probability greater than $\mathrm{p} > 1/2$.

The precise version of Theorem 8.3 can be found in Proposition 9.44.

Remark 8.4 (Failure on the training set). One may be puzzled by (8.7) and the contrast to the computational problem (8.1) that deals with the problem of computing $\{\phi(y^j)\}^s_{j=1}$ on the validation set $\mathcal{V} = \{y^1, \ldots, y^s\}$. However, this is of no consequence, as Theorem 8.3 simply says that the process of constructing the neural network through the optimisation problem (8.1) and evaluating it even on the training set only is non-computable. In fact, this cannot be done even with one digit accuracy with a probability better than a coin flip.

Remark 8.5 (Irrelevance of local minima). Note that Theorem 8.3 has nothing to do with the potential issue of the optimisation problem having several local minima. Indeed, Theorem 8.3 is true for a general algorithm according to Definition 9.3 in §9, which is more powerful than any Turing machine or BSS machine, or even such a machine with access to an oracle that provides a minimiser $\phi \in \operatorname{argmin}_{\tilde{\phi} \in \mathcal{NN}_{\mathbf{N},L}} \mathcal{R}\left(\{\tilde{\phi}(x^j)\}_{j=1}^r, \{f(x^j)\}_{j=1}^r\right)$ for every inexact input provided to the algorithm.

Remark 8.6 (The fixed-architecture instability problem). Theorem 8.1 demonstrates the delicate issue of the instability problem. The key is that as long as the dimensions of the network is fixed, one can get 100% success rate on the trained network, however, it will become arbitrarily unstable (even around the training set), as will other networks of the same dimensions. Moreover, as the last statement of Theorem 8.1 reveals, if one changes the dimensions, then one can again find a stable network on the given training and classification test data, however, there is no recipe on how to construct it. In fact, Theorem 8.3 demonstrates that even if one knew the dimensions of the stable neural network, producing such a neural network through the standard optimisation may be non-computable. Furthermore, Theorem 8.1 may be applied again with the architecture of the stable network suggesting that the same architecture with a different set of input data will again lead to an unstable network.

9. The SCI hierarchy – Mathematical preliminaries for the proofs

In this section we present outlines of the proofs of the main theorems and how to navigate through the manuscript. In addition we present the mathematical preliminaries needed. We begin by giving a brief account of the new concepts introduced in the paper in order to prove the theorems. Note that the main abstract results that eventually lead to the proofs of the main theorems are deliberately made as general as possible as these techniques can be used for many other types of problems in the future. Moreover, the techniques used in proofs as well as the mathematical preliminaries are based on the recent program on the Solvability Complexity Index (SCI) hierarchy [49]–[51].

9.1. **Condition - precise definitions.** We recall the standard definitions of condition used in optimisation [52], [53], [42] [54]. The classical condition number of a matrix A is given by

$$\operatorname{Cond}(A) = \|A\|_2 \|A^{-1}\|_2.$$

For different types of condition numbers related to a mapping $\Xi : \Omega \subset \mathbb{C}^n \to \mathbb{C}^m$ we need to establish what types of perturbations we are interested in. For example, if Ω denotes the set of diagonal matrices, 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

$$\mathcal{A}(\Omega) = \{j \in \{1, 2, \ldots, n\} \mid \exists x, y \in \Omega, x_j \neq y_j\},$$

which we refer to as the active set as well as

$$\Omega^{\text{act}} = \{x \in \mathbb{R}^n \mid \exists y \in \Omega \text{ such that } x_{\mathcal{A}^c} = y_{\mathcal{A}^c}\}, \quad \mathcal{A} = \mathcal{A}(\Omega). \tag{9.1}$$

In particular, Ω^{act} describes the perturbations allowed as dictated by the active set of coordinates, namely the perturbations along the non-constant coordinates of elements in Ω. We can now recall some of the classical condition numbers from the literature [54].

We begin with *condition of a mapping*. Let $\Xi : \Omega \subset \mathbb{R}^n \to \mathbb{R}^m$ be a linear or non-linear mapping, and suppose that Ξ is also defined on some set $\hat{\Omega}$ with $\Omega \subset \hat{\Omega}$. Then,

$$\operatorname{Cond}(\Xi) = \sup_{x \in \Omega} \lim_{\epsilon \to 0^+} \sup_{\substack{x+z \in \Omega^{\text{act}} \cap \hat{\Omega} \\ 0 < \|z\|_\infty \leq \epsilon}} \left\{ \frac{\operatorname{dist}_\infty(\Xi(x+z), \Xi(x))}{\|z\|_\infty} \right\}, \tag{9.2}$$

where we allow for multivalued functions by defining $\mathrm{dist}_\infty(\Xi(x), \Xi(z)) = \inf_{x' \in \Xi(x), z' \in \Xi(z)} \|x' - z'\|_\infty$ (here, the supremum of the empty set is understood to be 0). We typically choose $\hat{\Omega}$ to be the largest set on which Ξ can be defined in a natural fashion.

Next we have *distance to infeasibility* and the *Feasibility Primal* condition number, defined for both Linear Programming (1.1) and Basis Pursuit (1.2). Specifically, we define the set of *infeasible inputs* (which we denote by Σ^{FP}) for a linear program to be the set of (y, A) such that no x exists with $x \geq 0$ and $Ax = y$. Similarly, for basis pursuit denoising with parameter $\delta \geq 0$, we define the set of infeasible inputs (which we also denote, with a slight abuse of notation, by Σ^{FP}) as the set of inputs for which no x exists with $\|Ax - y\|_2 \leq \delta$.

We then define the *Feasibility Primal* (FP) condition number for either basis pursuit or linear programming with inputs in Ω according to

$$C_{\mathrm{FP}}(y, A) := \begin{cases} \frac{\|y\|_2 \vee \|A\|_2}{\mathrm{dist}_2[(y,A),\Sigma^{\mathrm{FP}} \cap \Omega^{\mathrm{act}}]} & \text{if } \mathrm{dist}_2\left[(y,A), \Sigma^{\mathrm{FP}} \cap \Omega^{\mathrm{act}}\right] \neq 0 \\ \infty & \text{otherwise} \end{cases} \tag{9.3}$$

where we use the standard $\vee, \wedge$ as notation for max/min respectively and where, for a pair $(y, A) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}$ and a set of pairs $S \subseteq \mathbb{R}^m \times \mathbb{R}^{m \times N}$, dist_2 is defined in the following way:

$$\mathrm{dist}_2[(y, A), S] = \inf\{\|y' - y\|_2 \vee \|A' - A\|_2 \mid (y', A') \in S\}.$$

Here, the infimum of the empty-set is understood to be ∞.

Note that UL (1.3), CL (1.4) and the neural network (1.6) problem are always feasible, and hence the Feasibility Primal condition number is not defined for these problems.

Finally, we have the *distance to inputs with several minimisers* and the *RCC condition number*: For any of problems (1.1) to (1.4), we define the set of inputs with several minimisers (which we denote by Σ^{RCC}) to be the set of pairs (y, A) such that the problem with input (y, A) has at least two distinct solutions. We then define the *RCC condition number* according to

$$C_{\mathrm{RCC}}(y, A) := \begin{cases} \frac{\|y\|_2 \vee \|A\|_2}{\mathrm{dist}_2[(y,A),\Sigma^{\mathrm{RCC}} \cap \Omega^{\mathrm{act}}]} & \text{if } \mathrm{dist}_2\left[(y,A), \Sigma^{\mathrm{RCC}} \cap \Omega^{\mathrm{act}}\right] \neq 0 \\ \infty & \text{otherwise} \end{cases}. \tag{9.4}$$

Remark 9.1 (RCC condition number). Note that the definition of the RCC condition number here is slightly weaker than the one in [30], that is to say, our RCC condition number is less than or equal to the one defined in [30]. However, our weaker definition simply makes our results stronger in that we show that $C_{\mathrm{RCC}}(y, A) = \infty$ yet the problem of computing minimisers is still in P. We have also split the definition in [30] into two parts - one for feasibility, and one for two minimisers. This allows for a more granular analysis of the influence of condition.

9.2. **SCI hierarchy.** To formalise our results and the underlying theory we recall the definitions from [46] and start by defining a *computational problem*.

Definition 9.2 (Computational problem). Let Ω be some set, which we call the *input* set, 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 *solution map*. We call the collection $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ a *computational problem*. When it is clear what $\mathcal{M}$ and Λ are we will sometimes write $\{\Xi, \Omega\}$ for brevity.

The set Ω is essentially the set of objects that give rise to the various instances of our computational problem. It can be a family of matrices (infinite or finite) and vectors, a collection of polynomials, a family of Schrödinger operators with a certain potential etc. The solution map $\Xi : \Omega \to \mathcal{M}$ is what we are interested in computing. It could be one of the problems in (1.1), the set of eigenvalues of an

$n \times n$ matrix, the spectrum of a Hilbert (or Banach) space operator, root(s) of a polynomial, etc. Finally, the set Λ is the collection of functions that provide us with the information we are allowed to read, say matrix elements and vector coefficients, polynomial coefficients or pointwise values of the potential of a Schrödinger operator, for example.

9.2.1. *Algorithms and tower of algorithms.* The cornerstone of the SCI framework is the definition of a general algorithm, introduced next.

Definition 9.3 (General Algorithm). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, a *general algorithm* is a mapping $\Gamma : \Omega \to \mathcal{M} \cup \{\mathrm{NH}\}$ such that, for every $\iota \in \Omega$, the following conditions hold:

(i) there exists a nonempty subset of evaluations $\Lambda_\Gamma(\iota) \subset \Lambda$, and, whenever $\Gamma(\iota) \neq \mathrm{NH}$, we have $|\Lambda_\Gamma(\iota)| < \infty$,

(ii) the action of Γ on ι is uniquely determined by $\{f(\iota)\}_{f \in \Lambda_\Gamma(\iota)}$,

(iii) for every $\iota' \in \Omega$ such that $f(\iota') = f(\iota)$ for all $f \in \Lambda_\Gamma(\iota)$, it holds that $\Lambda_\Gamma(\iota') = \Lambda_\Gamma(\iota)$.

Remark 9.4 (The purpose of a general algorithm: universal impossibility results). The purpose of a general algorithm is to have a definition that will encompass any model of computation, and that will allow impossibility results to become universal. Given that there are several non-equivalent models of computation, impossibility results will be shown with this general definition of an algorithm.

Remark 9.5 (The non-halting output NH). The non-halting "output" NH of a general algorithm may seem like an unnecessary distraction given that a general algorithm is just a mapping, which is strictly more powerful than a Turing or a BSS machine. However, the NH output is needed when the concept of a general algorithm is extended to a randomised general algorithm (as done in §9.5) needed to prove Theorem 3.4. Potentially surprisingly, as shown in Theorem 3.4, allowing a randomised algorithm to not halt with positive probability makes it more powerful. Note, however, that adding the NH as a possible output of a general algorithm does not make any results weaker. A technical remark about NH is also appropriate, namely that $\Lambda_\Gamma(\iota)$ is allowed to be infinite in the case when $\Gamma(\iota) = \mathrm{NH}$. This is to allow general algorithms to capture the behaviour of a Turing or a BSS machine not halting by virtue of requiring an infinite amount of input information.

Owing to the presence of the special non-halting "output" NH, we have to extend the metric $d_\mathcal{M}$ on $\mathcal{M} \times \mathcal{M}$ to $d_\mathcal{M} : \mathcal{M} \cup \{\mathrm{NH}\} \times \mathcal{M} \cup \{\mathrm{NH}\} \to \mathbb{R}_{\geq 0}$ in the following way:

$$d_\mathcal{M}(x,y) = \begin{cases} d_\mathcal{M}(x,y) & \text{if } x,y \in \mathcal{M} \\ 0 & \text{if } x = y = \mathrm{NH} \\ \infty & \text{otherwise.} \end{cases} \tag{9.5}$$

Definition 9.3 is sufficient for defining a randomised general algorithm, which is the only tool from the SCI theory needed in order to prove Theorem 3.4. However, for several other results, the full definition of the SCI hierarchy will be useful. The first is the concept of a tower of algorithms, which is important in the general SCI theory in order to allow for problems that require several limits, such as spectral problems for operators on Hilbert spaces.

Definition 9.6 (Tower of algorithms). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, a *general tower of algorithms of height k for* $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ is a family of sequences of functions

$$\Gamma_{n_k} : \Omega \to \mathcal{M}, \ \Gamma_{n_k, n_{k-1}} : \Omega \to \mathcal{M}, \dots \Gamma_{n_k, \dots, n_1} : \Omega \to \mathcal{M},$$

where $n_k, \ldots, n_1 \in \mathbb{N}$ and the functions $\Gamma_{n_k,\ldots,n_1}$ are general algorithms in the sense of Definition 9.3. Moreover, for every $\iota \in \Omega$,

$$\Xi(\iota) = \lim_{n_k \to \infty} \Gamma_{n_k}(\iota), \quad \Gamma_{n_k,\ldots,n_{j+1}}(\iota) = \lim_{n_j \to \infty} \Gamma_{n_k,\ldots,n_j}(\iota) \quad j = k-1, \ldots, 1. \tag{9.6}$$

Remark 9.7 (Multivalued functions). When dealing with optimisation problems one needs a framework that can handle multiple solutions. As the setup above does not allow Ξ to be multi-valued we need some slight changes. We allow Ξ to be multivalued, even though towers of algorithms are not. Hence, the only difference to the standard SCI hierarchy is that the first limit in (9.6) is replaced by

$$\mathrm{dist}_{\mathcal{M}}(\Xi(\iota), \Gamma_{n_k}(\iota)) \longrightarrow 0, \qquad n_k \to \infty,$$

where $\mathrm{dist}_{\mathcal{M}}(\Xi(\iota), \Gamma_{n_k}(\iota)) := \inf_{x \in \Xi(\iota)} d_{\mathcal{M}}(x, \Gamma_{n_k}(\iota))$.

As already mentioned above, the purpose of a general algorithm is to obtain universal impossibility results. Conversely, for a more granular analysis of the complementary positive results, i.e., the analysis of problems where it is possible to construct an algorithm, we define *arithmetic towers* depending on the underlying computational model.

Definition 9.8 (Arithmetic towers). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where Λ is countable, an *arithmetic tower of algorithms of height k* for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ is defined as a tower of algorithms whose lowest-level functions $\Gamma = \Gamma_{n_k,\ldots,n_1} : \Omega \to \mathcal{M}$ satisfy the following: For each $\iota \in \Omega$ the mapping $(n_k, \ldots, n_1) \mapsto \Gamma_{n_k,\ldots,n_1}(\iota) = \Gamma_{n_k,\ldots,n_1}(\{f(\iota)\}_{f \in \Lambda})$ is recursive, and $\Gamma_{n_k,\ldots,n_1}(\iota)$ is a finite string of complex numbers that can be identified with an element in $\mathcal{M}$.

Remark 9.9 (Recursiveness). For the positive results we will always construct *recursive* algorithms, the meaning of which depends on whether one is working with the Turing or the BSS model. In the Turing case, recursive means that $f(\iota) \in \mathbb{Q}$, for all $f \in \Lambda$ and $\iota \in \Omega$, Λ is countable, and $\Gamma_{n_k,\ldots,n_1}(\{f(\iota)\}_{f \in \Lambda})$ can be executed by a Turing machine [44], that takes $(n_k, \ldots, n_1)$ as input, and that has an oracle tape providing $f(\iota)$ for every $f \in \Lambda$, see for example [55]. In the BSS model, recursive means that $f(\iota) \in \mathbb{R}$ (or $\mathbb{C}$) for all $f \in \Lambda$, and $\Gamma_{n_k,\ldots,n_1}(\{f(\iota)\}_{f \in \Lambda})$ can be executed by a Blum-Shub-Smale (BSS) machine [42] that takes $(n_k, \ldots, n_1)$, as input, and that has an oracle node that provides $f(\iota)$, for every $f \in \Lambda$. See also Remark 9.16.

9.2.2. *Solvability Complexity Index and Hierarchy.* With the concept of towers of algorithms established we can finally define the Solvability Complexity Index (SCI).

Definition 9.10 (Solvability Complexity Index). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, it is said to have *Solvability Complexity Index* $\mathrm{SCI}(\Xi, \Omega, \mathcal{M}, \Lambda)_\alpha = k$ of type α, where $\alpha = G$ (for 'general') or $\alpha = A$ (for 'arithmetic'), if k is the smallest integer for which there exists a tower of algorithms of height k and type α. If no such tower exists then $\mathrm{SCI}(\Xi, \Omega, \mathcal{M}, \Lambda)_\alpha = \infty$. If there exists such a tower $\{\Gamma_n\}_{n \in \mathbb{N}}$ of height 1, and additionally $\Xi = \Gamma_{n_1}$ for some $n_1 < \infty$, then we define $\mathrm{SCI}(\Xi, \Omega, \mathcal{M}, \Lambda)_\alpha = 0$.

The Solvability Complexity Index induces the SCI hierarchy. The key is that this hierarchy does not collapse regardless of the computational model [46]. Thus, it provides a universal framework for classifying problems in scientific computing.

Definition 9.11 (The Solvability Complexity Index Hierarchy). Consider a collection $\mathcal{C}$ of computational problems and a type $\alpha = A$ or $\alpha = G$ for the computational problems in $\mathcal{C}$. Define

$$\Delta_0^\alpha := \{\{\Xi, \Omega\} \in \mathcal{C} \mid \mathrm{SCI}(\Xi, \Omega)_\alpha = 0\}$$
$$\Delta_{m+1}^\alpha := \{\{\Xi, \Omega\} \in \mathcal{C} \mid \mathrm{SCI}(\Xi, \Omega)_\alpha \leq m\}, \qquad \forall m \in \mathbb{N},$$

as well as

$$\Delta_1^\alpha := \{\{\Xi, \Omega\} \in \mathcal{C} \mid \exists \{\Gamma^n\}_{n \in \mathbb{N}} \text{ tower of type } \alpha \text{ s.t. } \forall n \; \mathrm{dist}_\mathcal{M}(\Gamma^n(\iota), \Xi(\iota)) \le 2^{-n}\}.$$

9.2.3. *The computational problems.* We are now ready to formalise the problems of interest (1.1)-(1.6) as computational problems in the sense of Definition 9.2 for fixed dimensions m and N. In all the problems under consideration we consider $(\mathcal{M}, d) = (\mathbb{R}^m \cup \{\infty\}, \|\cdot\|)$, where the norm depends on the different problems. In all cases except for neural networks, the set Ω will always contain vectors and matrices, and thus the set of evaluations will naturally be the collection of the coordinate functions. Concretely, we will often consider input sets Ω whose elements are of the form $\iota = (y, A)$ with $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times N}$, in which case we let

$$\Lambda_{m,N} = \{f_j^{\mathrm{vec}}\}_{j=1}^m \cup \{f_{j,k}^{\mathrm{mat}}\}_{j=1,k=1}^{j=m,k=N},$$

where $f_j^{\mathrm{vec}}(\iota) = y_j$ and $f_{j,k}^{\mathrm{mat}}(\iota) = A_{j,k}$. In the following we write $\mathcal{J}$ for $\|\cdot\|_1$ or $\|\cdot\|_{\mathrm{TV}}$ depending on the context.

(i) *Linear Programming*: $\iota \in \Omega$ is of the form $\iota = (y, A)$, where $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times N}$. For fixed $c \in \mathbb{R}^N$ the solution map is given by

$$\Xi(\iota) = \operatorname*{argmin}_{x \in \mathbb{R}^N} \langle x, c \rangle \text{ such that } Ax = y, \quad x \ge 0.$$

(ii) Basis Pursuit: $\iota \in \Omega$ is of the form $\iota = (y, A)$ where $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times N}$. For a fixed parameter $\delta \ge 0$ the solution map is given by

$$\Xi(\iota) = \begin{cases} \operatorname*{argmin}_{x \in \mathbb{R}^N} \mathcal{J}(x) \text{ such that } \|Ax - y\|_2 \le \delta, & \text{if } \iota \text{ is feasible} \\ \varnothing & \text{else} \end{cases}.$$

(iii) *Constrained Lasso*: Ω consists of inputs of the same form as for basis pursuit. For a fixed parameter $\tau \ge 0$, the solution map is

$$\Xi(\iota) = \operatorname*{argmin}_{x \in \mathbb{R}^N} \|Ax - y\|_2 \text{ such that } \|x\|_1 \le \tau.$$

(iv) *Unconstrained Lasso*: Ω consists of inputs of the same form as for basis pursuit. For a fixed parameter $\lambda > 0$, the solution map is

$$\Xi(\iota) = \operatorname*{argmin}_{x \in \mathbb{R}^N} \|Ax - y\|_2^2 + \lambda \, \mathcal{J}(x).$$

(v) *Neural Networks (Deep Learning)*: We only consider lower bounds and thus concentrate on the simplest classification functions $f : \mathbb{R}^d \to \{0, 1\}$. Given such f and a training set $\mathcal{T} \subset \mathbb{R}^d$, the task is to compute a neural network $\phi \in \mathcal{NN}_{\mathbf{N}, L, d}$ and evaluate it on a validation set $\mathcal{V}$. We fix s, let $(\mathcal{M}, d) = (\mathbb{R}^s, \|\cdot\|_\infty)$ and consider input sets Ω consisting of various $\iota = (\mathcal{T}, \mathcal{V})$, where $\mathcal{T} = \{x^1, \ldots, x^r\}$ and $\mathcal{V} = \{y^1, \ldots, y^s\}$ are finite subsets of $\mathbb{R}^d$. The set of evaluations is given by the coordinates of the training and validation points, i.e.,

$$\Lambda_{d,r,s} = \{f_{j,k}^{\mathcal{T}}\}_{j=1,k=1}^{j=d,k=r} \cup \{f_{j,k}^{\mathcal{V}}\}_{j=1,k=1}^{j=d,k=s}, \tag{9.7}$$

where $f_{j,k}^{\mathcal{T}}(\iota) = x_j^k$ and $f_{j,k}^{\mathcal{V}}(\iota) = y_j^k$, and finally, for a cost function $\mathcal{R} \in \mathcal{CF}_r$, where $\mathcal{CF}_r$ is defined in (8.2), the solution map of interest is given by (8.1).

9.3. **Inexact input and perturbations.** Suppose we are given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, and that $\Lambda = \{f_j\}_{j \in \beta}$, where β is some index set that can be finite or 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 $f_j(\iota)$, but rather $f_{j,n}(\iota)$ where $f_{j,n}(\iota) \to f_j(\iota)$ as $n \to \infty$. In this paper we will be interested in the case when this can be done with error control. In particular, we consider $f_{j,n} : \Omega \to \mathbb{D}_n + i\mathbb{D}_n$, where $\mathbb{D}_n := \{k \, 2^{-n} \mid k \in \mathbb{Z}\}$, such that

$$\|\{f_{j,n}(\iota)\}_{j \in \beta} - \{f_j(\iota)\}_{j \in \beta}\|_\infty \leq 2^{-n}, \quad \forall \iota \in \Omega. \tag{9.8}$$

By analogy with Δ_1 classification, we will call a collection of such functions Δ_1-information for the computational problem. Formally, we have the following.

Definition 9.12 (Δ_1-information). Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem with $\Lambda = \{f_j\}_{j \in \beta}$. Suppose that, for each $j \in \beta$ and $n \in \mathbb{N}$, there exists an $f_{j,n} : \Omega \to \mathbb{D}_n + i\mathbb{D}_n$ such that (9.8) holds. We then say that the set $\hat{\Lambda} = \{f_{j,n} \mid j \in \beta, n \in \mathbb{N}\}$ provides Δ_1-information for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$. Moreover, we denote the family of all such $\hat{\Lambda}$ by $\mathcal{L}^1(\Lambda)$.

Remark 9.13. One can take the analogy with Δ_1 classification in the SCI hierarchy even further by considering f_j that are higher up in the hierarchy, and analogously define Δ_m-information for $m > 1$ (see [46]). However, this is beyond the scope of the present paper.

Note that we want to have algorithms that can the computational problems $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ for all possible choices of $\hat{\Lambda} \in \mathcal{L}^1(\Lambda)$. In order to formalise this we define what we mean by a computational problem with Δ_1-information.

Definition 9.14 (Computational problem with Δ_1-information). Given $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ with $\Lambda = \{f_j\}_{j \in \beta}$, the corresponding *computational problem with Δ_1-information* is defined as

$$\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} := \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\},$$

where

$$\tilde{\Omega} = \left\{ \tilde{\iota} = \left\{(f_{j,1}(\iota), f_{j,2}(\iota), f_{j,3}(\iota), \dots)\right\}_{j \in \beta} \mid \iota \in \Omega, \ f_{j,n} : \Omega \to \mathbb{D}_n + i\mathbb{D}_n \text{ satisfy (9.8)} \right\}, \tag{9.9}$$

$\tilde{\Xi}(\tilde{\iota}) = \Xi(\iota)$, and $\tilde{\Lambda} = \{\tilde{f}_{j,n}\}_{j,n \in \beta \times \mathbb{N}}$, where $\tilde{f}_{j,n}(\tilde{\iota}) = f_{j,n}(\iota)$. Given an $\tilde{\iota} \in \tilde{\Omega}$, there is a unique $\iota \in \Omega$ for which $\tilde{\iota} = \left\{(f_{j,1}(\iota), f_{j,2}(\iota), f_{j,3}(\iota), \dots)\right\}_{j \in \beta}$ (by Definition 9.2). We say that this $\iota \in \Omega$ *corresponds* to $\tilde{\iota} \in \tilde{\Omega}$.

Remark 9.15. Note that the correspondence of a unique ι to each $\tilde{\iota}$ in Definition 9.14 ensures that $\tilde{\Xi}$ and the elements of $\tilde{\Lambda}$ are well-defined.

One may interpret the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ as follows. The collection $\tilde{\Omega}$ is the family of all sequences approximating the inputs in Ω. For an algorithm to be successful for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$ it must work for all $\tilde{\iota} \in \tilde{\Omega}$, that is, for any sequence approximating ι, as opposed to a particular choice of Δ_1-information for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ according to Definition 9.12. The relation between these two closely related concepts will be further elucidated in §9.6 below.

Remark 9.16 (Oracle tape/node providing Δ_1-information). For impossibility results we use general algorithms and randomised general algorithms (as defined below), and thus, due to their generality, we do not need to specify how the algorithms read the information. However, all positive results are done for problems $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} := \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ with either a Turing or a BSS machine, and we thus need to specify how $\tilde{\iota} \in \tilde{\Omega}$ in (9.9) is passed to the algorithm as an input. Suppose that β, the index set for

Λ, is countable. In the Turing case we follow the standard convention in the literature (see for example [55]), in particular, $\tilde{\iota}$ is represented by an oracle tape that, on input $(j, n) \in \beta \times \mathbb{N}$ (where j is an integer or otherwise encoded in a finite alphabet), prints the (unique) finite binary string representing the dyadic number $\tilde{f}_{j,n}(\tilde{\iota})$. Similarly, for a BSS machine, we assume the standard setup (see [42]) with an oracle node that on input $(j, n) \in \beta \times \mathbb{N}$ returns $\tilde{f}_{j,n}(\tilde{\iota})$.

9.4. **Breakdown-epsilons – the key to proving K, $K-1$, $K-2$ -type theorems.** The purpose of breakdown epsilons is to characterise the lower bounds on the accuracy that can be achieved by algorithms. As the name suggests, the breakdown epsilons are the fundamental barriers on the best possible accuracy that any algorithm can achieve. There are two types of (deterministic) breakdown epsilons, namely the *strong* and the *weak* one, as well as their probabilistic versions. We define and discuss each of them in detail below. We begin with the strong breakdown epsilon.

Definition 9.17 (Strong breakdown epsilon). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, we define its *strong breakdown epsilon* as follows:

$$\epsilon_{\mathrm{B}}^{\mathrm{s}} = \sup\{\epsilon \geq 0 \,|\, \forall \text{ general algorithms } \Gamma, \, \exists \iota \in \Omega \text{ such that } \mathrm{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) > \epsilon\}.$$

Hence, the strong breakdown epsilon is the largest number $\epsilon \geq 0$ such that no algorithm can provide accuracy exceeding ϵ.

Remark 9.18 (The breakdown epsilons with respect to a specific computational model). The purpose of the strong and other breakdown epsilons is to establish universal impossibility results in the form of lower bounds on the achievable accuracy, and hence we define them in terms of general algorithms (and randomised general algorithms to be introduced below). However, occasionally, it may be convenient to work with the concept of a breakdown epsilon specific to the computational model. This can easily be done by replacing the words 'general algorithm' in Definition 9.17 by either 'Turing machine' or 'BSS machine', for example. In this case we will use the superscript 'A' (for arithmetic):

$$\epsilon_{\mathrm{B}}^{\mathrm{s,A}} \text{ is the strong breakdown epsilon in the Turing (or BSS) model},$$

where, as in Remark 9.9, the Turing model is understood if the inputs are rationals, and the BSS model otherwise.

The weak breakdown epsilon, introduced next, is the largest $\epsilon \geq 0$ such that all algorithms will need to use an arbitrarily large amount of input information to reach ϵ accuracy. To make this precise we first need to define the *minimum amount of input information*, which, in turn, we define in terms of an enumeration of the elements in Λ. We thus assume that Λ is countable and enumerated according to

$$\Lambda = \{f_k \,|\, k \in \mathbb{N}, \, k \leq |\Lambda|\}, \tag{9.10}$$

where $|\Lambda|$ denotes the cardinality of Λ. As we will see below, although we assume that Λ has a specific enumeration, many of the key concepts defined below, including the weak breakdown epsilon, are independent of the specific enumeration, and so the only assumption needed is that Λ be countable.

Definition 9.19 (Minimum amount of input information). Given the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \,|\, k \in \mathbb{N}, \, k \leq |\Lambda|\}$ and a general algorithm Γ, we define the *minimum amount of input information* $T_{\Gamma}(\iota)$ for Γ and $\iota \in \Omega$ as

$$T_{\Gamma}(\iota) := \sup\{m \in \mathbb{N} \,|\, f_m \in \Lambda_{\Gamma}(\iota)\}.$$

Note that, for ι such that $\Gamma(\iota) = \mathrm{NH}$, the set $\Lambda_{\Gamma}(\iota)$ may be infinite (see Definition 9.3), in which case $T_{\Gamma}(\iota) = \infty$.

We are now ready to define the weak breakdown epsilon.

Definition 9.20 (Weak breakdown epsilon). Given the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \mid k \in \mathbb{N}, \, k \leq |\Lambda|\}$, we define the *weak breakdown epsilon* by

$$\epsilon_{\mathrm{B}}^{\mathrm{w}} = \sup\{\epsilon \geq 0 \mid \forall \text{ general algorithms } \Gamma \text{ and } M \in \mathbb{N} \, \exists \iota \in \Omega \text{ such that}$$

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) > \epsilon \text{ or } T_\Gamma(\iota) > M\}.$$

As mentioned above, it is easy to see that the *weak breakdown epsilon* is independent of the ordering of Λ. In words, if the breakdown epsilons are greater than ϵ, we have the following:

  (i) (Strong Breakdown Epsilon) For any algorithm there is an input such that the algorithm fails to produce an ϵ-accurate solution on that input.
 (ii) (Weak Breakdown Epsilon) One can choose an arbitrary large integer M and find an input such that the algorithm will need to have used at least the M-th input and still not have reached ϵ accuracy. In other words, the amount of information needed is unbounded to reach ϵ-accuracy.

Remark 9.21 (Independence of ordering). Although the minimum amount of input information is dependent of the enumeration of Λ in (9.10), it is easy to see that the weak breakdown epsilon and the probabilistic weak breakdown epsilon (to be defined below) are independent of the choice of the enumeration.

9.4.1. *The weak breakdown epsilon for problems with Δ_1-information.* In the case of computational problems with Δ_1-information, the minimum amount of input information is related to the 'accuracy needed' on the input to the algorithm.

Definition 9.22 (Number of correct 'digits' on the input). Suppose that $\hat{\Lambda} = \{f_{k,m} \mid (k, m) \in \beta \times \mathbb{N}\}$ provides Δ_1-information for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$. Given a general algorithm Γ for the problem $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$, we define the *'number of digits' required on the input* according to

$$D_\Gamma(\iota) = \sup\{m \in \mathbb{N} \mid \exists k \in \beta \text{ s.t. } f_{k,m} \in \hat{\Lambda}_\Gamma(\iota)\}.$$

Remark 9.23 (Interpretation of T_Γ and D_Γ: accuracy on input and lower bound on runtime). Consider the computational problem $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ above. Then, independently of the enumeration of $\hat{\Lambda}$ used to define T_Γ, we have that

$$D_\Gamma(\iota_n) \to \infty \quad \text{as } n \to \infty \qquad \Longrightarrow \qquad T_\Gamma(\iota_n) \to \infty \quad \text{as } n \to \infty, \tag{9.11}$$

for every general algorithm Γ and every sequence $\{\iota_n\}_{n=0}^\infty$ in Ω. Moreover, in the case when Λ is finite, the implication (9.11) also holds in reverse. Therefore, the weak breakdown epsilon for problems with Δ_1-information derived from a finite set of evaluations is equivalently given by

$$\epsilon_{\mathrm{B}}^{\mathrm{w}} = \sup\{\epsilon \geq 0 \mid \forall \text{ general algorithms } \Gamma \text{ and } M \in \mathbb{N} \, \exists \iota \in \Omega \text{ such that}$$

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) > \epsilon \text{ or } D_\Gamma(\iota) > M\}.$$

Note that any reasonable complexity model would have a definition of runtime of an algorithm Γ such that the runtime is at least the number of digits it acquires from the input. In particular, any reasonable definition of the runtime of Γ (such as the standard definition to be specified in §9.8) will have

$$\mathbf{Runtime}_\Gamma(\iota) \geq D_\Gamma(\iota). \tag{9.12}$$

Hence, for such computational problems, D_Γ can be used to show that if the weak breakdown epsilon is greater than ϵ then any algorithm will have arbitrarily large runtime when attempting to achieve ϵ-accuracy.

9.5. **Randomised algorithms.** In many contemporary fields of mathematics of information such as deep learning, the use of randomised algorithms is widespread. We therefore need to extend the concept of a general algorithm to a *randomised random algorithm.*

Definition 9.24 (Randomised General Algorithm). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \,|\, k \in \mathbb{N}, \, k \leq |\Lambda|\}$, a *randomised general algorithm* (RGA) is a collection X of general algorithms $\Gamma : \Omega \to \mathcal{M} \cup \{\mathrm{NH}\}$, 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:

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

(Pii) For each $n \in \mathbb{N}$ and $\iota \in \Omega$, we have $\{\Gamma \in X \,|\, T_\Gamma(\iota) \leq n\} \in \mathcal{F}$.

(Piii) For all $\iota_1, \iota_2 \in \Omega$ and $E \in \mathcal{F}$ so that, for every $\Gamma \in E$ and every $f \in \Lambda_\Gamma(\iota_1)$, we have $f(\iota_1) = f(\iota_2)$, it holds that $\mathbb{P}_{\iota_1}(E) = \mathbb{P}_{\iota_2}(E)$.

It is not immediately clear whether condition (Pii) for a given RGA $(X, \mathcal{F}, \{\mathbb{P}_\iota\}_{\iota \in \Omega})$ holds independently of the choice of the enumeration of Λ. This is indeed the case and will be established in Lemma 10.2 further below.

Remark 9.25 (Assumption (Pii)). Note that (Pii) in Definition 9.24 is needed in order to ensure that the minimum amount of input information also becomes a valid random variable. More specifically, for each $\iota \in \Omega$, we define the random variable

$$T_{\Gamma^{\mathrm{ran}}}(\iota) : X \to \mathbb{N} \cup \{\infty\} \text{ according to } \Gamma \mapsto T_\Gamma(\iota).$$

As the minimum amount of input information is typically related to the minimum runtime, one would be dealing with a rather exotic probabilistic model if $T_{\Gamma^{\mathrm{ran}}}(\iota)$ were not a random variable. Indeed, note that the standard models of randomised algorithms (see [56]) can be considered as RGAs (in particular, satisfying (Pii)).

Remark 9.26 (The purpose of a randomised general algorithm: universal lower bounds). As for a general algorithm, the purpose of a randomised general algorithm is to have a definition that will encompass every model of computation, which will allow lower bounds and impossibility results to be universal. Indeed, randomised Turing and BSS machines can be viewed as randomised general algorithms.

We will, with a slight abuse of notation, also write RGA for the family of all randomised general algorithms for a given a computational problem and refer to the algorithms in RGA by Γ^{ran}. With the definitions above we can now make probabilistic version of the strong breakdown epsilon as follows.

Definition 9.27 (Probabilistic strong breakdown epsilon). Given a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \,|\, k \in \mathbb{N}, \, k \leq |\Lambda|\}$, we define the *probabilistic strong breakdown epsilon* $\epsilon_{\mathbb{PB}}^{\mathrm{s}} : [0, 1) \to \mathbb{R}$ according to

$$\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) = \sup\{\epsilon \geq 0, \,|\, \forall \Gamma^{\mathrm{ran}} \in \mathrm{RGA} \; \exists \iota \in \Omega \text{ such that } \mathbb{P}_\iota(\mathrm{dist}_{\mathcal{M}}(\Gamma_\iota^{\mathrm{ran}}, \Xi(\iota)) > \epsilon) > \mathrm{p}\},$$

where $\Gamma_\iota^{\mathrm{ran}}$ is defined in (Pi) in Definition 9.24.

As already mentioned in previous sections, impossibility results for randomised algorithms can differ if one considers only those algorithms that halt on every input, leading to the following two definitions.

Definition 9.28 (Halting randomised general algorithms). A randomised general algorithm Γ^{ran} for a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ is called a *halting randomised general algorithm* (hRGA) if $\mathbb{P}_\iota(\Gamma_\iota^{\mathrm{ran}} = \mathrm{NH}) = 0$, for all $\iota \in \Omega$.

We denote the class of all halting randomised general algorithms by hRGA.

Definition 9.29 (Probabilistic strong halting breakdown epsilon). Given the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \mid k \in \mathbb{N}, \, k \leq |\Lambda|\}$, we define the *halting probabilistic strong Breakdown-epsilon* $\epsilon^{\mathrm{s}}_{\mathbb{P}h\mathrm{B}} : [0, 1) \to \mathbb{R}$ according to

$$\epsilon^{\mathrm{s}}_{\mathbb{P}h\mathrm{B}}(\mathrm{p}) = \sup\{\epsilon \geq 0, \, | \, \forall \Gamma^{\mathrm{ran}} \in \mathrm{hRGA} \; \exists \iota \in \Omega \text{ such that } \mathbb{P}_\iota(\mathrm{dist}_{\mathcal{M}}(\Gamma^{\mathrm{ran}}_\iota, \Xi(\iota)) > \epsilon) > \mathrm{p}\},$$

where $\Gamma^{\mathrm{ran}}_\iota$ is defined in (Pi) in Definition 9.24.

Note that the probabilistic strong Breakdown-epsilon is not a single number but a function of p. Specifically, it is the largest ϵ so that the probability of failure with at least ϵ-error is greater than p. Similarly, there is a probabilistic version of the weak breakdown epsilon.

Definition 9.30 (Probabilistic weak breakdown epsilon). Given the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_k \mid k \in \mathbb{N}, \, k \leq |\Lambda|\}$, we define the *probabilistic weak breakdown epsilon* $\epsilon^{\mathrm{w}}_{\mathbb{P}\mathrm{B}} : [0, 1) \to \mathbb{R}$ according to

$$\epsilon^{\mathrm{w}}_{\mathbb{P}\mathrm{B}}(\mathrm{p}) = \sup\{\epsilon \geq 0 \, | \, \forall \Gamma^{\mathrm{ran}} \in \mathrm{RGA} \text{ and } M \in \mathbb{N} \; \exists \iota \in \Omega \text{ such that}$$
$$\mathbb{P}_\iota(\mathrm{dist}_{\mathcal{M}}(\Gamma^{\mathrm{ran}}_\iota, \Xi(\iota)) > \epsilon \text{ or } T_{\Gamma^{\mathrm{ran}}}(\iota) > M) > \mathrm{p}\},$$

where $\Gamma^{\mathrm{ran}}_\iota$ is defined in (Pi) in Definition 9.24.

As for the (deterministic) weak breakdown epsilon discussed above, it is easy to see that the probabilistic weak breakdown epsilon is independent of the enumeration of Λ. The probabilistic weak breakdown epsilon describes a weaker form of failure than the probabilistic strong breakdown epsilon. In particular, the weak breakdown epsilon of p is the largest ϵ so that for, any randomised algorithm and $M \in \mathbb{N}$, the probability of either getting an error of size at least ϵ or having spent runtime longer than M, is greater than p. The connection between the different breakdown epsilons will be summarised in Proposition 10.1 further below.

9.6. **Different types of impossibility results.** There are several non-equivalent statements about the non-existence of algorithms for a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ with Δ_1-information that we will discuss. For a fixed $\varepsilon > 0$, the statements are as follows:

(i) $\exists \hat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that, for the computational problem $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$, we have $\epsilon^{\mathrm{s}}_{\mathrm{B}} \geq \varepsilon$.
(ii) When considering the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$, we have $\epsilon^{\mathrm{s}}_{\mathrm{B}} \geq \varepsilon$.
(iii) $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} \notin \Delta_1^G$.

Note that (i) $\Rightarrow$ (ii) $\Rightarrow$ (iii). Observe also that (i) says that there is a particular choice $\hat{\Lambda}$ of collections of approximations that provide Δ_1-information for Λ such that no algorithm, given this specific Δ_1-information, will be able to secure ε accuracy. Note that the strong breakdown epsilon in (i) and (ii) could be replaced with any of the other breakdown epsilons (at a fixed p, in the case of the probabilistic breakdown epsilons), and the implication (i) $\Rightarrow$ (ii) would still hold. However, the implication (ii) $\Rightarrow$ (iii) holds only for the various strong breakdown epsilons.

Remark 9.31 (Turing's classical definition of non-computability). Note that in the SCI hierarchy framework, the classical Turing definition of non-computability [44], [55] is equivalent to $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} \notin \Delta_1^A$. However, as the main results in this paper demonstrate, in order to characterise the many facets of scientific computing one needs the richness of the SCI hierarchy framework.

9.7. **Theorem 3.4 – Part (i) and (ii) in the SCI language.** Let Ξ denote the solution map (as in (1.7)) to any of the problems (1.1) - (1.4). The statements in Theorem 3.4 are well-defined up to the definition of (random) algorithms and their runtime. Now that we have defined these concepts and the breakdown

epsilons, we state (i) and (ii) in Theorem 3.4 in the precise SCI language. For $m, N \in \mathbb{N}$ such that $m \leq N$ and a class of inputs Ω for any of the problems (1.1) - (1.4), define

$$\Omega_{m,N} = \{(y, A) \in \Omega \mid (y, A) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}\}, \quad \mathcal{M}_N = \mathbb{R}^N. \tag{9.13}$$

In the case of linear programming, we set $c = (1, \ldots, 1) \in \mathbb{R}^N$ for each fixed dimension N. We then have the following.

Proposition 9.32. *Let Ξ denote the solution map to any of the problems* (1.1) - (1.4) *with the regularisation parameters satisfying $\delta \in [0, 1]$, $\lambda \in (0, 1/3]$, and $\tau \in [1/2, 2]$ (and additionally being rational in the Turing case) and let the metric on $\mathcal{M}_N$ be induced by the $\| \cdot \|_p$-norm, for an arbitrary $p \in [1, \infty]$. Let $K \geq 1$ be an integer. There exist a set of inputs*

$$\Omega = \bigcup_{\substack{m, N \in \mathbb{N} \\ 4 \leq m < N}} \Omega_{m,N}, \text{ so that } \Xi : \Omega_{m,N} \rightrightarrows \mathcal{M}_N, \tag{9.14}$$

where $\Omega_{m,N}$ and $\mathcal{M}_N$ are defined as in (9.13)*, and $\mathcal{M}_N$ is equipped with the $\| \cdot \|_p$ norm for some $p \in [1, \infty]$, as well as sets of Δ_1-information $\hat{\Lambda}_{m,N} \in \mathcal{L}^1(\Lambda_{m,N})$ such that we have the following.*

  (i) *For the computational problem $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \hat{\Lambda}_{m,N}\}$ with any $m, N \in \mathbb{N}$, $4 \leq m < N$, we have $\epsilon^s_{\mathbb{P}hB}(\mathrm{p}) > 10^{-K}$, for all $\mathrm{p} > \frac{1}{2}$. Hence $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}^{\Delta_1} \notin \Delta_1^G$.*

  (ii) *For the same $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \hat{\Lambda}_{m,N}\}$ as in (i) we have $\epsilon^s_{\mathbb{P}B}(\mathrm{p}) > 10^{-K}$, for all $\mathrm{p} > \frac{2}{3}$. However, when considering the problems $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}_{m,N}, \mathcal{M}_N, \tilde{\Lambda}_{m,N}\}$, there exists a randomised algorithm Γ^{ran} (a randomised Turing machine or a randomised BSS machine), with a non-zero probability of not halting, that takes the dimensions m, N and any $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ as input (see Remark 9.16) and satisfies*

$$\mathbb{P}_{\tilde{\iota}}\big(\mathrm{dist}_{\mathcal{M}}(\Gamma^{\mathrm{ran}}(m, N, \tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}\big) \geq \frac{2}{3}.$$

The statements above are true even when we require the inputs in $\Omega_{m,N}$ to be well-conditioned for all m, N and bounded from above and below. In particular, for any input $\iota = (y, A) \in \Omega_{m,N}$ we have $\mathrm{Cond}(AA^*) \leq 3.2$, $C_{\mathrm{FP}}(\iota) \leq 4$, $\mathrm{Cond}(\Xi) \leq 179$, $\|y\|_\infty \leq 2$, and $\|A\|_{\max} = 1$.

9.8. **Runtime of algorithms.** In order to provide exact statements of the rest of Theorem 3.4 we need the precise definition of the runtime of an algorithm, for which one has to specify a computational model. Recall that we want to prove results about algorithms for problems $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$ with Δ_1-information, as such results are stronger than, say, the corresponding results for $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \hat{\Lambda}\}$ with a particular choice of $\hat{\Lambda} \in \mathcal{L}^1(\Lambda)$. We have already discussed specific computational models in Definition 9.8 and Remark 9.9, however, in order to make precise statements about complexity we need to specify such models more closely. Concretely, we will consider the following models.

  (i) (**Turing model**). We follow the classical setup in [55], wherein the components of the inputs $\tilde{\iota} = \{f_{j,n}(\iota)\}_{j,n \in \beta \times \mathbb{N}} \in \tilde{\Omega}$ are provided by an oracle tape that, at various times during the computation, contains a binary string representing one of the dyadic numbers $f_{j,n}(\iota)$. Specifically, the Turing machine can make a query to the *input oracle* $\mathscr{O}$, which then writes $f_{j,n}(\iota)$ onto the oracle tape at cost $j + n$, always choosing the unique finite binary string representing $f_{j,n}(\iota)$. Now, given a Turing machine Γ for the problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$, we define its runtime on $\tilde{\iota}$ as follows:

Runtime$_\Gamma(\tilde{\iota}) = $ the number of steps performed by the Turing machine Γ before halting

$\qquad\qquad$ + the sum of costs of all the calls to the oracle for $\tilde{\iota}$ (with the cost as above).

  Note that this model accounts implicitly for the cost of all the calls to the oracle, as the output of the oracle is printed on the work tape, and hence read by the machine.

(ii) (**Arithmetic Turing model**). In the case where one only wishes to count the number of arith-
metic operations and comparisons that an algorithm performs, there is no canonical account of
the cost of calling the oracle, as opposed to the classical Turing model above. Hence, in this
model (as discussed in §2.1.1 and (see for example Lovász [41, p. 36])), it is natural to adopt
the convention that the cost of calling the oracle $\mathscr{O}$ (as specified in item (i) above) is given by
$\mathrm{pol}(n)$, for all $j \in \beta$, $n \in \mathbb{N}$, and $\iota \in \Omega$, for some non-zero polynomial pol fixed beforehand.
Thus, we define the arithmetic runtime of an algorithm Γ in the Turing model on $\tilde{\iota}$ as follows:

$\mathbf{Runtime}_\Gamma(\tilde{\iota}) =$ the number of arithmetic operations and comparisons

performed by Γ before halting

\+ the sum of costs of all the calls to the oracle for $\tilde{\iota}$ (with the cost as above).

(iii) (**The BSS model**). In this model, the BSS machine is equipped with an extra oracle node $\mathscr{O}$ for
$\tilde{\iota} = \{f_{j,n}(\iota)\}_{j,n \in \beta \times \mathbb{N}} \in \tilde{\Omega}$ that outputs $f_{j,n}$ on input (j, n) (see [42] for details). As discussed
in §2.1.1 and in item (ii) above, it is customary to assume that the cost of calling the oracle
is polynomial in n, as in the arithmetic Turing model. Thus, given a BSS machine Γ for the
problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$, we define its runtime on $\tilde{\iota}$ as follows:

$\mathbf{Runtime}_\Gamma(\tilde{\iota}) =$ the number of arithmetic operations and comparisons

performed by the BSS machine Γ before halting

\+ the sum of costs of all the calls to the oracle for $\tilde{\iota}$ (with the cost as above).

9.9. **Theorem 3.4 – Part (iii) - (v) in the SCI language.** We follow the setup in §9.7. Note that
with the definitions of runtime in §9.8 all satisfy (9.12), and hence we have that, for a given prob-
lem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$ with Λ finite and a probabilistic weak breakdown epsilon $\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) > r$, for some
$\mathrm{p}, r > 0$, any randomised algorithm will require an arbitrarily long runtime in order to achieve accuracy
of at least r with probability at least p.

We are now ready to formalise the rest of Theorem 3.4.

Proposition 9.33. *There exist Ω and $\hat{\Lambda}_{m,N} \in \mathcal{L}^1(\Lambda_{m,N})$ as described in* (9.14) *for which the conclusion
of Proposition 9.32 holds with $K \geq 2$, and we additionally have the following.*

(iii) *For the problem $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \hat{\Lambda}_{m,N}\}$ with any $m, N \in \mathbb{N}$, $4 \leq m < N$, we have that
$\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) > 10^{-(K-1)}$, for all $\mathrm{p} > \frac{1}{2}$. However, when considering the problems $\{\Xi, \Omega_{m,N}, \mathcal{M}_N,
\Lambda_{m,N}\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}_{m,N}, \mathcal{M}_N, \tilde{\Lambda}_{m,N}\}$, there exists an algorithm Γ (a Turing machine or a BSS
machine) that takes the dimensions m, N and any $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ as input (see Remark 9.16) and
satisfies*

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(m, N, \tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \leq 10^{-(K-1)}.$$

(iv) *There exists a polynomial $\mathrm{pol} : \mathbb{R} \to \mathbb{R}$ and an algorithm Γ (a Turing machine or a BSS machine)
that takes the dimensions m, N and any $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ as input (see Remark 9.16) and satisfies*

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(m, N, \tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \leq 10^{-(K-2)}.$$

*Furthermore, in the Turing case, the arithmetic runtime and the space complexity of the Turing
machine are bounded by $\mathrm{pol}(n_{\mathrm{var}})$, where $n_{\mathrm{var}} = mN + m$ is the number of variables, and
the number of digits read from the oracle tape is bounded by $\mathrm{pol}(\log(n_{\mathrm{var}}))$. In particular, the
runtime in the Turing model is polynomial in n_{var}. In the BSS case, the runtime is likewise
bounded by $\mathrm{pol}(n_{\mathrm{var}})$.*

In the proof of Proposition 9.33 we will additionally need the following concepts of the size of a
dyadic rational and the encoding length of rational vectors.

Definition 9.34 (Size of dyadic rationals). For a dyadic rational d given by its binary representation $d = \pm s_n s_{n-1} \cdots s_0 . t_1 t_2 \cdots t_m$, we call m its precision, and we define its bit size as $m + n + 3$. The bit size of a dyadic rational vector or matrix is defined as the sum of the bit sizes of its entries. We define the bit size of a vector of dyadic rationals as the sum of bit sizes of its entries.

Definition 9.35 (Encoding length of rational numbers). We define the encoding length of an integer $n \in \mathbb{Z}$ as $\mathrm{Len}(n) = 1 + \lceil \log_2(|n| + 1) \rceil$. For a rational $p/q \in \mathbb{Q}$, where p and $q > 0$ are coprime integers, we define the encoding length of p/q as $\mathrm{Len}(p/q) = \mathrm{Len}(p) + \mathrm{Len}(q)$. Similarly, if w is a vector of rationals, we define the encoding length $\mathrm{Len}(w)$ as the sum of the encoding lengths of the components of w.

9.10. **Theorem 7.1 in the SCI language.** We recall the setup for Theorem 7.1. Fix real constants $\rho \in (1/3, 1)$, $\tau > 10$, $b_1 > 3$, and $b_2 > 6$. For $s, m, N \in \mathbb{N}$ such that $m \leq N$ and $\epsilon \in [0, 1]$, define

$$\Omega_{s,m,N}^{\epsilon} = \{(y, A) \in \mathbb{R}^m \times \mathbb{R}^{m \times N} \mid (y, A) \text{ satisfies (9.16)}\}, \tag{9.15}$$

where

$$A \text{ satisfies the RNP (7.1) of order } s \text{ with parameters } \rho \text{ and } \tau,$$

$$\|y - Ax\|_2 \leq \epsilon \text{ for some } x \text{ that is } s\text{-sparse, and } \|y\|_2 \leq b_1, \ \|A\|_2 \leq b_2 \sqrt{N/m}. \tag{9.16}$$

Finally, define

$$\Omega^{\epsilon} = \bigcup_{\substack{s,m,N \in \mathbb{N} \\ m \leq N}} \Omega_{s,m,N}^{\epsilon}. \tag{9.17}$$

Proposition 9.36. *Let* Ξ_{BP} *and* Ξ_{UL} *denote the solution maps respectively to the* ℓ^1*-BP problem* (1.2) *and the* ℓ^1*-Lasso problem* (1.3)*, and consider the* $\|\cdot\|_2$*-norm for measuring the error. If* $\Omega_{s,m,N}^{\epsilon}$ *and* Ω^{ϵ} *are as in* (9.15) *and* (9.17) *the following holds.*

   (i) There exists a constant $C > 0$*, independent of* ρ, τ, b_1 *and* b_2*, such that if we fix* $s = 2^k$ *for some* $k \in \mathbb{N}$ *and any* m, N *such that* $N > m$ *with* $m \geq C s \log^2(2s) \log(N)$*, then we have the following. For an arbitrary* $\delta \in (0, 1]$ *consider the computational problem* $\{\Xi_{\mathrm{BP}}, \Omega_{s,m,N}^0, \mathcal{M}, \Lambda\}^{\Delta_1}$*, where* $\mathcal{M} = \mathbb{R}^N$*. Let*

$$K \geq \lceil \log_{10}(2\delta^{-1}) \rceil.$$

   Then, for any $\mathrm{p} > \frac{1}{2}$*, we have* $\epsilon_{\mathbb{P}h\mathrm{B}}^s(\mathrm{p}) > 10^{-K}$*. Hence* $\{\Xi_{\mathrm{BP}}, \Omega_{s,m,N}^0, \mathcal{M}, \Lambda\}^{\Delta_1} \notin \Delta_1^G$*.*

   (ii) Let s*,* m*, and* N *be as in (i), and for an arbitrary* $\lambda \in (0, 1/(2\sqrt{s})]$ *consider the computational problem* $\{\Xi_{\mathrm{UL}}, \Omega_{s,m,N}^0, \mathcal{M}, \Lambda\}^{\Delta_1}$ *and*

$$K \geq \lceil \log_{10}(2 s^{-1/2} \lambda^{-1}) \rceil.$$

   Then, for any $\mathrm{p} > \frac{1}{2}$*, we have* $\epsilon_{\mathbb{P}h\mathrm{B}}^s(\mathrm{p}) > 10^{-K}$*. Hence,* $\{\Xi_{\mathrm{UL}}, \Omega_{s,m,N}^{\epsilon}, \mathcal{M}, \Lambda\}^{\Delta_1} \notin \Delta_1^G$*.*

   (iii) There exists a polynomial $\mathrm{pol} : \mathbb{R}^2 \to \mathbb{R}$ *and, for every* $\delta \in [0, (1 - \rho)/(16\tau)]$*, there exists an algorithm* Γ_δ *(a Turing machine or a BSS machine) that takes any dimensions* m*,* N*, the accuracy parameter* K*, and any* $\tilde{\iota} \in \tilde{\Omega}_{s,m,N}^\delta$ *as input (as described in Remark 9.16) and satisfies*

$$\mathrm{dist}_{\mathcal{M}}(\Gamma_\delta(m, N, K, \tilde{\iota}), \tilde{\Xi}_{\mathrm{BP}}(\tilde{\iota})) \leq 10^{-K}$$

   for all $K \in \mathbb{N}$ *satisfying*

$$K \leq \lfloor \log_{10}((1 - \rho)(16\tau)^{-1}\delta^{-1}) \rfloor. \tag{9.18}$$

   In the case that $\delta = 0$*,* (9.18) *is to be interpreted as* $K < \infty$*. Moreover, the runtime of* Γ_δ *(steps performed by the Turing machine, arithmetic operations performed by the BSS machine) is bounded by* $\mathrm{pol}(n, K)$*, where* $n = m + mN$ *is the number of variables. In particular, for* $\delta = 0$*, the runtime is bounded by* $\mathrm{pol}(n, K)$ *for all* $K \in \mathbb{N}$*.*

(iv) *Consider the ℓ^1-BP problem (1.2) with $\delta = 0$. For any fixed $s \geq 3$, there are infinitely many pairs (m, N) and inputs $\iota = (Ax, A) \in \Omega^0$, where $x \in \mathbb{R}^m$ is s-sparse and $A \in \mathbb{R}^{m \times N}$ is a subsampled Hadamard, Bernoulli, or Hadamard-to-Haar matrix such that*

$$C_{\mathrm{RCC}}(\iota) = \infty.$$

In particular, appreciating (iii) and (iv), there exist inputs in Ω with infinite RCC condition number, yet the problem is in P.

Remark 9.37. The independence of the runtime on δ in part (iii) is a consequence of (9.18).

Remark 9.38. One can strengthen the result in part (iii) in the following way: it is possible to use Nemirovski's surface-following method (concretely, the result in Section 4.2 in [57]) to devise an algorithm that solves the basis pursuit problem with ℓ^1 regularisation up to precision $10^{-K} > 16\tau\delta/(1 - \rho)$ to obtain the explicit complexity bound $\mathcal{O}(1) \cdot N^{3.5} \cdot K \cdot \log\left(2 + \frac{N\tau}{1-\rho}\right)$ in the arithmetic model of computation (more precisely, the BSS model with an oracle for the square root). We choose not to do so in this paper and instead base our proof of part (iii) on the ellipsoid algorithm to obtain simple polynomial time complexity bounds in both the BSS and Turing models of computation.

9.11. **Computing the exit flag – Theorem 5.1 in the SCI language.** Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$ be a computational problem and recall the definition of $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ from Definition 9.14. Suppose that we are interested in finding an algorithm Γ that, for every $\tilde{\iota} \in \tilde{\Omega}$, computes an approximation to $\tilde{\Xi}(\tilde{\iota})$ with κ accuracy (i.e. so that $\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \Xi(\tilde{\iota})) \leq \kappa$). If the breakdown epsilon $\epsilon_{\mathrm{B}}^{\mathrm{s}}$ is greater than κ then this task is impossible – there will be at least one input for which the algorithm fails to get κ accuracy. However, that does not mean that a given algorithm will fail on *all* possible inputs. There may be a set of inputs for which the algorithm succeeds in producing an output that is κ away from a true solution. This leads to the following exit flag question:

Can we design an algorithm that identifies whether or not Γ fails on a given input (meaning that Γ fails to produce κ accuracy on its input)?

Phrased more precisely in the language of computational problems, given a recursive (either in the Turing or the BSS model) algorithm Γ for $\{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$, we define

$$\tilde{\Omega}_\Gamma := \{\tilde{\iota} \in \tilde{\Omega} \,|\, \mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \leq \kappa\} \subset \tilde{\Omega}, \tag{9.19}$$

then $\epsilon_{\mathrm{B}}^{\mathrm{s}} > \kappa$ implies that $\tilde{\Omega}_\Gamma$ is a strict subset of $\tilde{\Omega}$. Now, defining

$$\Xi^E : \tilde{\Omega} \ni \tilde{\iota} \mapsto \begin{cases} 1 & \text{if } \tilde{\iota} \in \tilde{\Omega}_\Gamma \\ 0 & \text{if } \tilde{\iota} \in \tilde{\Omega} \setminus \tilde{\Omega}_\Gamma, \end{cases} \tag{9.20}$$

the exit flag problem is to design an algorithm Γ^E to solve the computational problem

$$\{\Xi^E, \Omega, \{0, 1\}, \Lambda\}^{\Delta_1} = \{\Xi^E, \tilde{\Omega}, \{0, 1\}, \tilde{\Lambda}\}, \tag{9.21}$$

where the metric on the space $\{0, 1\}$ is inherited from $\mathbb{R}$.

Remark 9.39 (Key assumption). Of course, this problem is trivial if the algorithm produces outputs that are far away from the set $\Xi(\Omega)$. This is a somewhat contrived scenario as such algorithms would not be sensible candidates for attempting to solve the problems defined in equations (1.1)-(1.6). We thus need to make a technical assumption on the type of algorithms we will analyse for the exit flag problem. Concretely, we fix an $\alpha < \kappa$ and assume that our algorithm Γ defined on Ω satisfies

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\Omega})) < \alpha \text{ for all } \tilde{\iota} \in \tilde{\Omega}. \tag{9.22}$$

Given the computational problem $\{\Xi^E, \tilde{\Omega}, \{0, 1\}, \tilde{\Lambda}\}$, one may ask where it lies in the SCI hierarchy. We will provide results on this, however, in some circumstances we will actually prove stronger results by considering the problem of computing the exit flag with additional access to an oracle that provides an exact solution to the problem that Γ is trying to solve. In particular, one can ask:

> *Given an oracle for the true solution, can we design an algorithm that identifies whether or not Γ fails on a given input (meaning that Γ fails to produce K correct digits when given the input)?*

Concretely, we will assume that we are allowed to design the exit flag algorithm Γ^E so that Γ^E has access to both and element ρ of $\tilde{\Xi}(\tilde{\iota})$ as well as the input $\tilde{\iota}$. We will consider Γ^E to be successful only if it correctly computes the exit flag for every such ρ. Indeed, it is reasonable to preclude the exit algorithm from being able to access a "convenient" element of $\tilde{\Xi}(\tilde{\iota})$, as this would make it too powerful for any nontrivial statements to be made.

9.11.1. *Formalising the oracle computational problem.* In order to make the concept of an oracle for the true solution precise we need to formalise the definition of an oracle computational problem. Consider two computational problems $\{\Xi_1, \Omega, \mathcal{M}_1, \Lambda_1\}$ and $\{\Xi_2, \Omega, \mathcal{M}_2, \Lambda_2\}$ with the same input set Ω, and assume that $\mathcal{M}_2 \subset \mathbb{C}^M$. Given $\omega \geq 0$, suppose that functions $\{g_k\}_{k=1}^M$, $g_k : \Omega_1 \to \mathbb{D}$, are such that

$$\{g_k(\iota)\}_{k=1}^M \in \mathcal{B}_\omega^\infty(\Xi_2(\iota)). \tag{9.23}$$

Then $\{g_k(\iota)\}_{k=1}^M$ is an ω-accurate approximation of an element of $\Xi_2(\iota)$. We can thus define

$$\mathcal{L}^{\mathcal{O}, \omega, \Xi_2}(\Lambda_1) = \{\Lambda_1 \cup \{g_k\}_{k=1}^M \mid g_k : \Omega \to \mathbb{D}, k \in \{1, \ldots, M\}, \text{ satisfy (9.23)}\} \tag{9.24}$$

Note that it is crucial to allow for the approximation parameter ω since a Turing machine would be unable to access the true solution if it has irrational entries.

Now, for a fixed $\Lambda_1^+ \in \mathcal{L}^{\mathcal{O}, \omega, \Xi_2}(\Lambda_1)$, the computational problem $\{\Xi_1, \Omega, \mathcal{M}_1, \Lambda_1^+\}$ would allow the algorithm to access information about Ξ_2. However, as discussed above, a sensible algorithm relying on "oracle information" ought to work with every choice of such an oracle. We thus define a new oracle computational problem $\{\Xi_1^{\mathcal{O}}, \Omega^{\mathcal{O}}, \mathcal{M}_1, \Lambda_1^{\mathcal{O}}\}$ by analogy to Definition 9.14 according to

$$\Omega^{\mathcal{O}} = \Omega^{\mathcal{O}}(\omega) = \left\{\iota^{\mathcal{O}} = \iota \oplus \{g_k(\iota)\}_{k=1}^M \mid \iota \in \Omega, \{g_k\}_{k=1}^M \text{ satisfy (9.23)}\right\}, \tag{9.25}$$

$\Xi_1^{\mathcal{O}}(\iota^{\mathcal{O}}) = \Xi_1(\iota)$ and $\Lambda_1^{\mathcal{O}} = \{f^{\mathcal{O}} \mid f \in \Lambda_1\} \cup \{g_k^{\mathcal{O}}\}_{k=1}^M$, where the $f^{\mathcal{O}}$ and $g_k^{\mathcal{O}}$ are defined as

$$f^{\mathcal{O}}(\iota^{\mathcal{O}}) = f(\iota), \quad g_k^{\mathcal{O}}(\iota^{\mathcal{O}}) = g_k(\iota).$$

The dependency on ω in the definition of $\Omega^{\mathcal{O}}(\omega)$ will usually be suppressed to lighten the notation. We can now make the following formal definition.

Definition 9.40 (Oracle computational problem)**.** Given two computational problems $\{\Xi_1, \Omega, \mathcal{M}_1, \Lambda_1\}$ and $\{\Xi_2, \Omega, \mathcal{M}_2, \Lambda_2\}$ as specified above, we say that

$$\{\Xi_1, \Omega, \mathcal{M}_1, \Lambda_1\}^{\mathcal{O}, \omega, \{\Xi_2, \Omega, \mathcal{M}_2, \Lambda_2\}} := \{\Xi_1^{\mathcal{O}}, \Omega^{\mathcal{O}}, \mathcal{M}_1, \Lambda_1^{\mathcal{O}}\}$$

where $\Omega^{\mathcal{O}}$, $\Xi_1^{\mathcal{O}}$, and $\Lambda_1^{\mathcal{O}}$ are as specified above, is the oracle computational problem with respect to $\{\Xi_2, \Omega, \mathcal{M}_2, \Lambda_2\}$. Whenever it is clear which computational problem the oracle problem is relative to, we will simply write $\{\Xi_1, \Omega, \mathcal{M}_1, \Lambda_1\}^{\mathcal{O}, \omega}$ for $\{\Xi_1, \Omega, \mathcal{M}_{,1} \Lambda_1\}^{\mathcal{O}, \omega, \{\Xi_2, \Omega, \mathcal{M}_2, \Lambda_2\}}$.

9.11.2. *Computing the exit flag with an oracle.* Returning to the exit flag problem, one could also ask the opposite question to what we asked above:

> *Given an oracle for the exit flag can we design an algorithm that produces ω accurate solution to the original problem?*

A positive answer to this question and a negative answer to the original question can be seen as evidence that the exit flag problem is strictly harder than the original problem, which is the topic of the following proposition formalising Theorem 5.1.

Proposition 9.41 (Impossibility of computing the exit flag)**.** *Let Ξ denote the solution map to any of the problems* (1.1) - (1.4) *with the regularisation parameters satisfying $\delta \in [0,1]$, $\lambda \in (0, 1/3]$, and $\tau \in [1/2, 2]$ (and additionally being rational in the Turing case) and let the metric on $\mathcal{M}$ be induced by the $\|\cdot\|_p$ norm. Let $K \in \mathbb{N}$ and fix α and ω so that $0 < \alpha \leq \omega < 10^{-K}$. Then, for any fixed dimensions $m < N$ with $m \geq 4$, there exists a class of inputs Ω such that, if Γ is a general algorithm for the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ satisfying* (9.22) *(where we write Λ for $\Lambda_{m,N}$ to lighten the notation) and $\{\Xi^E, \Omega, \{0,1\}, \Lambda\}^{\Delta_1} = \{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}$ is the exit flag computational problem defined in* (9.21)*, we then have the following.*

(i) *There exists a $\tilde{\Lambda}^+ \in \mathcal{L}^{\mathcal{O}, \omega, \tilde{\Xi}}(\tilde{\Lambda})$ such that, for $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}^+\}$ we have $\epsilon_{\mathbb{PB}}^{s}(\mathrm{p}) \geq 1/2$, for all $\mathrm{p} > \frac{1}{2}$.*

(ii) *The problem of computing the exit flag of Γ is strictly harder than computing a K digit approximation to Ξ in the following sense: For the oracle problem $\{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}^{\mathcal{O}, \omega}$ with respect to $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}$ we have $\epsilon_B^{s,A} \leq 10^{-K}$ (see Remark 9.18). However, when considering the oracle problem $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}^{\mathcal{O}, \omega}$ with respect to $\{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$, we have $\epsilon_B^s \geq 1/2$ (which follows from (i)).*

(iii) *If Ξ is the solution map to either linear programming or basis pursuit, there exists a class of inputs $\Omega^\sharp \neq \Omega$ such that, if Γ is a general algorithm for the computational problem $\{\Xi, \Omega^\sharp, \mathcal{M}, \Lambda\}^{\Delta_1}$ satisfying* (9.22) *with α, then there is a $\hat{\Lambda} \in \mathcal{L}^1(\Lambda)$ so that for $\{\Xi^E, \Omega^\sharp, \{0,1\}, \hat{\Lambda}\}$ we have $\epsilon_{\mathbb{PB}}^s(\mathrm{p}) \geq 1/2$, for all $\mathrm{p} > \frac{1}{2}$. However, if we consider the oracle problem $\{\Xi^E, \tilde{\Omega}^\sharp, \{0,1\}, \tilde{\Lambda}\}^{\mathcal{O}, \omega}$ with respect to $\{\Xi, \Omega^\sharp, \mathcal{M}, \Lambda\}^{\Delta_1}$, then $\{\Xi^E, \tilde{\Omega}^\sharp, \{0,1\}, \tilde{\Lambda}\}^{\mathcal{O}, \omega} \in \Delta_1^A$.*

The statements above are true even when we require the inputs in $\Omega_{m,N}$ to be well-conditioned for all m, N and bounded from above and below. In particular, for any input $\iota = (y, A) \in \Omega_{m,N}$ we have $\mathrm{Cond}(AA^*) \leq 3.2$, $C_{\mathrm{FP}}(\iota) \leq 4$, $\mathrm{Cond}(\Xi) \leq 179$, $\|y\|_\infty \leq 2$, and $\|A\|_{\max} = 1$.

9.12. The full SCI hierarchy and the feasibility part of the extended Smale's 9th problem.

In order to prove Theorem 6.1 we need the full SCI hierarchy. Building on the SCI hierarchy introduced in Definition 9.11, one may assume that the metric space $\mathcal{M}$, say $\mathcal{M} = \mathbb{R}$ or $\mathcal{M} = \{0, 1\}$ with the standard metric, is equipped with extra structure, say a total order, that allows one to define convergence from above or below of functions valued in $\mathcal{M}$. By analogy to using the metric $d_{\mathcal{M}}$ in the approximation property (9.6) of towers of algorithms, the extra structure on $\mathcal{M}$ allows us to define for a new type of approximation. As we argue below, this is important, for example, in computer-assisted proofs and scientific computing.

For simplicity of exposition, in this subsection we consider only computational problems whose solution map Ξ is single-valued.

Definition 9.42 (The SCI Hierarchy for a totally ordered set)**.** Given the setup in Definition 9.11 and additionally assuming that $\mathcal{M}$ is totally ordered, we define the following sets:

$$\Sigma_0^\alpha = \Pi_0^\alpha = \Delta_0^\alpha,$$

$$\Sigma_m^\alpha = \{\{\Xi, \Omega, \mathcal{M}, \Lambda\} \in \Delta_{m+1}^\alpha \mid \exists \text{ tower } \{\Gamma_{n_m,\dots,n_1}\} \text{ of height } m \text{ s.t. } \Gamma_{n_m}(\iota) \nearrow \Xi(\iota) \ \forall \iota \in \Omega\},$$

$$\Pi_m^\alpha = \{\{\Xi, \Omega, \mathcal{M}, \Lambda\} \in \Delta_{m+1}^\alpha \mid \exists \text{ tower } \{\Gamma_{n_m,\dots,n_1}\} \text{ of height } m \text{ s.t. } \Gamma_{n_m}(\iota) \searrow \Xi(\iota) \ \forall \iota \in \Omega\},$$

for $m \in \mathbb{N}$, where $\nearrow$ and $\searrow$ denotes convergence from below and above respectively, and the towers of algorithms are assumed to consist of halting algorithms.

In the special case when $\mathcal{M} = \{0, 1\}$, Definition 9.42 yields the full SCI hierarchy for arbitrary decision problems, where 1 is interpreted as *true* and 0 as *false*. One can then naturally consider a computational problem with Δ_1-information $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1}$ and ask in which of the sets Δ_k^α, Σ_k^α, or Π_k^α it lies.

9.13. **Theorem 6.1 in the SCI language.** Recall that, for $k \in \mathbb{N} \cup \{\infty\}$ and $M \in \mathbb{R}$, our decision problem of interest is to decide whether there is an $x \in \mathbb{R}^N$ such that

$$\langle x, c \rangle_k \leq M \text{ subject to } Ax = y, \quad x \geq 0 \text{ coordinatewise}, \tag{9.26}$$

where $\langle x, c \rangle_k := \lfloor 10^k \langle x, c \rangle \rfloor 10^{-k}$ with c set to $(1, \dots, 1) \in \mathbb{R}^N$ for each fixed N as in §9.7, corresponding to a computational problem with solution map

$$\Xi_k : \Omega \to \{0, 1\}, \quad \Xi_k(\iota) = \begin{cases} 1, & \exists x \in \mathbb{R}^N \text{ satisfying (9.26)} \\ 0, & \text{else} \end{cases}, \tag{9.27}$$

and set Ω of inputs of the form $\iota = (y, A) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}$. Now, for $k \in \mathbb{N}$, define the set

$$\mathcal{S}_k^9 := \bigcup_{n \in \mathbb{N} \cup \{0\}} \left[(n + 0.9) \cdot 10^{-(k-1)}, (n + 1) \cdot 10^{-(k-1)} \right), \tag{9.28}$$

in other words, $\mathcal{S}_k^9$ is the set of positive real numbers whose k-th digit after the decimal point is 9. Note that, for all $k \in \mathbb{N}$, we have

$$M \in \mathcal{S}_k^9 \implies \{x \in \mathbb{R}^N \mid \langle x, c \rangle_k \leq M\} = \{x \in \mathbb{R}^N \mid \langle x, c \rangle_{k-1} \leq M\} \implies \Xi_k = \Xi_{k-1},$$

and therefore the computational problem (9.27) is indistinguishable for $k = K$ and $k = K - 1$ whenever $M \in \mathcal{S}_K^9$. When this is not the case, the following proposition shows that the problem for $k = K$ and $k = K - 1$ can have fundamentally different computational properties on the same set of inputs.

Proposition 9.43. *Consider a real $M \geq 0$ and an integer $K \geq 2$. There exist a set of inputs*

$$\Omega = \bigcup_{m < N} \Omega_{m,N} \text{ so that } \quad \Xi_k : \Omega_{m,N} \to \{0, 1\},$$

where $\Omega_{m,N}$ is defined as in (9.13) and sets of Δ_1-information $\hat{\Lambda}_{m,N} \in \mathcal{L}^1(\Lambda_{m,N})$ such that

(i) $\{\Xi_K, \Omega_{m,N}, \{0,1\}, \hat{\Lambda}_{m,N}\} \notin \Sigma_1^G$, for all $m, N \in \mathbb{N}$ with $m < N$, and

(ii) for $\{\Xi_K, \Omega_{m,N}, \{0,1\}, \hat{\Lambda}_{m,N}\}$ with any $m, N \in \mathbb{N}$ such that $m < N$, we have $\epsilon_{\mathbb{P}B}^s(p) \geq \frac{1}{2}$, for all $p > \frac{1}{2}$.

Additionally, depending on whether M is in one of $\mathcal{S}_K^9$ or $\mathcal{S}_{K-1}^9$, one or both of the following hold as well:

(iii) For $\{\Xi_{K-1}, \Omega_{m,N}, \{0,1\}, \hat{\Lambda}_{m,N}\}$ with any $m, N \in \mathbb{N}$ such that $m < N$, we have $\epsilon_{\mathbb{P}B}^w(p) \geq \frac{1}{2}$, for all $p > \frac{1}{2}$. However, when considering the problems $\{\Xi_{K-1}, \Omega_{m,N}, \{0,1\}, \Lambda_{m,N}\}^{\Delta_1} =$

$\{\tilde{\bar{\Xi}}_{K-1}, \tilde{\Omega}_{m,N}, \{0,1\}, \tilde{\Lambda}_{m,N}\}$, *there exists an algorithm* Γ *(a Turing machine or a BSS machine) that takes the dimensions* m, N *and any* $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ *as input and satisfies*

$$\Gamma(m, N, \tilde{\iota}) = \tilde{\bar{\Xi}}_{K-1}(\tilde{\iota}).$$

(iv) Considering the problems $\{\Xi_{K-2}, \Omega_{m,N}, \{0,1\}, \Lambda_{m,N}\}^{\Delta_1} = \{\tilde{\bar{\Xi}}_{K-2}, \tilde{\Omega}_{m,N}, \{0,1\}, \tilde{\Lambda}_{m,N}\}$, *there exists a polynomial* $\mathrm{pol} : \mathbb{R} \to \mathbb{R}$ *and an algorithm* Γ *(a Turing machine or a BSS machine) that takes the dimensions* m, N *and any* $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ *as input and satisfies*

$$\Gamma(m, N, \tilde{\iota}) = \tilde{\bar{\Xi}}_{K-2}(\tilde{\iota}).$$

Furthermore, in the Turing case, the arithmetic runtime and the space complexity of the Turing machine are bounded by $\mathrm{pol}(n_{\mathrm{var}})$, *where* $n_{\mathrm{var}} = mN + m$ *is the number of variables, and the number of digits read from the oracle tape is bounded by* $\mathrm{pol}(\log(n_{\mathrm{var}}))$. *In particular, the runtime in the Turing model is polynomial in* n_{var}. *In the BSS case, the runtime is likewise bounded by* $\mathrm{pol}(n_{\mathrm{var}})$.

Concretely, (iii) holds whenever $M \notin \mathcal{S}_K^9$, *whereas (iv) holds whenever* $M \notin \mathcal{S}_{K-1}^9$.

9.14. **Theorem 8.3 in the SCI language.** Recall the setup in §8.2 as well as the definition of $\Lambda_{d,r,s}$ in (9.7). The formal statement of Theorem 8.3 is now as follows.

Proposition 9.44 (Non-existence of algorithms for deep learning). *There is an uncountable family of classification functions* $f : \mathbb{R}^d \to \{0,1\}$ *with* $d \geq 2$ *such that, for any neural network dimensions* $\mathbf{N} = (N_L = 1, N_{L-1}, \ldots, N_1, N_0 = d)$ *with* $L \geq 2$ *and any* $r \geq 3(N_1+1)\cdots(N_{L-1}+1)$, *there exist uncountably many training sets* $\mathcal{T} = \{x^1, \ldots, x^r\} \in \mathcal{S}_{\varepsilon(r)}^f$ *of size* r *(where* $\varepsilon(n) := [(4n+3)(4n+4)]^{-1}$*) such that we have the following.*

Let $\mathcal{R} \in \mathcal{CF}_r$ *be a cost function, let* $\nu < \varepsilon(r)$, *and consider the computational problem* $\{\Xi, \Omega_{\mathcal{T},\nu}, \mathbb{R}^r, \Lambda_{d,r,r}\}$, *where* Ξ *is given by (8.1) and* $\Omega_{\mathcal{T},\nu}$ *is given by (8.6). Then, there exits a set of* Δ_1-*information* $\hat{\Lambda}_{d,r,r} \in \mathcal{L}^1(\Lambda_{d,r,r})$ *so that, for the computational problem* $\{\Xi, \Omega_{\mathcal{T},\nu}, \mathbb{R}^r, \hat{\Lambda}_{d,r,r}\}$, *we have* $\epsilon_{\mathbb{P}hB}^s(\mathrm{p}) \geq 1/4$, *for all* $\mathrm{p} \in [0, 1/2)$.

10. Tools for proving impossibility results – Breakdown epsilons and exit flags

The main tools can be summed up in the following three statements. We commence with a proposition describing the relationship between the different Breakdown epsilons.

10.1. **The key mechanisms for the Breakdown epsilons.** The connection between the different Breakdown epsilons can be summarised in Proposition 10.1 further below.

Proposition 10.1. *Given a computational problem* $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ *with* $\Lambda = \{f_k \mid k \in \mathbb{N}, k \leq |\Lambda|\}$, *and* $p, q \in (0,1)$ *with* $p \leq q$, *we have*

$$\epsilon_{\mathbb{P}B}^s(q) \leq \epsilon_{\mathbb{P}B}^s(p) \leq \epsilon_B^s, \tag{10.1}$$

$$\epsilon_{\mathbb{P}B}^w(q) \leq \epsilon_{\mathbb{P}B}^w(p) \leq \epsilon_B^w, \tag{10.2}$$

$$\epsilon_{\mathbb{P}hB}^s(p) \leq \epsilon_{\mathbb{P}B}^s(p) \leq \epsilon_{\mathbb{P}B}^w(p), \quad and \tag{10.3}$$

$$\epsilon_B^s \leq \epsilon_B^w. \tag{10.4}$$

Proof of Proposition 10.1. We start with (10.1), and observe that $\epsilon_{\mathbb{P}B}^s(q) \leq \epsilon_{\mathbb{P}B}^s(p)$ follows directly from the definition. To see that $\epsilon_{\mathbb{P}B}^s(p) \leq \epsilon_B^s$ we argue by contradiction and suppose that $\epsilon_{\mathbb{P}B}^s(p) > \epsilon_B^s$. Then there exists an $\epsilon > 0$ such that $\epsilon_{\mathbb{P}B}^s(p) > \epsilon > \epsilon_B^s$. Hence,

$$\forall \Gamma^{\mathrm{ran}} \in \mathrm{RGA} \; \exists \iota \in \Omega \text{ such that } \mathbb{P}_\iota(\mathrm{dist}_{\mathcal{M}}(\Gamma_\iota^{\mathrm{ran}}, \Xi(\iota)) > \epsilon) > p. \tag{10.5}$$

However, since $\epsilon > \epsilon_{\mathrm{B}}^{\mathrm{s}}$ there exists a general algorithm Γ such that for all $\iota \in \Omega$, we have that $\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) \leq \epsilon$. Since Γ can be seen as an RGA with $X = \{\Gamma\}$ and $\mathbb{P}_{\iota}(\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\iota)) > \epsilon) = 0$, which violates (10.5). Note that the argument to establish (10.2) is identical to the proof of (10.1). Finally, we notice that (10.3) and (10.4) follow directly from the definitions. $\qquad\square$

As the next two propositions will rely on the use of $D_{\Gamma}(\iota)$, the minimum number of digits on the input, together with randomised general algorithms, it will be opportune to immediately state and prove that $D_{\Gamma}(\iota)$ is a random variable. Concretely, we have the following two lemmas:

Lemma 10.2. *Let Γ^{ran} be an RGA for a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, where $\Lambda = \{f_j \mid j \leq |\Lambda|\}$ is countable, and let X, $\mathcal{F}$, and $\{\mathbb{P}_{\iota}\}_{\iota \in \Omega}$ be as in Definition 9.24. Then,*

(i) for every $\iota \in \Omega$ and every $f \in \Lambda$, $\{\Gamma \in X \mid f \in \Lambda_{\Gamma}(\iota)\} \in \mathcal{F}$, and

(ii) for every bijection $\theta : \mathbb{N} \to \mathbb{N}$ and every $n \in \mathbb{N}$, we have

$$\{\Gamma \in X \mid \Lambda_{\Gamma}(\iota) \subset \{f_{\theta(1)}, \ldots, f_{\theta(n)}\}\} \in \mathcal{F}.$$

In particular, Γ^{ran} is an RGA for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ independently of the enumeration of Λ.

Proof. Proof of (i): Take arbitrary $\iota \in \Omega$ and $f_j \in \hat{\Lambda}$. The claim follows from (Pii) in Definition 9.24 after observing that $\{\Gamma \in X \mid f_j \in \Lambda_{\Gamma}(\iota)\} = \{\Gamma \in X \mid T_{\Gamma}(\iota) \leq j\} \setminus \{\Gamma \in X \mid T_{\Gamma}(\iota) \leq j-1\}$.

Proof of (ii): Consider an arbitrary $n \in \mathbb{N}$ and define the set $S = \{\theta(j) \mid j \in \mathbb{N}, j \leq n\}$. Then, for every $\iota \in \Omega$,

$$\{\Gamma \in X \mid \Lambda_{\Gamma}(\iota) \subset \{f_{\theta(1)}, \ldots, f_{\theta(n)}\}\} = X \setminus \bigcup_{j \in \mathbb{N} \setminus S} \{\Gamma \in X \mid f_j \in \Lambda_{\Gamma}(\iota)\},$$

which must be an element of $\mathcal{F}$, since each of the sets $\{\Gamma \in X \mid f_j \in \Lambda_{\Gamma}(\iota)\}$ is in $\mathcal{F}$, by the already established item (i), and $\mathcal{F}$ is a σ-algebra. $\qquad\square$

Lemma 10.3. *Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem with Λ countable and let $\hat{\Lambda} = \{f_{k,m} \mid k \leq |\Lambda|, m \in \mathbb{N}\} \in \mathcal{L}^1(\Lambda)$ be arbitrary. Furthermore, let Γ^{ran} be an RGA for $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ with X and $\mathcal{F}$ as in Definition 9.24. Then, for each $\iota \in \Omega$, the function $D_{\Gamma^{\mathrm{ran}}}(\iota) : X \to \mathbb{N} \cup \{\infty\}$ defined by $\Gamma \mapsto D_{\Gamma}(\iota)$ is $\mathcal{F}$-measurable.*

Proof. Consider an arbitrary $n \in \mathbb{N}$ and define the set $S = \{(k, m) \in \mathbb{N}^2 \mid k \leq |\Lambda|, m \leq n\}$. Then, for each $\iota \in \Omega$, we have

$$\{\Gamma \in X \mid D_{\Gamma}(\iota) \leq n\} = \{\Gamma \in X \mid \hat{\Lambda}_{\Gamma}(\iota) \subset \{f_{k,m} \mid (k,m) \in S\}\}$$

$$= X \setminus \bigcup_{(k,m) \in \mathbb{N}^2 \setminus S} \{\Gamma \in X \mid f_{k,m} \in \hat{\Lambda}_{\Gamma}(\iota)\} \in \mathcal{F},$$

since each of the sets $\{\Gamma \in X \mid f_{k,m} \in \Lambda_{\Gamma}(\iota)\}$ is in $\mathcal{F}$, by item (i) of Lemma 10.2, and $\mathcal{F}$ is a σ-algebra. Then clearly $\{\Gamma \in X \mid D_{\Gamma}(\iota) = \infty\} = X \setminus \bigcup_{n \in \mathbb{N}} \{\Gamma \in X \mid D_{\Gamma}(\iota) \leq n\} \in \mathcal{F}$, and thus $D_{\Gamma^{\mathrm{ran}}}(\iota)$ is measurable, as desired. $\qquad\square$

Remark 10.4. It will be useful throughout the paper to note that any of the breakdown epsilons of a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ is at least as large as the corresponding breakdown epsilon of any other computational problem $\{\Xi, \Omega', \mathcal{M}, \Lambda'\}$ with $\Omega' \subset \Omega$ and $\Lambda' = \{f \mid_{\Omega'} \mid f \in \Lambda\}$.

The next proposition serves as the key building block in the impossibility results in all our main theorems except for Theorem 5.1. Note that the proposition is about arbitrary computational problems, and is hence also a tool for demonstrating lower bounds on the breakdown epsilons for general computational problems.

Proposition 10.5. *Let $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$ be a computational problem with $\Lambda = \{f_k \mid k \in \mathbb{N}, k \leq |\Lambda|\}$ countable, and let $\{\iota_n^1\}_{n=1}^\infty$, $\{\iota_n^2\}_{n=1}^\infty$ be sequences in Ω. Consider the following conditions:*

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

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

(c) *There is an $\iota^0 \in \Omega$ such that for every $k \leq |\Lambda|$ we have that (b) is satisfied with $c_k = f_k(\iota^0)$.*

(d) *There is an $\iota^0 \in \Omega$ for which condition (c) holds and additionally $\iota_n^1 = \iota^0$, for all $n \in \mathbb{N}$.*

(e) *Ξ is single-valued, $\mathcal{M}$ is a totally ordered set, and $\inf \mathcal{B}_\delta(S^1) > \sup S^2$, for some $\delta > 0$.*

Depending on which of the conditions (a) – (e) are fulfilled, there exists a $\hat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that some of the following claims about the computational problem $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ hold:

(i) *$\epsilon_{\mathrm{B}}^{\mathrm{w}} \geq \epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) \geq \kappa/2$ for $\mathrm{p} \in [0, 1/2)$,*

(ii) *$\epsilon_{\mathrm{B}}^{\mathrm{s}} \geq \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for $\mathrm{p} \in [0, 1/2)$ and $\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for $\mathrm{p} \in [0, 1/3)$,*

(iii) *$\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for $\mathrm{p} \in [0, 1/2)$.*

(iv) *$\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\} \notin \Sigma_1^G$.*

Concretely, if (a) and (b) are fulfilled, then (i) holds, if (a) – (c) are fulfilled, then (i) and (ii) hold, if (a) – (d) are fulfilled, then (i) – (iii) hold, and finally, if (a) – (e) are fulfilled, then (i) – (iv) hold.

Proof of Proposition 10.5. We begin by defining the inputs $\iota_n \in \Omega$, for $n \geq 1$, according to $\iota_{2n} = \iota_{n+1}^1, \iota_{2n-1} = \iota_{n+1}^2$. Now, if only (a) and (b) are fulfilled, we can assume w.l.o.g. $\Omega = \{\iota_n \mid n \geq 1\}$ (see Remark 10.4), and, in the cases when (c) holds as well, we define $\iota_0 := \iota^0$ and assume w.l.o.g. $\Omega = \{\iota_n \mid n \geq 0\}$.

Our aim now is to produce the desired Δ_1-information for Ω. For $m, n \in \mathbb{N}$ and $k \leq |\Lambda|$, choose $d_k^{n,m} \in \mathbb{D}_m + i\mathbb{D}_m$ such that $|f_k(\iota_n) - d_k^{n,m}| \leq 2^{-m}$ as well as $c_k^m \in \mathbb{D}_m + i\mathbb{D}_m$ such that $|c_k^m - c_k| \leq 2^{-m}/\sqrt{2}$. Now, for $k \leq |\Lambda|$ and $m \in \mathbb{N}$, define $f_{k,m} : \Omega \to \mathbb{D}_m + i\mathbb{D}_m$ according to

$$f_{k,m}(\iota_n) = \begin{cases} d_k^{n,m}, & \text{if } 1 \leq n \leq m \\ c_k^m, & \text{if } n = 0 \text{ or } n > m \end{cases}, \quad \text{for } \iota_n \in \Omega. \tag{10.6}$$

We claim that $\hat{\Lambda} := \{f_{k,m} \mid k \leq |\Lambda|, m \in \mathbb{N}\}$ provides Δ_1-information for $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$. To this end, first note that, due to assumption (b) and our construction of $\{\iota_n\}_{n \geq 1}$, we have $|c_k - f_k(\iota_n)| \leq 2^{-(n+2)}$, for all $n \geq 1$. Now, for $n, m \in \mathbb{N}$ with $n > m$, we have $2^{-(n+2)} \leq 2^{-(m+3)}$, and therefore

$$|f_{k,m}(\iota_n) - f_k(\iota_n)| = |c_k^m - f_k(\iota_n)| \leq |c_k^m - c_k| + |c_k - f_k(\iota_n)|$$
$$\leq 2^{-m}/\sqrt{2} + 2^{-(n+2)} \leq (1/\sqrt{2} + 1/8) \cdot 2^{-m} < 2^{-m}.$$

Similarly, in the cases when (c) holds so that $\iota_0 \in \Omega$, we have $|f_{k,m}(\iota_0) - f_k(\iota_0)| = |c_k^m - f_k(\iota_0)| = |c_k^m - c_k| \leq 2^{-m}/\sqrt{2} < 2^{-m}$. Finally, for $1 \leq n \leq m$ we have $|f_{k,m}(\iota_n) - f_k(\iota_n)| = |d_k^{n,m} - f_k(\iota_n)| \leq 2^{-m}$ by the above definition of $d_k^{n,m}$. Therefore $\hat{\Lambda}$ provides Δ_1-information for the computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}$, as desired.

Proof of (a), (b) $\implies$ (i): We argue by contradiction and assume that

$$\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) < \kappa/2, \qquad \text{for some } \mathrm{p} \in (0, 1/2). \tag{10.7}$$

for the computational problem $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$. Therefore, there exist an $N \in \mathbb{N}$ and a $\Gamma^{\mathrm{ran}} \in \mathrm{RGA}$ such that

$$\mathbb{P}_\iota(\mathrm{dist}_{\mathcal{M}}(\Gamma_\iota^{\mathrm{ran}}, \Xi(\iota)) \geq \kappa/2 \text{ or } T_{\Gamma^{\mathrm{ran}}}(\iota) > N) \leq \mathrm{p}, \quad \forall \iota \in \Omega. \tag{10.8}$$

Now, recalling the definitions of T_Γ and D_Γ, we see that there must exist an $M \in \mathbb{N}$ such that

$$D_\Gamma(\iota) > M \implies T_\Gamma(\iota) > N, \tag{10.9}$$

for all algorithms Γ for $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ and all $\iota \in \Omega$. Next, let the measurable space $(X, \mathcal{F})$ and the probability measures $\{\mathbb{P}_{\iota_n}\}_{\iota_n \in \Omega}$ associated with Γ^{ran} be as in Definition 9.24 of an RGA. We now define the following sets that form the basis for the rest of the argument:

$$F_j := \{\Gamma \in X \mid \mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_{M+j}), \Xi(\iota_{M+j})) \geq \kappa/2 \text{ or } D_\Gamma(\iota_{M+j}) > M\}, \qquad j = 1, 2,$$

and

$$\hat{F}_2 = \{\Gamma \in X \mid \mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_{M+2}), \Xi(\iota_{M+2})) \geq \kappa/2\},$$
$$\mathring{F}_2 = \{\Gamma \in X \mid D_\Gamma(\iota_{M+2}) > M\}.$$

Note that (Pi) in Definition 9.24, the continuity of the metric $d_{\mathcal{M}}$, and Lemma 10.3 together imply that F_1, F_2, $\hat{F}_2$ and $\mathring{F}_2$ are measurable. Also observe that (10.8) and (10.9) imply that $\mathbb{P}_{\iota_{M+j}}(F_j) \leq \mathrm{p}$ for $j = 1, 2$.

Claim 1: We now claim the following.

(I) $X = F_1 \cup \hat{F}_2$.
(II) $\mathbb{P}_{\iota_{M+1}}(\hat{F}_2 \cap \mathring{F}_2^c) = \mathbb{P}_{\iota_{M+2}}(\hat{F}_2 \cap \mathring{F}_2^c)$.
(III) $\mathbb{P}_{\iota_{M+1}}(\mathring{F}_2) = \mathbb{P}_{\iota_{M+2}}(\mathring{F}_2)$

To see (I) we argue as follows. Consider an arbitrary $\Gamma \in X$. Suppose by way of contradiction that $\Gamma \notin F_1$ and $\Gamma \notin \hat{F}_2$. In particular, $D_\Gamma(\iota_{M+1}) \leq M$. Consequently, whenever $f_{k,m} \in \hat{\Lambda}_\Gamma(\iota_{M+1})$ we must have $m \leq M$. Therefore, by (10.6) it follows that $f_{k,m}(\iota_{M+1}) = f_{k,m}(\iota_{M+2}) = c_k^m$.

Hence, by property (iii) in Definition 9.3 of a general algorithm, it follows that $\hat{\Lambda}_\Gamma(\iota_{M+1}) = \hat{\Lambda}_\Gamma(\iota_{M+2})$. Thus, by (ii) in Definition 9.3, $\Gamma(\iota_{M+1}) = \Gamma(\iota_{M+2})$. The assumption that $\Gamma \notin F_1$ and $\Gamma \notin \hat{F}_2$ yields

$$\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_{M+1}), \Xi(\iota_{M+1})) < \kappa/2 \qquad \text{and}$$
$$\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_{M+1}), \Xi(\iota_{M+2})) = \mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_{M+2}), \Xi(\iota_{M+2})) < \kappa/2.$$

Therefore $\inf_{\xi_1 \in \Xi(\iota_{M+1}), \xi_2 \in \Xi(\iota_{M+2})} d_{\mathcal{M}}(\xi_1, \xi_2) < \kappa$, which contradicts (a), and hence we conclude that $\Gamma \in F_1$ or $\Gamma \in \hat{F}_2$, establishing (I).

To prove (II) and (III) it suffices to demonstrate that

$$\mathbb{P}_{\iota_{M+1}}(E \cap \mathring{F}_2^c) = \mathbb{P}_{\iota_{M+2}}(E \cap \mathring{F}_2^c) \qquad \forall E \in \mathcal{F}. \tag{10.10}$$

Indeed, given (10.10) (II) follows immediately since $\hat{F}_2 \in \mathcal{F}$, whereas (III) follows by letting $E = X$ since then $\mathbb{P}_{\iota_{M+1}}(\mathring{F}_2) = 1 - \mathbb{P}_{\iota_{M+1}}(\mathring{F}_2^c) = 1 - \mathbb{P}_{\iota_{M+2}}(\mathring{F}_2^c) = \mathbb{P}_{\iota_{M+2}}(\mathring{F}_2)$. To show (10.10) consider any $E \in \mathcal{F}$ with $E \cap \mathring{F}_2^c \neq \varnothing$ (if instead $E \cap \mathring{F}_2^c$ were empty there is nothing to prove). Now, for $\Gamma \in E \cap \mathring{F}_2^c$ and $f_{k,m} \in \hat{\Lambda}_\Gamma(\iota_{M+2})$, it follows from

$$\mathring{F}_2^c = \{\Gamma \in X \mid D_\Gamma(\iota_{M+2}) \leq M\}$$

that $m \leq M$. Hence, by (10.6), we have $f_{k,m}(\iota_{M+1}) = f_{k,m}(\iota_{M+2}) = c_k^m$, and thus (10.10) follows immediately by (Piii) in Definition 9.24 of a randomised general algorithm, completing the proof of the claim.

Armed with the claim we are ready to derive the desired contradiction. In particular,

$$\begin{aligned}
1 &\leq \mathbb{P}_{\iota_{M+1}}(F_1) + \mathbb{P}_{\iota_{M+1}}(\hat{F}_2) && \text{by (I)} \\
&= \mathbb{P}_{\iota_{M+1}}(F_1) + \mathbb{P}_{\iota_{M+1}}(\hat{F}_2 \cap \mathring{F}_2^c) + \mathbb{P}_{\iota_{M+1}}(\hat{F}_2 \cap \mathring{F}_2) \\
&\leq \mathbb{P}_{\iota_{M+1}}(F_1) + \mathbb{P}_{\iota_{M+2}}(\hat{F}_2 \cap \mathring{F}_2^c) + \mathbb{P}_{\iota_{M+1}}(\mathring{F}_2) && \text{by (II)} \\
&= \mathbb{P}_{\iota_{M+1}}(F_1) + \mathbb{P}_{\iota_{M+2}}(\hat{F}_2 \cap \mathring{F}_2^c) + \mathbb{P}_{\iota_{M+2}}(\mathring{F}_2) && \text{by (III)} \\
&\leq \mathbb{P}_{\iota_{M+1}}(F_1) + \mathbb{P}_{\iota_{M+2}}(F_2) \leq 2\mathrm{p} < 1 && \text{since } \hat{F}_2 \subset F_2,
\end{aligned}$$

which yields the final contradiction.

Proof of (a) – (c) $\implies$ (i) – (ii) : Item (i) holds by the argument above. For item (ii), we will first show that $\epsilon_{\mathbb{P}B}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for all $\mathrm{p} \in [0, 1/3)$ and then demonstrate that $\epsilon_{\mathrm{B}}^{\mathrm{s}} \geq \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for all $\mathrm{p} \in [0, 1/2)$, for which it will suffice to show that $\epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \geq \kappa/2$ for $\mathrm{p} \in [0, 1/2)$ because by Proposition 10.1 we have that $\epsilon_{\mathrm{B}}^{\mathrm{s}} \geq \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p})$. To this end, first recall from the beginning of the proof that the set Ω now contains the element ι_0.

Suppose by way of contradiction that $\epsilon_{\mathbb{P}B}^{\mathrm{s}}(p) < \kappa/2$ for some $\mathrm{p} \in [0, 1/3)$, so that there exists a $\Gamma^{\mathrm{ran}} \in \mathrm{RGA}$ with the corresponding measurable space $(X, \mathcal{F})$ and probability measures $\{\mathbb{P}_{\iota_n}\}_{\iota_n \in \Omega}$ such that

$$\mathbb{P}_{\iota_n}(F^n) \leq \mathrm{p}, \quad \text{for all } n \in \mathbb{N} \cup \{0\}, \tag{10.11}$$

where we define

$$F^n := \{\Gamma \in X \mid \mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_n), \Xi(\iota_n)) \geq \kappa/2\}, \qquad n \in \mathbb{N} \cup \{0\}. \tag{10.12}$$

Note that by (Pi) in Definition 9.24 and the continuity of the metric $d_{\mathcal{M}}$ it follows that the F^n are measurable. Additionally, for $\iota \in \Omega$ and $n \in \mathbb{N} \cup \{0\}$, we define $\mathcal{G}^n(\iota) := \{\Gamma \in X \mid D_\Gamma(\iota) \leq n\}$, i.e., the collection of general algorithms whose number of required input digits is bounded by $n \in \mathbb{N}$ when applied to ι, and note that these sets are measurable due to Lemma 10.3. Now, $X \setminus \bigcup_{n=1}^\infty \mathcal{G}^n(\iota_0) \subset F^0$, as every $\Gamma \in X$ for which $D_\Gamma(\iota_0) = \infty$ must have $|\hat{\Lambda}_\Gamma(\iota_0)| = \infty$ (by Definition 9.22) and therefore $\Gamma(\iota_0) = \mathrm{NH}$ (by Definition 9.3), which then implies $\mathrm{dist}_{\mathcal{M}}(\Gamma(\iota_0), \Xi(\iota_0)) = \infty$ (by Definition (9.5) of the extended metric on $\mathcal{M} \cup \{\infty\}$). Therefore, $\mathbb{P}_{\iota_0}(F^0) \leq \mathrm{p}$ implies

$$\mathbb{P}_{\iota_0}\left(X \setminus \bigcup_{n=1}^\infty \mathcal{G}^n(\iota_0)\right) \leq \mathrm{p}. \tag{10.13}$$

We next make the following claim.

Claim 2: There is an $n \in \mathbb{N}$ such that we have the following:

(I) $\mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) > 2\mathrm{p}$,

(II) $\mathcal{G}^n(\iota_0) \subset F^{n+1} \cup F^{n+2}$,

(III) $\mathbb{P}_{\iota_0}(F^{n+1} \cap \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_{n+1}}(F^{n+1} \cap \mathcal{G}^n(\iota_0))$, and $\mathbb{P}_{\iota_0}(F^{n+2} \cap \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_{n+2}}(F^{n+2} \cap \mathcal{G}^n(\iota_0))$.

The contradiction arises by combining these results: indeed, by (I) and (II), $2\mathrm{p} < \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_0}((F^{n+1} \cap \mathcal{G}^n(\iota_0)) \cup (F^{n+2} \cap \mathcal{G}^n(\iota_0)))$ and by (III) we get that $\mathbb{P}_{\iota_0}(F^{n+1} \cap \mathcal{G}^n(\iota_0)) + \mathbb{P}_{\iota_0}(F^{n+2} \cap \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_{n+1}}(F^{n+1} \cap \mathcal{G}^n(\iota_0)) + \mathbb{P}_{\iota_{n+2}}(F^{n+2} \cap \mathcal{G}^n(\iota_0))$. Therefore

$$2\mathrm{p} < \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_0}\left((\mathcal{G}^n(\iota_0) \cap F^{n+1}) \cup (\mathcal{G}^n(\iota_0) \cap F^{n+2})\right) \tag{10.14}$$

$$\leq \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0) \cap F^{n+1}) + \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0) \cap F^{n+2}) \tag{10.15}$$

$$= \mathbb{P}_{\iota_{n+1}}(\mathcal{G}^n(\iota_0) \cap F^{n+1}) + \mathbb{P}_{\iota_{n+2}}(\mathcal{G}^n(\iota_0) \cap F^{n+2}) \tag{10.16}$$

$$\leq \mathbb{P}_{\iota_{n+1}}(F^{n+1}) + \mathbb{P}_{\iota_{n+2}}(F^{n+2}) \leq 2\mathrm{p} \qquad \text{by (10.11),} \tag{10.17}$$

which is the desired contradiction establishing item (ii). It thus remains to prove Claim 2.

For (I), suppose on the contrary that we have $\mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) \leq 2\mathrm{p}$ for all $n \in \mathbb{N}$. Then, as $\mathcal{G}^n(\iota_0) \subset \mathcal{G}^{n+1}(\iota_0)$ for all n, monotone convergence implies

$$\mathrm{p} \geq \mathbb{P}_{\iota_0}\left(X \setminus \bigcup_{n=1}^\infty \mathcal{G}^n(\iota_0)\right) = \lim_{n\to\infty} 1 - \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) \geq 1 - 2\mathrm{p}, \tag{10.18}$$

which is a contradiction since $\mathrm{p} \in [0, 1/3)$. Therefore, there exists an $n \in \mathbb{N}$ for which (I) holds. We now prove (II) and (III) for this n. To this end, we make the intermediary step of showing that

$$\Gamma(\iota_{n+1}) = \Gamma(\iota_{n+2}), \text{ for all } \Gamma \in \mathcal{G}^n(\iota_0). \tag{10.19}$$

Indeed, for $\Gamma \in \mathcal{G}^n(\iota_0)$, it follows immediately from the definition of the set $\mathcal{G}^n$ and the definition of D_Γ that if $f_{k,m} \in \hat{\Lambda}_\Gamma(\iota_0)$, then $m \leq n < n+1 < n+2$. Thus, by the construction of $\hat{\Lambda}$, it follows that $f_{k,m}(\iota_0) = f_{k,m}(\iota_{n+1}) = f_{k,m}(\iota_{n+2})$, for all $f_{k,m} \in \hat{\Lambda}_\Gamma(\iota_0)$. Hence, by property (iii) in Definition 9.3 of a general algorithm, it follows that $\hat{\Lambda}_\Gamma(\iota_0) = \hat{\Lambda}_\Gamma(\iota_{n+1})$, and similarly we get that $\hat{\Lambda}_\Gamma(\iota_0) = \hat{\Lambda}_\Gamma(\iota_{n+2})$. However, the fact that $\hat{\Lambda}_\Gamma(\iota_0) = \hat{\Lambda}_\Gamma(\iota_{n+1}) = \hat{\Lambda}_\Gamma(\iota_{n+2})$ together with (ii) in Definition 9.3 imply that $\Gamma(\iota_0) = \Gamma(\iota_{n+1})$ and $\Gamma(\iota_0) = \Gamma(\iota_{n+2})$, and hence $\Gamma(\iota_{n+1}) = \Gamma(\iota_{n+2})$, establishing (10.19).

We can now show (II). Consider an arbitrary $\Gamma \in \mathcal{G}^n(\iota_0)$. We have shown that $\Gamma(\iota_{n+1}) = \Gamma(\iota_{n+2})$. Hence, if $\Gamma \notin F^{n+1}$ and $\Gamma \notin F^{n+2}$ then $\mathrm{dist}_\mathcal{M}(\Gamma(\iota_{n+1}), \Xi(\iota_{n+1})) < \kappa/2$ and $\mathrm{dist}_\mathcal{M}(\Gamma(\iota_{n+1}), \Xi(\iota_{n+2}))$ $= \mathrm{dist}_\mathcal{M}(\Gamma(\iota_{n+2}), \Xi(\iota_{n+2})) < \kappa/2$ together yield $\inf_{\xi_1 \in \Xi(\iota_{n+1}), \xi_2 \in \Xi(\iota_{n+2})} d_\mathcal{M}(\xi_1, \xi_2) < \kappa$, which would contradict (a). Therefore (II) must hold.

Finally, to prove (III), we note that both $\mathcal{G}^n(\iota_0) \cap F^{n+1}$ and $\mathcal{G}^n(\iota_0) \cap F^{n+2}$ are measurable because $\mathcal{G}^n(\iota_0)$, F^{n+1}, and F^{n+2} are all measurable. We will show the result only for $\mathcal{G}^n(\iota_0) \cap F^{n+1}$ as the corresponding argument for $\mathcal{G}^n(\iota_0) \cap F^{n+2}$ is similar. If $\Gamma \in \mathcal{G}^n(\iota_0) \cap F^{n+1}$ then $\Gamma \in \mathcal{G}^n(\iota_0)$, which by (10.19) implies that $f_{k,m}(\iota_{n+1}) = f_{k,m}(\iota_0)$, for all $f_{k,m} \in \hat{\Lambda}_\Gamma(\iota_0)$. The result (III) now follows immediately from (Piii) in Definition 9.24 of an RGA.

The proof that $\epsilon^{\mathrm{s}}_{\mathbb{P}h\mathrm{B}}(\mathrm{p}) \geq \kappa/2$ for all $\mathrm{p} \in [0, 1/2)$ is almost identical to the argument above, and we will point out the minor differences. We again argue by contradiction assuming that $\epsilon^{\mathrm{s}}_{\mathbb{P}h\mathrm{B}}(\mathrm{p}) < \kappa/2$, and the proof is identical up to (10.13). At this point it follows by Definition 9.28 of a halting randomised general algorithm that $\mathbb{P}_{\iota_0}(\Gamma^{\mathrm{ran}}_{\iota_0} = \mathrm{NH}) = 0$. Hence, (10.13) becomes

$$\mathbb{P}_{\iota_0}\left(X \setminus \bigcup_{n=1}^{\infty} \mathcal{G}^n(\iota_0)\right) = 0 \tag{10.20}$$

and thus 10.18 is to be replaced by

$$0 \geq \mathbb{P}_{\iota_0}\left(X \setminus \bigcup_{n=1}^{\infty} \mathcal{G}^n(\iota_0)\right) = \lim_{n \to \infty} 1 - \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0)) \geq 1 - 2\mathrm{p},$$

which contradicts $\mathrm{p} \in [0, 1/2)$. The rest of the proof is identical to the argument above.

Proof of (a) – (d) $\implies$ (i) –(iii): Items (i) and (ii) have already been established. For (iii), the proof stays close to the proof of (ii). Indeed, the proof is identical to the proof of (ii) up to and including (10.11). Now, however, condition (d) implies that $\iota_{2n} = \iota_n^1 = \iota^0 = \iota_0$ and thus $F^0 = F^{2n}$ for all $n \in \mathbb{N}$. Next, instead of (10.13) we write

$$X = F^0 \cup \bigcup_{n=1}^{\infty} \mathcal{G}^n(\iota_0). \tag{10.21}$$

We then change Claim 2 to the following:

Claim 2′: There is an *odd* $n \in \mathbb{N}$ such that we have the following:

(I′) $\mathbb{P}_{\iota_0}(F^0 \cup \mathcal{G}^n(\iota_0)) > 2\mathrm{p}$

(II′) $\mathcal{G}^n(\iota_0) \subset F^{n+1} \cup F^{n+2}$,

(III′) $\mathbb{P}_{\iota_0}(F^{n+1} \cap \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_{n+1}}(F^{n+1} \cap \mathcal{G}^n(\iota_0))$, and $\mathbb{P}_{\iota_0}(F^{n+2} \cap \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_{n+2}}(F^{n+2} \cap \mathcal{G}^n(\iota_0))$.

The proof of (I′) follows from (10.21) and monotonicity of the sets $F^0 \cup \mathcal{G}^n(\iota_0)$: indeed, if by contradiction $\mathbb{P}_{\iota_0}(F^0 \cup \mathcal{G}^n(\iota_0)) \leq 2p$ for all odd n then

$$1 = \mathbb{P}(X) = \mathbb{P}_{\iota_0}\left(\lim_{n \to \infty} F^0 \cup \mathcal{G}^n(\iota_0)\right) = \lim_{n \to \infty} \mathbb{P}_{\iota_0}(F^0 \cup \mathcal{G}^n(\iota_0)) \leq 2p$$

by the monotone convergence theorem, which contradicts $\mathrm{p} \in [0, 1/2)$. The proofs of part (II′) and (III′) are then identical to those presented in the proof of Claim 2.

The calculation in (10.14), (10.15), (10.16), (10.17) is then changed to

$$2p < \mathbb{P}_{\iota_0}(F^0 \cup \mathcal{G}^n(\iota_0)) = \mathbb{P}_{\iota_0}\left(F^0 \cup (\mathcal{G}^n(\iota_0) \cap F^{n+1}) \cup (\mathcal{G}^n(\iota_0) \cap F^{n+2})\right)$$

$$\leq \mathbb{P}_{\iota_0}(F^0) + \mathbb{P}_{\iota_0}(\mathcal{G}^n(\iota_0) \cap F^{n+2}) \leq 2p \qquad \text{since } F^0 = F^{n+1} \text{ as } n \text{ is odd,}$$

which yields the desired contradiction that establishes that (a) – (d) $\implies$ (i) –(iii).

Proof of (a) – (e) $\implies$ *(i) – (iv)*: Suppose by way of contradiction that there exists a tower of algorithms $\{\Gamma_n\}_{n\in\mathbb{N}}$ of height 1 for $\{\Xi, \Omega, \mathcal{M}, \hat{\Lambda}\}$ such that $\Gamma_n(\iota) \nearrow \Xi(\iota)$ as $n \to \infty$, for all $\iota \in \Omega$. Thus, recalling that $\iota^0 = \iota_n^1$ due to condition (d) and that $\Xi(\iota_n^1) \in S^1$ by (a), there exists an $n_0 \in \mathbb{N}$ so that $\Gamma_{n_0}(\iota_0) \subset \mathcal{B}_\delta\big(\Xi(\iota^0)\big) \subset \mathcal{B}_\delta\big(S^1\big)$.

Now, let M be a sufficiently large odd natural number such that $M > D_{\Gamma_{n_0}}(\iota_0)$. It then follows immediately from the definition of $D_{\Gamma_{n_0}}$ that if $f_{k,m} \in \hat{\Lambda}_{\Gamma_{n_0}}(\iota_0)$, then $m < M$. Thus, by the construction (10.6) of $\hat{\Lambda}$, it follows that $f_{k,m}(\iota_0) = f_{k,m}(\iota_M)$, for all $f_{k,m} \in \hat{\Lambda}_{\Gamma_{n_0}}(\iota_0)$, and hence by property (iii) in Definition 9.3 of a general algorithm, it follows that $\hat{\Lambda}_{\Gamma_{n_0}}(\iota_0) = \hat{\Lambda}_{\Gamma_{n_0}}(\iota_M)$, which further by (ii) in Definition 9.3 implies that $\Gamma_{n_0}(\iota_M) = \Gamma_{n_0}(\iota_0) \in \mathcal{B}_\delta\big(S^1\big)$.

On the other hand, we have $\Gamma_{n_0}(\iota_M) \leq \Xi(\iota_M)$ as we assumed $\Gamma_n(\iota_M) \nearrow \Xi(\iota_M)$ as $n \to \infty$, and hence $\Gamma_{n_0}(\iota_M) \leq \Xi(\iota_M) = \Xi\left(\iota_{(M+1)/2}^2\right)$. Furthermore, by (d) $\Xi\left(\iota_{(M+1)/2}^2\right) \in S^2$ and thus $\Gamma_{n_0}(\iota_M) \leq \sup S^2 < \inf \mathcal{B}_\delta\big(S^1\big)$ where the last inequality follows from (e). This stands in contradiction to $\Gamma_{n_0}(\iota_M) \in \mathcal{B}_\delta\big(\Xi(S^1)\big)$, thereby concluding the proof. $\square$

10.2. Impossibility results for the exit flag.
In this section we will state and prove a result that will be useful when proving the negative results contained in Theorem 5.1.

Proposition 10.6. *Suppose that $\Gamma : \tilde{\Omega} \to \mathcal{M}$ is a general algorithm for a computational problem $\{\Xi, \Omega, \mathcal{M}, \Lambda\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ specified according to Remark 9.39 and assume that the assumption 9.22 holds. Furthermore suppose that there exists an $\iota_0 \in \Omega$ and, for $j = 1, 2$, there exist a set $S^j \subset \mathcal{M}$ and a sequence $\{\iota_n^j\}_{n=0}^\infty \subset \Omega$ satisfying the following:*

(a) $\inf_{\xi_1 \in S^1, \xi_2 \in S^2} d_\mathcal{M}(\xi_1, \xi_2) > 2\kappa$, for some $\kappa > 0$.
(b) $\Xi(\iota_n^j) \subseteq S^j$ for all $n \in \mathbb{N}$ and $j \in \{1, 2\}$.
(c) There exists $x^j \in S^j$ such that $\text{dist}_\mathcal{M}(x^j, \Xi(\iota_n^j)) \to 0$ as $n \to \infty$ for $j = 1, 2$.
(d) For every $f \in \Lambda$ then $|f(\iota_n^j) - f(\iota_0)| \leq 1/2^n$ for all $n \in \mathbb{N}$ and $j = 1, 2$.
(e) $\Xi(\Omega) \subseteq \overline{\mathcal{B}}_{\kappa-\omega}(\Xi(\iota_0)) \cup \overline{\mathcal{B}}_{\kappa-\omega}\big(x^1\big) \cup \overline{\mathcal{B}}_{\kappa-\omega}\big(x^2\big)$, for some $\omega \in (0, \kappa)$.

Then, for the exit flag problem $\{\Xi^E, \tilde{\Omega}, \{0, 1\}, \tilde{\Lambda}\}$ relative to Γ, as specified in (9.21), we have $\epsilon_{\mathbb{PB}}^s(p) \geq 1/2$. Furthermore, whenever $\mathcal{M} = \mathbb{R}^N$, for some $N \in \mathbb{N}$, if we additionally assume that

(f) $\text{dist}_\mathcal{M}(x^j, \Xi(\iota^0)) < \omega$, for $j \in \{1, 2\}$,

then there exits a $\tilde{\Lambda}^+ \in \mathcal{L}^{\mathcal{O}, \omega, \tilde{\Xi}}(\tilde{\Lambda})$ such that, for the computational problem $\{\Xi^E, \tilde{\Omega}, \{0, 1\}, \tilde{\Lambda}^+\}$, we have $\epsilon_{\mathbb{PB}}^s(p) \geq 1/2$.

Proof. Using the sequences $\{\iota_n^1\}$, $\{\iota_n^2\}$ and the input $\iota^0 \in \Omega$, we define the following inputs in $\tilde{\Omega}$.

$$\tilde{\iota}_n^1 := \{(\underbrace{f_j(\iota^0), f_j(\iota^0), \ldots, f_j(\iota^0)}_{n \text{ times}}, f_j(\iota_n^1), f_j(\iota_n^1), \ldots)\}_{j \in \beta},$$

$$\tilde{\iota}_n^2 := \{(\underbrace{f_j(\iota^0), f_j(\iota^0), \ldots, f_j(\iota^0)}_{n \text{ times}}, f_j(\iota_n^2), f_j(\iota_n^2), \ldots)\}_{j \in \beta}, \tag{10.22}$$

$$\tilde{\iota}^0 := \{(f_j(\iota^0), f_j(\iota^0), \ldots, f_j(\iota^0), f_j(\iota^0), f_j(\iota^0), \ldots)\}_{j \in \beta},$$

where β is the indexing set (recalling Definition 9.14). By assumption (d), each of $\tilde{\iota}^0$, $\tilde{\iota}_n^1$ and $\tilde{\iota}_n^2$ are in $\tilde{\Omega}$, for every $n \in \mathbb{N}$. We will prove each of the following:

A) There exists $N_0 \in \mathbb{N}$ such that, for all $n > N_0$, $\Gamma(\tilde{\iota}_n^1) = \Gamma(\tilde{\iota}_n^2) = \Gamma(\tilde{\iota}^0)$ and at least one of the following occurs:

   Ai) $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^1), \tilde{\Xi}(\tilde{\iota}_n^1)) > \kappa$ for all $n > N_0$.

   Aii) $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^2), \tilde{\Xi}(\tilde{\iota}_n^2)) > \kappa$ for all $n > N_0$.

B) There exists $N_1 \in \mathbb{N}$ such that at least one of the following occurs:

   Bi) $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \tilde{\Xi}(\tilde{\iota}^0)) \leq \kappa$.

   Bii) $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^1), \tilde{\Xi}(\tilde{\iota}_n^1)) \leq \kappa$ for all $n > N_1$.

   Biii) $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^2), \tilde{\Xi}(\tilde{\iota}_n^2)) \leq \kappa$ for all $n > N_1$.

C) There exists a sequence $\{\tilde{\iota}_n\}_{n=0}^{\infty} \subseteq \tilde{\Omega}$ (constructed from ι^0 and subsequences of $\{\iota_n^1\}$ and $\{\iota_n^2\}$) satisfying the following:

   Ci) For all $n \in \mathbb{N}$ both $\Xi^E(\tilde{\iota}_{2n}) = 1$ and $\Xi^E(\tilde{\iota}_{2n-1}) = 0$.

   Cii) For all $m \leq n$ and $j \in \beta$ we have $\tilde{f}_{j,m}(\tilde{\iota}_n) = \tilde{f}_{j,m}(\tilde{\iota}_0)$, where the $\tilde{f}_{j,m}$ are as given in Definition 9.14.

   Ciii) Either $\tilde{\iota}_{2n} = \tilde{\iota}_0$ for all n or $\tilde{\iota}_{2n-1} = \tilde{\iota}_0$ for all n.

   Moreover, under assumption (f) we have

   Civ) There exists a dyadic vector $v \in \mathbb{D}^M$ such that, for all n, $\text{dist}_{\mathcal{M}}(v, \tilde{\Xi}(\tilde{\iota}_n)) < \omega$.

Proof of A): The proof of this step is similar to the proof of Proposition 10.5. Recall the definition of the number of digits on the input

$$D_{\Gamma}(\tilde{\iota}) := \sup\{k \in \mathbb{N} \,|\, \exists j \in \beta \text{ with } \tilde{f}_{j,k} \in \tilde{\Lambda}_{\Gamma}(\tilde{\iota})\}.$$

required by Γ, and set $N_0 = T_{\Gamma}(\tilde{\iota}^0)$, which is guaranteed to be finite since the assumption 9.22 implies that $\Gamma(\tilde{\iota}^0) \neq \text{NH}$. Note that by the definition of the sequences $\{\iota_n^1\}_{n=1}^{\infty}$ and $\{\iota_n^2\}_{n=1}^{\infty}$ and $\tilde{\iota}^0$ in (10.22), we have that $f(\tilde{\iota}_0) = f(\tilde{\iota}_n^1) = f(\tilde{\iota}_n^1)$ for all $f \in \tilde{\Lambda}_{\Gamma}(\tilde{\iota}_0)$ whenever $n \geq N_0$. Thus, by (iii) in Definition 9.3 of a general algorithm, it follows that $\tilde{\Lambda}_{\Gamma}(\tilde{\iota}_0) = \tilde{\Lambda}_{\Gamma}(\tilde{\iota}_n^1) = \tilde{\Lambda}_{\Gamma}(\tilde{\iota}_n^2)$. Consequently, by (ii) and (iii) in Definition 9.3 we get that, for some $x \in \mathcal{M}$,

$$x = \Gamma(\tilde{\iota}_0) = \Gamma(\tilde{\iota}_n^1) = \Gamma(\tilde{\iota}_n^2), \quad \text{for } n \geq N_0. \tag{10.23}$$

Suppose now that neither Ai) nor Aii) hold. Then there are $n_1 \geq N_0$ and $n_2 \geq N_0$ such that both $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota}_{n_1}^1)) = \text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_{n_1}^1), \tilde{\Xi}(\tilde{\iota}_{n_1}^1)) \leq \kappa$ and $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota}_{n_2}^2)) = \text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^2), \tilde{\Xi}(\tilde{\iota}_n^2)) \leq \kappa$. But then by assumption (b) we have $\text{dist}_{\mathcal{M}}(x, S^1) \leq \kappa, \text{dist}_{\mathcal{M}}(x, S^2) \leq \kappa$ and hence $\text{dist}_{\mathcal{M}}(S^1, S^2) \leq 2\kappa$, contradicting assumption (a).

Proof of B): By the assumption 9.22 on Γ, there exist a $y \in \Xi(\Omega)$ and an $\epsilon > 0$ such that $d_{\mathcal{M}}(x, y) = \alpha - \epsilon \leq \omega - \epsilon$ where x is the common value in (10.23). Moreover, by assumption (e), we have $y \in \overline{\mathcal{B}}_{\kappa-\omega}(\Xi(\iota^0)) \cup \overline{\mathcal{B}}_{\kappa-\omega}(x^1) \cup \overline{\mathcal{B}}_{\kappa-\omega}(x^2)$. We now split into two cases:

Case 1: $y \in \overline{\mathcal{B}}_{\kappa-\omega}(\Xi(\iota^0))$. In this case, there exists $y^2 \in \Xi(\iota^0)$ (depending on ϵ) such that $\text{dist}_{\mathcal{M}}(y, y^2) \leq \kappa - \omega$. Thus

$$\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \tilde{\Xi}(\tilde{\iota}^0)) = \text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \Xi(\iota^0)) \leq \text{dist}_{\mathcal{M}}(x, y^2) \leq \text{dist}_{\mathcal{M}}(x, y) + \text{dist}_{\mathcal{M}}(y, y^2)$$
$$\leq (\omega - \epsilon) + (\kappa - \omega) < \kappa,$$

and so Bi) is satisfied.

Case 2: $y \in \overline{\mathcal{B}}_{\kappa-\omega}(x^j)$, *for some* $j \in \{1, 2\}$. We take $N_1 \geq N_0$ to be sufficiently large so that, for every $n > N_1$, there is a $v^n \in \Xi(\iota_n^j)$ with $d_{\mathcal{M}}(x^j, v^n) \leq \epsilon$. The existence of such an N_1 is guaranteed

by assumption (c). Then, for $n > N_1$,

$$
\begin{aligned}
\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^j), \tilde{\Xi}(\tilde{\iota}_n^j)) = \mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}_n^j), \Xi(\iota_n^j)) &\leq \mathrm{dist}_{\mathcal{M}}(x, v^n) \\
&\leq \mathrm{dist}_{\mathcal{M}}(x, y) + \mathrm{dist}_{\mathcal{M}}(y, x^j) + \mathrm{dist}_{\mathcal{M}}(x^j, v^n) \\
&\leq (\omega - \epsilon) + (\kappa - \omega) + \epsilon = \kappa.
\end{aligned}
$$

If $j = 1$ then we have shown that Bii) occurs, whereas if $j = 2$ we have shown that Biii) holds.

Proof of C): To construct $(\tilde{\iota}_n)_{n=1}^{\infty}$, we distinguish two cases depending on whether Bi) is true or false.

Case 1: $\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \tilde{\Xi}(\tilde{\iota}^0)) \leq \kappa$ *occurs.* We set $\tilde{\iota}_{2n} = \tilde{\iota}^0$ for all n so that Ciii) holds and $\Xi^E(\tilde{\iota}_{2n}) = 1$. To define $\tilde{\iota}_{2n-1}$, we distinguish between whether Ai) or Aii) holds. Suppose for now that Ai) holds. If assumption (f) additionally holds, we let $N_2 > N_0$ be such that $\mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota_n^1)) < \omega/2$ for all $n > N_2$ (the existence of N_2 is guaranteed by assumption (c)), and else we let $N_2 = 0$. Now, Ai) implies that $\Xi^E(\tilde{\iota}_{n+N_2}^1) = 0$, for all n, and so setting $\tilde{\iota}_{2n-1} = \tilde{\iota}_{n+N_2}^1$ establishes Ci). Moreover, Cii) follows by (10.22) and the definition of $\tilde{f}_{j,m}$. Under assumption (f), we use the density of the dyadic numbers in $\mathbb{R}$ to choose a dyadic vector v satisfying both $d_{\mathcal{M}}(x^1, v) \leq \omega/2$ and $\mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota^0)) + d_{\mathcal{M}}(x^1, v) < \omega$. Then $\mathrm{dist}_{\mathcal{M}}(v, \tilde{\Xi}(\tilde{\iota}_{2n-1})) \leq \mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota_{n+N_2}^1)) + d_{\mathcal{M}}(x^1, v) < \omega/2 + \omega/2 = \omega$ and similarly $\mathrm{dist}_{\mathcal{M}}(v, \tilde{\Xi}(\tilde{\iota}_{2n})) \leq \mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota^0)) + d_{\mathcal{M}}(x^1, v) < \omega$, allowing us to conclude Civ). If instead Aii) holds, we similarly let $N_2 > N_0$ be such that $\mathrm{dist}_{\mathcal{M}}(x^2, \Xi(\iota_n^2)) < \omega/2$ for all $n > N_2$, provided (f) holds, and else we set $N_2 = 0$. Letting $\tilde{\iota}_{2n-1} = \tilde{\iota}_{n+N_2}^2$ and choosing v to be sufficiently close to x^2 then yields Ci), Cii), and Civ) by the same argument as above.

Case 2: $\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \tilde{\Xi}(\tilde{\iota}^0)) > \kappa$ *occurs.* We set $\tilde{\iota}_{2n-1} = \tilde{\iota}^0$ for all n so that Ciii) holds and $\Xi^E(\tilde{\iota}_{2n-1}) = 0$. To define $\tilde{\iota}_{2n}$ for all n, we distinguish between whether Bii) or Biii) holds, one of which must occur since Bi) is false by the assumption $\mathrm{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}^0), \tilde{\Xi}(\tilde{\iota}^0)) > \kappa$. Suppose for now that Bii) holds. Provided assumption (f) holds, we let $N_2 > N_1$ be such that $\mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota_n^1)) < \omega/2$, for all $n > N_2$ (the existence of N_2 is guaranteed by assumption (c)), and else we set $N_2 = 0$. Then, setting $\tilde{\iota}_{2n} = \tilde{\iota}_{n+N_2}^1$, we have $\Xi^E(\tilde{\iota}_{n+N_2}^1) = 1$ for all n, thus establishing Ci), whereas Cii) follows by (10.22) and the definition of $\tilde{f}_{j,m}$. Under assumption (f) we again the density of the dyadic numbers in $\mathbb{R}$ to choose a $v \in \mathbb{D}^N$ so that both $d_{\mathcal{M}}(v, x^1) < \omega/2$ and $\mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota^0)) + d_{\mathcal{M}}(v, x^1) < \omega$. Then $\mathrm{dist}_{\mathcal{M}}(v, \tilde{\Xi}(\tilde{\iota}_{2n-1})) \leq \mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota^0)) + d_{\mathcal{M}}(v, x^1) < \omega$ and similarly $\mathrm{dist}_{\mathcal{M}}(v, \tilde{\Xi}(\tilde{\iota}_{2n})) \leq \mathrm{dist}_{\mathcal{M}}(x^1, \Xi(\iota_{n+N_2}^1)) + d_{\mathcal{M}}(v, x^1) < \omega/2 + \omega/2 = \omega$, establishing Civ). Similarly, if Biii) holds, we let $N_2 > N_0$ be such that $\mathrm{dist}_{\mathcal{M}}(\Xi(\iota_n^2), x^2) < \omega$ for all $n > N_2$, provided assumption (f) holds, and else we set $N_2 = 0$. Letting $\tilde{\iota}_{2n-1} = \tilde{\iota}_{n+N_2}^2$ and v sufficiently close to x^2 then yields Ci), Cii), and Civ) by the same argument as before, thus completing the proof of C).

Now that we have shown A) to C), we can show $\epsilon_{\mathbb{PB}}^s(p) \geq 1/2$ for both the exit flag problem $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}$ itself as well as the oracle problem $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}^+\}$, where $\tilde{\Lambda}^+ \in \mathcal{L}^{\mathcal{O}, \omega, \tilde{\Xi}}(\tilde{\Lambda})$ will be specified in due course. The two proofs are very similar so we will discuss differences only when they occur. For the oracle problem (under assumption (f)), we fix a $g^* = (g_1^*, \ldots, g_M^*) : \tilde{\Omega} \to \mathbb{D}^M$ such that, for $\tilde{\iota} \in \tilde{\Omega}$, we have $g^*(\tilde{\iota}) = v$ if $\tilde{\iota} = \tilde{\iota}_n$ for some n, and else $g^*(\tilde{\iota})$ is assigned a value satisfying $\mathrm{dist}_{\mathcal{M}}(g^*(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) < \omega$, but is otherwise arbitrary. Then $\{g_k(\tilde{\iota})\}_{k=1}^M \in \mathcal{B}_\omega^\infty(\tilde{\Xi}(\tilde{\iota}))$, for all $\tilde{\iota} \in \tilde{\Omega}$, and thus, letting $\tilde{\Lambda}^+ = \tilde{\Lambda} \cup \{g_k^*\}_{k=1}^M$, we have $\tilde{\Lambda}^+ \in \mathcal{L}^{\mathcal{O}, \omega, \tilde{\Xi}}(\tilde{\Lambda})$.

Next, we will argue by contradiction and assume that, for some fixed $\mathrm{p} \in [0, 1/2)$,

$$
\exists\, \Gamma^{\mathrm{ran}} \in \mathrm{RGA} \text{ such that } \forall\, \iota \in \tilde{\Omega} \quad \mathbb{P}_\iota(\Gamma_\iota^{\mathrm{ran}} \neq \Xi^E(\iota)) \leq p. \tag{10.24}
$$

Continuing with this $\Gamma^{\mathrm{ran}} \in \mathrm{RGA}$ we define the failure sets F^n by $F^n := \{\Gamma^E \in X \mid \Gamma^E(\tilde{\iota}_n) \neq \Xi^E(\tilde{\iota}_n)\}$, for $n \in \mathbb{N}$. For each $\iota \in \tilde{\Omega}$ and $n \in \mathbb{N}$ we also define the set $\mathcal{G}^n(\iota) := \{\Gamma^E \in X \mid T_{\Gamma^E}(\iota) \leq n\}$. Note that it is clear from (Pi) and (Pii) in Definition 9.24 of an RGA that F^n and $\mathcal{G}^n(\iota)$ are measurable.

By (10.24) it follows that

$$\mathbb{P}_{\tilde{\iota}_n}(F^n) \leq p \quad \forall \iota \in \tilde{\Omega}, \, n \in \mathbb{N}. \tag{10.25}$$

We will show that this leads to the desired contradiction. From here the proof is very similar to the proof of (ii) and (iii) in Proposition 10.5. Suppose we have the following.

D) There exits N_3 such that if $n \geq N_3$ then $\mathbb{P}_{\iota_0}(F^0 \cup \mathcal{G}^n(\iota_0)) > 2p$.

E) For all $n \in \mathbb{N}$, $\mathcal{G}^n(\tilde{\iota}_0) = (F^{n+1} \cap \mathcal{G}^n(\tilde{\iota}_0)) \cup (F^{n+2} \cap \mathcal{G}^n(\tilde{\iota}_0))$.

F) For all $n \in \mathbb{N}$, $\mathbb{P}_{\tilde{\iota}_0}(F^{n+2} \cap \mathcal{G}^n(\tilde{\iota}_0)) = \mathbb{P}_{\tilde{\iota}_{n+2}}(F^{n+2} \cap \mathcal{G}^n(\tilde{\iota}_0))$.

Assuming (D)-(F), we may choose an integer $r > N_3$ so that r is even if $\tilde{\iota}_0 = \tilde{\iota}_{2n-1}$ for all n and r is odd if $\tilde{\iota}_0 = \tilde{\iota}_{2n}$ for all n (we know at least one of these occurs by Ciii)). Then both $\tilde{\iota}_0 = \tilde{\iota}_{r+1}$ and the conclusion of D) holds for this r. Thus, by arguing similarly to in (10.14), (10.15), (10.16), (10.17), we obtain

$$
\begin{aligned}
2p &< \mathbb{P}_{\tilde{\iota}_0}(F^0 \cup \mathcal{G}^r(\iota_0)) = \mathbb{P}_{\tilde{\iota}_0}\left(F^0 \cup (\mathcal{G}^r(\iota_0) \cap F^{r+1}) \cup (\mathcal{G}^r(\tilde{\iota}_0) \cap F^{r+2})\right) && \text{by D) and E)} \\
&\leq \mathbb{P}_{\tilde{\iota}_0}(F^0) + \mathbb{P}_{\tilde{\iota}_0}(\mathcal{G}^r(\tilde{\iota}_0) \cap F^{r+2}) && \text{since } F^0 = F^{r+1} \text{ as } \tilde{\iota}_0 = \tilde{\iota}_{r+1} \\
&= \mathbb{P}_{\tilde{\iota}_0}(F^0) + \mathbb{P}_{\tilde{\iota}_{r+2}}(\mathcal{G}^r(\iota_0) \cap F^{r+2}) \leq \mathbb{P}_{\tilde{\iota}_0}(F^0) + \mathbb{P}_{\tilde{\iota}_{r+2}}(F^{r+2}) \leq 2p && \text{by F) and (10.25),}
\end{aligned}
$$

which yields the desired contradiction. Thus, to complete the proof, it remains to establish (D)-(F). To this end, note that D) follows from $\mathbb{P}_{\iota_0}(X) = 1 > 2p$,

$$X = F^0 \cup \bigcup_{n=1}^{\infty} \mathcal{G}^n(\iota_0),$$

and the fact that $\{\mathcal{G}^n(\iota_0)\}_{n \in \mathbb{N}}$ is an increasing sequence of sets. To show E) we start by assuming that $\Gamma^E \in \mathcal{G}^n(\tilde{\iota}_0)$ and then argue by contradiction. If $\Gamma^E \notin F^{n+1}$ and $\Gamma^E \notin F^{n+2}$ then $\Gamma^E(\tilde{\iota}_{n+1}) = \Xi^E(\tilde{\iota}_{n+1})$ and $\Gamma^E(\tilde{\iota}_{n+2}) = \Xi^E(\tilde{\iota}_{n+2})$. Since $\Gamma^E \in \mathcal{G}^n(\tilde{\iota}_0)$ we have $\Gamma^E(\tilde{\iota}_{n+1}) = \Gamma^E(\tilde{\iota}_{n+2})$ (this almost follows verbatim from the argument demonstrating (10.19) – the only differences here are that we use Cii) and, for the oracle case, the fact that $g_j(\tilde{\iota}_{n+1}) = v = g_j(\tilde{\iota}_{n+2})$, for all $j = 1, \ldots, N$ and $n \in \mathbb{N}$). Thus $\Xi^E(\tilde{\iota}_{n+1}) = \Xi^E(\tilde{\iota}_{n+2})$, which contradicts Ci). Finally, the proof of F) is verbatim from the proof of (III) in Claim 2 in (iii) in Proposition 10.5. $\square$

11. GEOMETRY OF SOLUTIONS TO PROBLEMS (1.1) - (1.4) – PART I

The purpose of this section is to provide some simple inputs and solutions of each of the problems listed in (1.1)- (1.4) that will be used to show that the breakdown epsilons for such computational problems can be non-zero. The intent will be to use these inputs and solutions to prove Theorems 3.4, 5.1, and 6.1. We will separately address counterexamples for ℓ^1 regularisation and TV regularisation as the examples we construct are somewhat different in these two settings. As the results of this section are simple consequences of elementary convex analysis, their proofs are deferred to Appendix A.

11.1. **Linear programming.** Our counterexample relies on a family of matrices $A(\alpha, \beta, m, N) \in \mathbb{R}^{m \times N}$ and a family of vectors $y^{\mathrm{A}}(y_1, m) \in \mathbb{R}^m$ for positive parameters α, β, y_1 where the dimensions m and N satisfy $m < N$ and $N \geq 3$. Additionally, where there is no ambiguity we write $A = A(\alpha, \beta, m, N)$ and $y^{\mathrm{A}} = y^{\mathrm{A}}(y_1, m)$ to simplify notation.

The families $A(\alpha, \beta, m, N)$ and $y^{\mathrm{A}}(y_1, m)$ are defined as follows:

$$
\begin{aligned}
A(\alpha, \beta, m, N) &= \begin{pmatrix} \alpha & \beta & -1 \end{pmatrix} \oplus \begin{pmatrix} I_{m-1} & 0_{m-1 \times N-m-2} \end{pmatrix} \\
y^{\mathrm{A}}(y_1, m) &= y_1 e_1.
\end{aligned} \tag{11.1}
$$

where y_1 is always assumed to be positive.

We now state a lemma that relates these inputs to the corresponding solutions of the LP problem.

Lemma 11.1. *Let $c = \mathbf{1}_N$ be the N-dimensional vector of ones. Then the solution Ξ_{LP} to the linear programming problem satisfies*

$$\Xi_{\mathrm{LP}}(y^{\mathrm{A}}, A) = \begin{cases} \left\{ \frac{y_1}{\alpha \vee \beta} e_1 \right\} & \text{if } \alpha > \beta \\ \left\{ \frac{y_1}{\alpha \vee \beta} e_2 \right\} & \text{if } \beta > \alpha \\ \left\{ \frac{y_1}{\alpha \vee \beta}(te_1 + (1-t)e_2) \,|\, t \in [0,1] \right\} & \text{if } \alpha = \beta \end{cases} \tag{11.2}$$

11.2. ℓ^1 regularisation. Here, our counterexamples rely on a family of matrices $L(\alpha, \beta, m, N) \in \mathbb{R}^{m \times N}$ and a family of vectors $y^{\mathrm{L}}(y_1, m) \in \mathbb{R}^m$ for positive parameters α, β, y_1 where the dimensions m and N satisfy $4 \leq m < N$. Additionally, where there is no ambiguity we write $L = L(\alpha, \beta, m, N)$ and $y^{\mathrm{L}} = y^{\mathrm{L}}(y_1, m)$. Specific to constrained lasso, we also introduce the family of vectors $y^{\mathrm{CL}}(y_1, m)$ that depend on the regularisation parameter τ and again if there is no ambiguity we write $y^{\mathrm{CL}} = y^{\mathrm{CL}}(y_1, m)$.

The families $L(\alpha, \beta, m, N)$, $y^{\mathrm{L}}(y_1, m)$ and $y^{\mathrm{CL}}(y_1, m)$ are defined as follows:

$$\begin{aligned} L(\alpha, \beta, m, N) &= \begin{pmatrix} \alpha & \beta \end{pmatrix} \oplus \begin{pmatrix} I_{m-1} & 0_{m-1 \times N-m-1} \end{pmatrix} \\ y^{\mathrm{L}}(y_1, m) &= y_1 e_1, \quad y^{\mathrm{CL}}(y_1, m) = y_1 e_1 + \tau e_2. \end{aligned} \tag{11.3}$$

where y_1 is always assumed to be positive. We now state some simple lemmas that relate these inputs to their solutions under unconstrained lasso, basis pursuit and constrained lasso.

Lemma 11.2. *Assuming that $\alpha \vee \beta \geq \lambda/(2y_1)$, the solution Ξ_{UL} to the unconstrained lasso problem satisfies*

$$\Xi_{\mathrm{UL}}(y^{\mathrm{L}}, L) = \begin{cases} \left\{ \frac{2\alpha y_1 - \lambda}{2\alpha^2} e_1 \right\} & \text{if } \alpha > \beta \\ \left\{ \frac{2\beta y_1 - \lambda}{2\beta^2} e_2 \right\} & \text{if } \alpha < \beta \\ \left\{ t \left(\frac{2\alpha y_1 - \lambda}{2\alpha^2} \right) e_1 + (1-t) \left(\frac{2\beta y_1 - \lambda}{2\beta^2} \right) e_2 \,|\, t \in [0,1] \right\} & \text{if } \alpha = \beta \end{cases} \tag{11.4}$$

Lemma 11.3. *Assuming that $y_1 \geq \delta$, we have*

$$\Xi_{\mathrm{BPDN}}(y^{\mathrm{L}}, L) = \begin{cases} \left\{ \frac{y_1 - \delta}{\alpha} e_1 \right\} & \text{if } \alpha > \beta \\ \left\{ \frac{y_1 - \delta}{\beta} e_2 \right\} & \text{if } \alpha < \beta \\ \left\{ t \frac{(y_1 - \delta)}{\alpha} e_1 + (1-t) \frac{(y_1 - \delta)}{\beta} e_2 \,|\, t \in [0,1] \right\} & \text{if } \alpha = \beta \end{cases} \tag{11.5}$$

Lemma 11.4. *Assuming that $y_1 \geq 0$ is so that $r = \frac{(\alpha \vee \beta)y_1}{1 + (\alpha \vee \beta)^2}$ satisfies $r \leq \tau$, the solution to the ℓ^1 constrained lasso problem Ξ_{CL} satisfies*

$$\Xi_{\mathrm{CL}}(y^{\mathrm{CL}}, L) = \begin{cases} \{ re_1 + (\tau - r)e_3 \} & \text{if } \alpha > \beta \\ \{ re_2 + (\tau - r)e_3 \} & \text{if } \alpha < \beta \\ \{ tre_1 + (1-t)re_2 + (\tau - r)e_3 \,|\, t \in [0,1] \} & \text{if } \alpha = \beta \end{cases} \tag{11.6}$$

11.3. TV regularisation. Our counterexamples rely on a family of matrices $T(\alpha, \beta, m, N) \in \mathbb{R}^{m \times N}$ and a family of vectors $y^{\mathrm{TV}}(y_1, m) \in \mathbb{R}^m$ for positive parameters α, β, y_1 where the dimensions m and N now satisfy $4 \leq m < N$. As before, where there is no ambiguity we write $T = T(\alpha, \beta, m, N)$ and $y^{\mathrm{TV}} = y(y_1, m)$.

The precise definitions of the families $T(\alpha, \beta, m, N)$ and $y^{\mathrm{TV}}(y_1, m)$ are as follows:

$$\begin{aligned} T &= \alpha\, e_1 \otimes e_1 + \beta\, e_1 \otimes e_N + e_m \otimes e_{N-1} + \sum_{r=2}^{m-1} e_r \otimes e_r \in \mathbb{R}^{m \times N}, \\ y^{\mathrm{TV}} &= y_1 e_1 \in \mathbb{R}^m, \ y_1 > 0. \end{aligned} \tag{11.7}$$

To simplify the results that follow, we also define the value $\theta = \theta(\alpha, \beta, m)$ by

$$\theta(\alpha, \beta, m) = \left((m-1)^2 + (\alpha + \beta)^2 (m-1)\right)^{\frac{1}{2}}. \tag{11.8}$$

We now present some important lemmas relating these inputs to the corresponding solutions of problems (1.2) to (1.4) under TV regularisation. In the following, we use the flipping operator $P_{\text{flip}} : \mathbb{R}^N \to \mathbb{R}^N$, defined as $(P_{\text{flip}} v)_i = v_{N-i+1}$.

Lemma 11.5. *Assume that $y_1 > \delta\theta/(m-1)$. Then*

$$\Xi_{\text{BPTV}}(y^{\text{TV}}, T) = \begin{cases} \{\eta(y_1, \alpha, \beta)\} & \text{if } \alpha < \beta \\ \{P_{\text{flip}}\eta(y_1, \alpha, \beta)\} & \text{if } \alpha > \beta , \\ \{t\eta(y_1, \alpha, \beta) + (1-t)(P_{\text{flip}}\eta(y_1, \alpha, \beta)) \mid t \in [0,1]\} & \text{if } \alpha = \beta \end{cases} \tag{11.9}$$

where

$$\eta(y_1, \alpha, \beta)_1 = \eta(y_1, \alpha, \beta)_2 = \cdots = \eta(y_1, \alpha, \beta)_{N-1} = \frac{\delta(\alpha + \beta)}{\theta},$$
$$\eta(y_1, \alpha, \beta)_N = \frac{\delta(\alpha + \beta)}{\theta} + \frac{y_1 - \delta\theta/(m-1)}{\alpha \vee \beta} \tag{11.10}$$

Lemma 11.6. *Let T and y^{TV} be as above, let $y_1 > \frac{\lambda\theta^2}{2(m-1)^2(\alpha\vee\beta)}$. Let Ξ_{ULTV} denote the solution map for TV Unconstrained Lasso with parameter λ. Then*

$$\Xi_{\text{ULTV}}(y^{\text{TV}}, T) = \begin{cases} \{\psi(y_1, \alpha, \beta)\} & \text{if } \alpha < \beta \\ \{(P_{\text{flip}}\psi(y_1, \alpha, \beta))\} & \text{if } \alpha > \beta , \\ \{[t\psi(y_1, \alpha, \beta) + (1-t)(P_{\text{flip}}\psi(y_1, \alpha, \beta)) \mid t \in [0,1]\} & \text{if } \alpha = \beta \end{cases} \tag{11.11}$$

where

$$\psi(y_1, \alpha, \beta)_1 = \psi(y_1, \alpha, \beta)_2 = \cdots = \psi(y_1, \alpha, \beta)_{N-1} = \frac{\lambda(\alpha + \beta)}{2(m-1)(\alpha \vee \beta)},$$
$$\psi(y_1, \alpha, \beta)_N = \frac{\lambda(\alpha + \beta)}{2(m-1)(\alpha \vee \beta)} + \frac{1}{\alpha \vee \beta}\left(y_1 - \frac{\lambda\theta^2}{2(m-1)^2(\alpha \vee \beta)}\right). \tag{11.12}$$

11.4. **Linear programming and basis pursuit examples for the exit flag theorem.** For $\alpha \geq 0$ let

$$A^{\text{E}}(\alpha, m, N) = \alpha \oplus \left(I_{m-1} \quad 0_{m-1 \times N-m-1}\right)$$
$$y^{\text{L}}(y_1, m) = y_1 e_1. \tag{11.13}$$

where y_1 is always assumed to be non-negative (but not necessarily non-zero). Additionally, assume that c is the N-dimensional vector of 1s. We then have the following:

Lemma 11.7. *For basis pursuit and linear programming we have*

$$\Xi_{\text{LP}}(y^{\text{L}}, A^{\text{E}}) = \Xi_{\text{BP}}(y^{\text{L}}, A^{\text{E}}) = \begin{cases} (y_1/\alpha)e_1 & \text{if } \alpha > 0 \\ 0 & \text{if } \alpha = y_1 = 0 \end{cases}.$$

11.5. **Objective values for linear programming and Smale's 9th problem.** Our counterexamples used when proving Theorem 6.1 rely on the matrix and vector pairs defined as follows for $\alpha, \beta \geq 0$:

$$A^{\text{LP,D}}(\alpha, \beta, m, N) = \left(\alpha \quad -\beta\right) \oplus \left(I_{m-1} \quad 0_{m-1 \times N-m-1}\right),$$
$$y^{\text{L}}(y_1, m) = y_1 e_1. \tag{11.14}$$

where y_1 is always assumed to be non-negative. We then have the following lemma:

Lemma 11.8. *For $k \in \mathbb{N}$ and real $M \geq 0$, consider the decision problem (9.26) with $c = \mathbf{1}_N$ (for each fixed dimension $N \in \mathbb{N}$) and the corresponding solution map Ξ_k as defined in (9.27). Then, for $y_1 > 0$, $\alpha > 0$, $\beta \geq 0$, we have*

$$\Xi_k(y^{\mathrm{L}}(y_1, m), A^{\mathrm{LP,D}}(\alpha, \beta, m, N)) = \begin{cases} 1 & \text{if } y_1/\alpha < 10^{-k}(\lfloor 10^k M \rfloor + 1) \\ 0 & \text{if } y_1/\alpha \geq 10^{-k}(\lfloor 10^k M \rfloor + 1) \end{cases}, \tag{11.15}$$

where Ξ_k is defined in (9.27). In addition, for all $\alpha, \beta \in \mathbb{R}$, we have $\Xi_k(y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(\alpha, \beta, m, N)) = 1$.

12. Proof of Theorem 3.4 – Preliminaries: constructing Ω

12.1. **Strategy for the proof.** The proof of Theorem 3.4 formally consists of the proofs of Proposition 9.33 (corresponding to parts (i) and (ii) of the theorem) and Proposition 9.32 (corresponding to parts (iii) and (iv) of the theorem), which will be proved in §14 and §15, respectively.

The strategy for these proofs is as follows. First, for each of problems (1.1) - (1.4), we need to construct a suitable input set (which will depend both on K and any relevant regularisation parameters) of the desired form, i.e., a set $\Omega = \bigcup_{m,N} \Omega_{m,N}$, where each $\Omega_{m,N}$ consists of inputs of fixed dimension. The computational problem corresponding to each $\Omega_{m,N}$ (for fixed m and N) will have a strong breakdown epsilon exceeding 10^{-K} as well as a weak breakdown epsilon exceeding 10^{-K+1}. This will be achieved by setting $\Omega_{m,N} = \Omega_{m,N}^{\mathrm{s}} \cup \Omega_{m,N}^{\mathrm{w}}$, where $\Omega_{m,N}^{\mathrm{s}}$ is a set of input for which the computational problem $\{\Xi, \Omega_{m,N}^{\mathrm{s}}, \mathcal{M}, \hat{\Lambda}\}$ has a strong breakdown epsilon of size exceeding 10^{-K} and $\Omega_{m,N}^{\mathrm{w}}$ is a set of input for which $\{\Xi, \Omega_{m,N}^{\mathrm{w}}, \mathcal{M}, \hat{\Lambda}\}$ has a weak breakdown epsilon exceeding 10^{-K+1}, where the evaluation set $\hat{\Lambda}$ will be provided by Proposition 10.5. The whole construction will make heavy use of the results presented in §11. Specifically, our inputs will take the form of matrix vector pairs (y^{L}, A), (y^{L}, L) and (y^{TV}, T) presented in that section.

The algorithms whose existence is claimed in the statements of Propositions 9.32 and 9.33 will be constructed with the help of various subroutines introduced in §13.

12.2. **Constructing the sets of inputs for Theorem 3.4.**

12.2.1. *The set of inputs.* Each individual problem will require a separate input set. Moreover, they will depend on the integer K from the statement of Theorem 3.4 as well as any relevant regularisation parameters. We will denote the input sets for LP, ℓ^1 BP, ℓ^1 UL, CL, TV BP and TV UL by

$$\Omega^{\mathrm{LP}}, \Omega^{\mathrm{BP},\ell^1}, \Omega^{\mathrm{UL},\ell^1}, \Omega^{\mathrm{CL},\ell^1}, \Omega^{\mathrm{BP,TV}} \text{ and } \Omega^{\mathrm{UL,TV}}, \tag{12.1}$$

respectively. We remark that these will depend on K and the regularisation parameters δ, λ, and τ, however, in order to lighten the notation, we omit making this dependence explicit. For linear programming and the ℓ^1 regularised problems, the set $\mathcal{L} \subset \mathbb{R}^2$ defined by $\mathcal{L} = ([1/4, 1/2] \times \{1/2\}) \cup (\{1/2\} \times [1/4, 1/2])$ will prove useful whereas for the TV problems we denote, for integers n (and for given basis pursuit denoising parameter δ and unconstrained lasso parameter λ)

$$r_n = \begin{cases} \frac{1}{4} \vee \frac{1}{2}\sqrt{1 - \frac{3 \cdot 10^{-n}}{4\delta}} & \text{if } 3 \cdot 10^{-n} \leq 4\delta \\ \frac{1}{4} & \text{otherwise} \end{cases}, \quad s_n = \begin{cases} \frac{1}{4} \vee \frac{1}{2}\sqrt{1 - \frac{3 \cdot 10^{-n}}{4\lambda}} & \text{if } 3 \cdot 10^{-n} \leq 4\lambda \\ \frac{1}{4} & \text{otherwise} \end{cases} \tag{12.2}$$

and define $\mathcal{L}^{\mathrm{BP,TV},n} = ([r_n, 1/2] \times \{1/2\}) \cup (\{1/2\} \times [r_n, 1/2])$ and $\mathcal{L}^{\mathrm{UL,TV},n} = ([s_n, 1/2] \times \{1/2\}) \cup (\{1/2\} \times [s_n, 1/2])$. Recall the definitions of $A(\alpha, \beta, m, N)$ and $y^{\mathrm{A}}(y_1, m)$ from (11.1), $L(\alpha, \beta, m, N)$

and $y^{\mathrm{L}}(y_1, m)$ from (11.3), as well as $T(\alpha, \beta, m, N)$ and $y^{\mathrm{TV}}(y_1, m)$ from (11.7), and, for $k \geq 1$, define

$$
\begin{aligned}
\Omega_{m,N,k}^{\mathrm{LP},\mathrm{s}} &:= \left\{ \left(y^{\mathrm{LP},k}(m), A(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \right\}, \\
\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}} &:= \left\{ \left(y^{\mathrm{BP},\ell^1,k}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \right\}, \\
\Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}} &:= \left\{ \left(y^{\mathrm{UL},\ell^1,k}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \right\}, \\
\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}} &:= \left\{ \left(y^{\mathrm{CL},k}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \right\}, \\
\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{s}} &:= \left\{ \left(y^{\mathrm{BP},\mathrm{TV},k}(m), T(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{BP},\mathrm{TV},k} \right\}, \\
\Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV},\mathrm{s}} &:= \left\{ \left(y^{\mathrm{UL},\mathrm{TV},k}(m), T(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{UL},\mathrm{TV},k} \right\}
\end{aligned}
\tag{12.3}
$$

where the superscript s notes that the input sets are designed for results concerning the strong breakdown epsilon, and

$$
\begin{aligned}
y^{\mathrm{LP},k}(m) &:= y^{\mathrm{A}}(2 \cdot 10^{-k}, m), \qquad y^{\mathrm{BP},\ell^1,k}(m) = y^{\mathrm{L}}(2 \cdot 10^{-K} + \delta, m), \\
y^{\mathrm{UL},\ell^1,k}(m) &:= y^{\mathrm{L}}\left(2 \cdot 10^{-k} + \lambda, m\right), \qquad y^{\mathrm{CL},k}(m) := y^{\mathrm{CL}}\left(10^{-k+1}, m\right), \\
y^{\mathrm{BP},\mathrm{TV},k}(m) &:= y^{\mathrm{TV}}\left(\frac{\delta m}{\theta(1/2, 1/2, m)} + \left[7 - \frac{3}{\theta(1/2, 1/2, m)} \right] \frac{10^{-k}}{4}, m \right), \\
y^{\mathrm{UL},\mathrm{TV},k}(m) &:= y^{\mathrm{TV}}\left(\frac{7 \cdot 10^{-k}}{4} + \lambda \left[1 + \frac{1}{m-1} \right] - \frac{3 \cdot 10^{-k}}{4(m-1)}, m \right),
\end{aligned}
\tag{12.4}
$$

where $\lambda \in (0, 1/3]$ is the regularisation parameter from the UL problem (1.3) and $\delta \in [0, 1]$ is the regularisation parameter from the BP problem (1.2). Where it is either clear or superfluous to the result proven, we will omit the dependency on m in the definition of the y vectors.

To create input sets that capture the required weak breakdown epsilons, we use the superscript w and, for $k \geq 2$, define

$$
\begin{aligned}
\Omega_{m,N,k}^{\mathrm{LP},\mathrm{w}} &= \left\{ \left(y^{\mathrm{LP},k-1}(m), A(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \setminus \{z^a\} \right\} \\
\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{w}} &= \left\{ \left(y^{\mathrm{BP},\ell^1,k-1}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \setminus \{z^a\} \right\} \\
\Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{w}} &= \left\{ \left(y^{\mathrm{UL},\ell^1,k-1}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \setminus \{z^a\} \right\} \\
\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{w}} &= \left\{ \left(y^{\mathrm{CL},k-1}(m), L(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L} \setminus \{z^a\} \right\}, \\
\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{w}} &= \left\{ \left(y^{\mathrm{BP},\mathrm{TV},k-1}(m), T(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{BP},\mathrm{TV},k-1} \setminus \{z^a\} \right\} \\
\Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV},\mathrm{w}} &= \left\{ \left(y^{\mathrm{UL},\mathrm{TV},k-1}(m), T(\alpha, \beta, m, N) \right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{UL},\mathrm{TV},k-1} \setminus \{z^a\} \right\},
\end{aligned}
\tag{12.5}
$$

where z^a is the corner point $(1/2, 1/2)$ of $\mathcal{L}$, whereas for $k = 1$ we let each of $\Omega_{m,N,k}^{\mathrm{LP},\mathrm{w}}$, $\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{w}}$, $\Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{w}}$, $\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{w}}$, $\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{w}}$, and $\Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV},\mathrm{w}}$ be the empty set. We now define the fixed-dimension input sets as the union of the corresponding "strong" and "weak" sets:

$$
\begin{aligned}
\Omega_{m,N,k}^{\mathrm{LP}} &= \Omega_{m,N,k}^{\mathrm{LP},\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{LP},\mathrm{w}}, \quad &\Omega_{m,N,k}^{\mathrm{CL},\ell^1} &= \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{w}}, \\
\Omega_{m,N,k}^{\mathrm{BP},\ell^1} &= \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{w}}, \quad &\Omega_{m,N,k}^{\mathrm{UL},\ell^1} &= \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{w}}, \\
\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV}} &= \Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{w}}, \quad &\Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV}} &= \Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV},\mathrm{s}} \cup \Omega_{m,N,k}^{\mathrm{UL},\mathrm{TV},\mathrm{w}},
\end{aligned}
\tag{12.6}
$$

for each $k \geq 1$, $m \geq 4$, and $N > m$, and the combined input sets as the union of the fixed-dimension input sets with the given value of K over all admissible dimensions:

$$
\Omega^{\mathrm{LP}} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{LP}}_{m,N,K}, \quad \Omega^{\mathrm{CL},\ell^1} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{CL},\ell^1}_{m,N,K}, \quad \Omega^{\mathrm{BP},\ell^1} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{BP},\ell^1}_{m,N,K},
$$

$$
\Omega^{\mathrm{UL},\ell^1} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{UL},\ell^1}_{m,N,K}, \quad \Omega^{\mathrm{BP},\mathrm{TV}} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{BP},\mathrm{TV}}_{m,N,k}, \quad \Omega^{\mathrm{UL},\mathrm{TV}} = \bigcup_{\substack{m,N \in \mathbb{N} \\ N > m \geq 4}} \Omega^{\mathrm{UL},\mathrm{TV}}_{m,N,K}
$$

$$(12.7)$$

12.3. **Size and conditioning of the inputs.** In this section we will analyse the bounds on the inputs as well as each of the condition numbers relevant to the input sets defined in §12.2.

12.3.1. *Size of the inputs.*

Lemma 12.1. *For natural numbers m and N, let $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times N}$. Then for an integer $k \geq 1$ the following hold:*

(1) *If (y, A) is an element of one of the sets $\Omega^{\mathrm{LP},\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{CL},\ell^1,\mathrm{s}}_{m,N,k}$ then $\|y\|_\infty \leq 2$ and $\|A\|_{\max} \leq 1$.*

(2) *If (y, A) is an element of $\Omega^{\mathrm{BP},\mathrm{TV},\mathrm{s}}_{m,N,k}$ then $\|y\|_\infty \leq 3/2$ and $\|A\|_{\max} = 1$.*

(3) *If (y, A) is an element of $\Omega^{\mathrm{UL},\mathrm{TV},\mathrm{s}}_{m,N,k}$ then $\|y\|_\infty \leq 107/180$ and $\|A\|_{\max} = 1$.*

Similarly, for an integer $k \geq 2$ the following statements hold:

(4) *If (y, A) is an element of one of the sets $\Omega^{\mathrm{LP},\mathrm{w}}_{m,N,k}, \Omega^{\mathrm{BP},\ell^1,\mathrm{w}}_{m,N,k}, \Omega^{\mathrm{UL},\ell^1,\mathrm{w}}_{m,N,k}, \Omega^{\mathrm{CL},\ell^1,\mathrm{w}}_{m,N,k}$ then $\|y\|_\infty \leq 2$ and $\|A\|_{\max} \leq 1$.*

(5) *If $\Omega = \Omega^{\mathrm{BP},\mathrm{TV},\mathrm{w}}_{m,N,k}$ then $\|y\|_\infty \leq 3/2$ and $\|A\|_{\max} = 1$.*

(6) *If $\Omega = \Omega^{\mathrm{UL},\mathrm{TV},\mathrm{w}}_{m,N,k}$ then $\|y\|_\infty \leq 107/180$ and $\|A\|_{\max} = 1$.*

Proof. We start with the case that (y, A) is in one of $\Omega^{\mathrm{LP},\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,k}, \Omega^{\mathrm{CL},\ell^1,\mathrm{s}}_{m,N,k}$. In this case, we must have exactly one of $A = A(\alpha, \beta, m, N)$ or $A = L(\alpha, \beta, m, N)$. In any of these cases it is easy to see that $\|A\|_{\max} = 1$ directly from the definitions. We also have $\|y\|_\infty \leq \max(2 \cdot 10^{-k}, 2 \cdot 10^{-k} + \delta, 2 \cdot 10^{-k} + \lambda, 10^{-k+1}, \tau) \leq 2$, as each of these terms is at most 2.

Next, we analyse $(y, A) \in \Omega^{\mathrm{BP},\mathrm{TV},\mathrm{s}}_{m,N,k}$: if $(y, A) \in \Omega^{\mathrm{BP},\mathrm{TV},\mathrm{s}}_{m,N,k}$ then (writing $\theta = \theta(1/2, 1/2, m)$ and by the definition of $y^{\mathrm{BP},\mathrm{TV},k}$)

$$
\|y\|_\infty = \frac{\delta m}{\theta} + \left(7 - \frac{3}{\theta}\right) \frac{10^{-k}}{4} \leq \frac{1}{m-1}\left(m - \frac{3 \cdot 10^{-k}}{4}\right) + \frac{7 \cdot 10^{-k}}{4}
$$

$$
= 1 + \frac{4 - 3 \cdot 10^{-k}}{4(m-1)} + \frac{7 \cdot 10^{-k}}{4} \leq \frac{4}{3} + \frac{3 \cdot 10^{-k}}{2}
$$

where the first inequality follows because $\delta \leq 1$ and $\theta \geq m - 1$ and the final inequality because $m \geq 4$. Since $k \geq 1$ we obtain $\|y\|_\infty < 3/2$. By the definition of $\Omega^{\mathrm{BP},\mathrm{TV},\mathrm{s}}_{m,N,k}$, we have $A = T(\alpha, \beta, m, N)$ for some $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP},\mathrm{TV},k}$. Thus from the definition of T and the fact that $\alpha, \beta \leq 1/2$ we conclude that $\|A\|_{\max} \leq 1$. Thus we have shown 2.

To analyse the case where $(y, A) \in \Omega^{\mathrm{UL},\mathrm{TV},\mathrm{s}}_{m,N,k}$, note that if $(y, A) \in \Omega^{\mathrm{UL},\mathrm{TV},\mathrm{s}}_{m,N,k}$ then

$$
\|y\|_\infty = \frac{7 \cdot 10^{-k}}{4} + \lambda\left(1 + \frac{1}{m-1}\right) - \frac{3 \cdot 10^{-k}}{4(m-1)}
$$

$$
\leq \frac{7 \cdot 10^{-k}}{4} + \frac{1}{3} + \left(\frac{1}{3} - \frac{3 \cdot 10^{-k}}{4}\right)\frac{1}{m-1} \leq \frac{3 \cdot 10^{-k}}{2} + \frac{4}{9}
$$

where the first inequality follows because $\lambda \leq 1/3$ and the final inequality because $m \geq 4$. Thus for $k \geq 1$ we obtain $\|y\|_\infty \leq 4/9 + 3/20 = 107/180$. The argument that $\|A\|_{\max} \leq 1$ is identical to

the analysis performed to prove the bounds on $\Omega^{\mathrm{BP,TV}}$, except now we replace all statements and sets referring to basis pursuit with those referring to unconstrained lasso. This proves 3.

Finally, we can prove 4 to 6 by using 1 to 3 and noting the inclusions

$$\Omega^{\mathrm{LP,w}}_{m,N,k} \subseteq \Omega^{\mathrm{LP,s}}_{m,N,k-1}, \quad \Omega^{\mathrm{BP},\ell^1,\mathrm{w}}_{m,N,k} \subseteq \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,k-1}, \quad \Omega^{\mathrm{UL},\ell^1,\mathrm{w}}_{m,N,k} \subseteq \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,k-1}, \quad \Omega^{\mathrm{CL},\ell^1,\mathrm{w}}_{m,N,k} \subseteq \Omega^{\mathrm{CL},\ell^1,\mathrm{s}}_{m,N,k-1}$$

$$\Omega^{\mathrm{BP,TV,w}}_{m,N,k} \subseteq \Omega^{\mathrm{BP,TV,s}}_{m,N,k-1}, \quad \Omega^{\mathrm{UL,TV,w}}_{m,N,k} \subseteq \Omega^{\mathrm{UL,TV,s}}_{m,N,k-1}$$

that hold whenever $k \geq 2$. $\qquad\square$

12.3.2. *Condition of the solution map.* We compute the condition of the solution map for the problems LP, ℓ^1 BP, ℓ^1 UL, TV BP, and TV UL, with the input sets as specified in §12.2. Concretely, we prove the following lemma.

Lemma 12.2. *We have*

$$\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega^{\mathrm{LP}}) \leq 15, \quad \mathrm{cond}(\Xi_{\mathrm{UL}}, \Omega^{\mathrm{UL},\ell^1}) \leq 28, \quad \mathrm{cond}(\Xi_{\mathrm{CL}}, \Omega^{\mathrm{CL},\ell^1}) \leq 2,$$

$$\mathrm{cond}(\Xi_{\mathrm{BP}}, \Omega^{\mathrm{BP},\ell^1}) \leq 24, \quad \mathrm{cond}(\Xi_{\mathrm{BPTV}}, \Omega^{\mathrm{BP,TV}}) \leq 35, \quad \mathrm{cond}(\Xi_{\mathrm{ULTV}}, \Omega^{\mathrm{UL,TV}}) \leq 179, \quad (12.8)$$

where Ω^{LP}, $\Omega^{\mathrm{BP},\ell^1}$, $\Omega^{\mathrm{UL},\ell^1}$, $\Omega^{\mathrm{BP,TV}}$, $\Omega^{\mathrm{CL},\ell^1}$, and $\Omega^{\mathrm{UL,TV}}$ are defined in (12.7).

This will give us the desired bounds on the condition numbers. Proving Lemma 12.2 will be straightforward for linear programming and the ℓ^1 regularised problems but will require a bit more effort for the TV problems. We thus state and prove three simple lemmas that will be useful.

Lemma 12.3. *Let $\alpha, \beta > 0$, and let $z = (z_\alpha, z_\beta) \in \mathbb{R}^2$. Then*

$$|(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta| \leq \|z\|_\infty.$$

Proof. Suppose w.l.o.g. that $\alpha \geq \beta$. Then $\alpha - \|z\|_\infty \leq \alpha + z_\alpha \leq (\alpha + z_\alpha) \vee (\beta + z_\beta) \leq \alpha + \|z\|_\infty$, and so $|(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha| \leq \|z\|_\infty$, as desired. $\qquad\square$

Lemma 12.4. *Let $(\alpha, \beta) \in \mathcal{L}^{BP,TV,n}$ for some natural number n and let m, N, $\theta = \theta(\cdot, \cdot)$, and $\eta = \eta(\cdot, \cdot, \cdot)$ be as defined in* (11.8) *and* (11.10). *Let $y_1 > 0$ be such that $\mu(y_1, \alpha, \beta) := y_1 - \delta\theta(\alpha, \beta, m)/(m-1) > 0$ and $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ with $\|z\|_\infty \leq \frac{1}{8} \wedge \frac{1}{2}\mu(y_1, \alpha, \beta)$. Then $\mu((y_1, \alpha, \beta) + z) > 0$ and*

$$\|\eta(y_1, \alpha, \beta) - \eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta)\|_\infty \leq 14(y_1 + 1)\|z\|_\infty.$$

Proof of Lemma 12.4. First note that, using the mean value theorem and the fact that $\alpha + z_\alpha, \beta + z_\beta \in [1/8, 5/8]$, we have

$$|\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)| \leq \max_{(u,v) \in [1/8,5/8]} \|\nabla_{u,v}\theta(u, v, m)\|_1 \|z\|_\infty$$

$$\leq \max_{(u,v) \in [1/8,5/8]} \left|\frac{2(u+v)(m-1)}{\theta(u, v, m)}\right| \|z\|_\infty \leq \max_{(u,v) \in [3/8,5/8]} 2(u+v)\|z\|_\infty \leq \frac{5}{2}\|z\|_\infty. \tag{12.9}$$

since $(u+v)(m-1)/\theta(u, v, m) = [1/(u+v)^2 + 1/(m-1)]^{-1/2} \leq (u+v)$. We can now write

$$\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) = \frac{\delta \cdot (\alpha + z_\alpha + \beta + z_\beta)}{\theta(\alpha + z_\alpha, \beta + z_\beta, m)}\mathbf{1}_N + \frac{\mu((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)}e_N,$$

where $\mathbf{1}_N \in \mathbb{R}^N$ is the vector of all ones, so that

$$\|\eta(y_1, \alpha, \beta) - \eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta)\|_\infty$$

$$\leq \left\|\frac{\delta \cdot (\alpha + z_\alpha + \beta + z_\beta)}{\theta(\alpha + z_\alpha, \beta + z_\beta, m)}\mathbf{1}_N - \frac{\delta \cdot (\alpha + \beta)}{\theta(\alpha, \beta, m)}\mathbf{1}_N\right\|_\infty + \left\|\frac{\mu((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)}e_N - \frac{\mu(y_1, \alpha, \beta)}{\alpha \vee \beta}e_N\right\|_\infty$$

$$= \left|\frac{\delta \cdot (\alpha + z_\alpha + \beta + z_\beta)}{\theta(\alpha + z_\alpha, \beta + z_\beta, m)} - \frac{\delta \cdot (\alpha + \beta)}{\theta(\alpha, \beta, m)}\right| + \left|\frac{\mu((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\mu(y_1, \alpha, \beta)}{\alpha \vee \beta}\right|.$$

$$(12.10)$$

We will bound each of the latter two terms separately. Write $f(u,v) = u/v$, for v non-zero. Note that, since $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV},n}$, we have $\alpha + z_\alpha + \beta + z_\beta \in (1/2, 5/4)$: indeed, $1/2 = 3/4 - 2/8 \leq \alpha + \beta - 2\|z\|_\infty \leq \alpha + z_\alpha + \beta + z_\beta$ and similarly $\alpha + z_\alpha + \beta + z_\beta \leq \alpha + \beta + 2\|z\|_\infty \leq 1 + 2/8 = 5/4$. In a similar way, $m - 1 \leq \sqrt{(m-1)^2 + (m-1)(\alpha + z_\alpha + \beta + z_\beta)} \leq \sqrt{(m-1)^2 + 5(m-1)/4} \leq m$ so that $\theta(\alpha + z_\alpha, \beta + z_\beta, m) \in [m-1, m]$. Thus, by the mean value theorem

$$
\left| \frac{\delta \cdot (\alpha + z_\alpha + \beta + z_\beta)}{\theta(\alpha + z_\alpha, \beta + z_\beta, m)} - \frac{\delta \cdot (\alpha + \beta)}{\theta(\alpha, \beta, m)} \right|
$$
$$
\leq \delta \max_{(u,v) \in [1/2, 5/4] \times [m-1, m]} \|\nabla f(u,v)\|_1 \cdot (|z_\alpha + z_\beta| \vee |\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)|)
$$
$$
\leq \delta \left(\frac{1}{m-1} + \frac{5/4}{(m-1)^2} \right) \cdot (2\|z\|_\infty \vee |\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)|) \leq \frac{85\|z\|_\infty}{72},
$$
$$(12.11)$$

where in the final line we used (12.9), $\delta \leq 1$, and the assumption that $m \geq 4$. Next, again using (12.9), we obtain $|\mu((y_1, \alpha, \beta) + z) - \mu(y_1, \alpha, \beta)| \leq |z_y| + \frac{\delta}{m-1} |\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)| \leq \frac{11}{6}\|z\|_\infty$ since $\delta \in [0,1]$ and $m \geq 4$. In particular, since $\mu(y_1, \alpha, \beta) \geq 2\|z\|_\infty$ and $\mu(y_1, \alpha, \beta) \leq y_1$ we obtain $\mu((y_1, \alpha, \beta) + z) \in [\|z\|_\infty/6, y_1 + 11\|z\|_\infty/6] \subset [0, y_1 + 1/4]$, establishing the claim that $\mu((y_1, \alpha, \beta) + z) > 0$. Now, by Lemma 12.3, $(\alpha + z_\alpha) \vee (\beta + z_\beta) \in [3/8, 5/8]$. Therefore, using the mean value theorem together with the bounds above we obtain

$$
\left| \frac{\mu((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\mu(y_1, \alpha, \beta)}{\alpha \vee \beta} \right| \leq \max_{(u,v) \in [0, y_1 + \frac{1}{4}] \times [\frac{3}{8}, \frac{5}{8}]} \|\nabla f(u,v)\|_1
$$
$$
\cdot (|\mu((y_1, \alpha, \beta) + z) - \mu(y_1, \alpha, \beta)| \vee |(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta|) \qquad (12.12)
$$
$$
\leq \left(\frac{1}{3/8} + \frac{(y_1 + \frac{1}{4})}{(3/8)^2} \right) \cdot \left(\frac{11}{6}\|z\|_\infty \vee \|z\|_\infty \right) = \left(\frac{220}{27} + \frac{352 y_1}{27} \right) \|z\|_\infty.
$$

Combining (12.10), (12.11), (12.12) as well as the assumption $\delta \leq 1$ yields $\|\eta(y_1, \alpha, \beta) - \eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta)\|_\infty \leq 14(y_1 + 1)$, as desired. $\qquad\square$

Lemma 12.5. *For some natural number n, let $(\alpha, \beta) \in \mathcal{L}^{\mathrm{UL,TV},n}$ and let m, N, $\theta = \theta(\cdot, \cdot)$, and $\psi = \psi(\cdot, \cdot, \cdot)$ be as in subsection 11.3. Suppose $y_1 > 0$ is such that*

$$
\varsigma(y_1, \alpha, \beta) := y_1 - \lambda \, \theta(\alpha, \beta, m)^2 \left[2(m-1)^2 (\alpha \vee \beta) \right]^{-1} > 0
$$

and $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^2$ with $\|z\|_\infty \leq \frac{1}{8} \wedge \frac{1}{10}\varsigma(y_1, \alpha, \beta)$. Then $\varsigma((y_1, \alpha, \beta) + z) > 0$ and

$$
\|\psi(y_1, \alpha, \beta) - \psi(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta)\|_\infty \leq 112(y_1 + 1)\|z\|_\infty. \qquad (12.13)
$$

Proof of Lemma 12.5. As in the proof of Lemma 12.4, we first obtain $|\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)| \leq \frac{5}{2}\|z\|_\infty$ and $\theta(\alpha + z_\alpha, \beta + z_\beta, m) \in [m-1, m]$. Moreover, using Lemma 12.3 we get $(\alpha + z_\alpha) \vee (\beta + z_\beta) \in [3/8, 5/8]$. Next, we write

$$
\psi(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) = \frac{\lambda \cdot (\alpha + z_\alpha + \beta + z_\beta)}{2(m-1)((\alpha + z_\alpha) \vee (\beta + z_\beta))} \mathbf{1} + \frac{\varsigma((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} e_N
$$

where $\mathbf{1}_N \in \mathbb{R}^N$ is the vector of all ones, so that

$$
\|\psi(y_1, \alpha, \beta) - \psi(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta)\|_\infty
$$
$$
\leq \left\| \frac{\lambda \cdot (\alpha + z_\alpha + \beta + z_\beta)}{2(m-1)((\alpha + z_\alpha) \vee (\beta + z_\beta))} \mathbf{1} - \frac{\lambda \cdot (\alpha + \beta)}{2(m-1)(\alpha \vee \beta)} \mathbf{1} \right\|_\infty +
$$
$$
+ \left\| \frac{\varsigma((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} e_N - \frac{\varsigma(y_1, \alpha, \beta)}{\alpha \vee \beta} e_N \right\|_\infty
$$
$$
= \left| \frac{\lambda \cdot (\alpha + z_\alpha + \beta + z_\beta)}{2(m-1)((\alpha + z_\alpha) \vee (\beta + z_\beta))} - \frac{\lambda \cdot (\alpha + \beta)}{2(m-1)(\alpha \vee \beta)} \right| + \left| \frac{\varsigma((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\varsigma(y_1, \alpha, \beta)}{\alpha \vee \beta} \right|.
$$
$$(12.14)$$

We will bound both of these terms separately. The first term can be bounded as follows. Let $f(u,v) = u/v$ and note that $\alpha + \beta + z_\alpha + z_\beta \in [1/2 + 1/4 - 2/8, 1/2 + 1/2 + 2/8] = [1/2, 5/4]$. We thus obtain

$$\left| \frac{\lambda \cdot (\alpha + z_\alpha + \beta + z_\beta)}{2(m-1)((\alpha + z_\alpha) \vee (\beta + z_\beta))} - \frac{\lambda \cdot (\alpha + \beta)}{2(m-1)(\alpha \vee \beta)} \right|$$

$$\leq \max_{(u,v) \in [\frac{1}{2}, \frac{5}{4}] \times [\frac{3}{8}, \frac{5}{8}]} \frac{\lambda}{2(m-1)} \|\nabla f(u,v)\|_1 \cdot (|z_\alpha + z_\beta| \vee |(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta|) \quad (12.15)$$

$$\leq \frac{\lambda}{2(m-1)} \left(\frac{1}{3/8} + \frac{5/4}{(3/8)^2} \right) \cdot (2\|z\|_\infty \vee \|z\|_\infty) \leq \frac{12\lambda\|z\|_\infty}{m-1} \leq \frac{4\|z\|_\infty}{3}$$

since $\lambda \leq 1/3$ and $m \geq 4$.

For the second term, using the mean value theorem with the function $g(u,v) = u^2/v$, we first obtain the following bound

$$\left| \frac{\theta(\alpha + z_\alpha, \beta + z_\beta, m)^2}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\theta(\alpha, \beta, m)^2}{\alpha \vee \beta} \right|$$

$$\leq \max_{\substack{u \in [m-1, m] \\ v \in [\frac{3}{8}, \frac{5}{8}]}} \|\nabla g(u,v)\|_1 \cdot [|\theta(\alpha + z_\alpha, \beta + z_\beta, m) - \theta(\alpha, \beta, m)| \vee |(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta|]$$

$$\leq \left(\frac{2m}{3/8} + \frac{m^2}{(3/8)^2} \right) \cdot \left(\frac{5}{2} \|z\|_\infty \vee \|z\|_\infty \right) \leq 18(1 + m)^2 \|z\|_\infty$$

and therefore, since $\lambda \leq 1/3$ and $m \geq 4$, we find

$$\frac{\lambda}{2(m-1)^2} \left| \frac{\theta(\alpha + z_\alpha, \beta + z_\beta, m)^2}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\theta(\alpha, \beta, m)^2}{\alpha \vee \beta} \right| \leq 9\lambda\|z\|_\infty \left(1 + \frac{2}{m-1} \right)^2 \leq \frac{25\|z\|_\infty}{3}.$$

Therefore,

$$|\varsigma((y_1, \alpha, \beta) + z) - \varsigma(y_1, \alpha, \beta)| \leq |z_y| + \frac{\lambda}{2(m-1)^2} \left| \frac{\theta(\alpha + z_\alpha, \beta + z_\beta, m)^2}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\theta(\alpha, \beta, m)^2}{\alpha \vee \beta} \right|$$

$$\leq \left(1 + \frac{25}{3} \right) \|z\|_\infty \leq 28\|z\|_\infty/3.$$

In particular, since $y_1 \geq \varsigma(y_1, \alpha, \beta) \geq 10\|z\|_\infty$, we must have $\varsigma((y_1, \alpha, \beta) + z) \in [2\|z\|_\infty/3, y_1 + 28\|z\|_\infty/3] \subset [0, y_1 + 7/6]$ and hence $\varsigma((y_1, \alpha, \beta) + z) > 0$. Using the mean value theorem together with the bounds above and Lemma 12.3 we obtain

$$\left| \frac{\varsigma((y_1, \alpha, \beta) + z)}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{\varsigma(y_1, \alpha, \beta)}{\alpha \vee \beta} \right|$$

$$\leq \max_{(u,v) \in [0, y_1 + \frac{7}{6}] \times [\frac{3}{8}, \frac{5}{8}]} \|\nabla f(u,v)\|_1 \cdot [|\varsigma((y_1, \alpha, \beta) + z) - \varsigma(y_1, \alpha, \beta)| \vee |(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta|]$$

$$\leq \left(\frac{1}{3/8} + \frac{(y_1 + \frac{7}{6})}{(3/8)^2} \right) \cdot (10\|z\|_\infty \vee \|z\|_\infty) \leq 110(y_1 + 1)\|z\|_\infty.$$

$$(12.16)$$

Combining (12.14), (12.15) and (12.16) gives the desired inequality (12.13). $\qquad\square$

With Lemma 12.3, Lemma 12.4 and Lemma 12.5 at hand, we are ready to prove Lemma 12.2.

Proof of Lemma 12.2. We first establish that

$$\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega^{\mathrm{LP,s}}_{m,N,k}) \leq 15, \quad \mathrm{cond}(\Xi_{\mathrm{UL}}, \Omega^{\mathrm{UL}, \ell^1, \mathrm{s}}_{m,N,k}) \leq 28, \quad \mathrm{cond}(\Xi_{\mathrm{CL}}, \Omega^{\mathrm{CL}, \ell^1, \mathrm{s}}_{m,N,k}) \leq 2,$$

$$\mathrm{cond}(\Xi_{\mathrm{BP}}, \Omega^{\mathrm{BP}, \ell^1, \mathrm{s}}_{m,N,k}) \leq 24, \quad \mathrm{cond}(\Xi_{\mathrm{BPTV}}, \Omega^{\mathrm{BP,TV,s}}_{m,N,k}) \leq 35, \quad \mathrm{cond}(\Xi_{\mathrm{ULTV}}, \Omega^{\mathrm{UL,TV,s}}_{m,N,k}) \leq 179$$

$$(12.17)$$

for integers $k \geq 1$ and $N > m \geq 4$, and then argue via the inclusions between the "strong" and "weak" input sets to prove (12.8).

We work through each of the problems in turn, calculating their condition numbers.

Linear programming: The active coordinates of $\Omega_{m,N,k}^{\mathrm{LP,s}}$ are $\{(1,1),(1,2)\}$ for the matrix part of the input and $\{1\}$ for the vector part. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{LP,s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{LP},k} + z_y e_1, A(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{LP,s}}$ with $\iota = (y^{\mathrm{LP},k}, A(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{LP},k}$.

Let us assume first that ι is such that $\alpha \neq \beta$. Then, for $\epsilon \in \left(0, y_1 \wedge \frac{1}{2}|\alpha - \beta|\right)$ and $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \leq \epsilon$ we have both $\alpha + z_\alpha \neq \beta + z_\beta$ and $\mathrm{sgn}(\alpha + z_\alpha - (\beta + z_\beta)) = \mathrm{sgn}(\alpha - \beta)$, and so by Lemma 11.1

$$\Xi_{\mathrm{LP}}(\iota^z) = \frac{y_1 + z_y}{(\alpha + z_\alpha) \vee (\beta + z_\beta)}\, v \quad \text{and} \quad \Xi_{\mathrm{LP}}(\iota) = \frac{y_1}{\alpha \vee \beta}\, v,$$

where $v = e_1$ if $\alpha > \beta$, and $v = e_2$ if $\alpha < \beta$.

Next, consider an input $\iota = (y, A(\alpha, \alpha, m, N)) \in \Omega_{m,N,K}^{\mathrm{LP,s}} \cup \Omega_{m,N,K}^{\mathrm{LP,w}}$, and let $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ be such that $0 < \|z\|_\infty \leq y_1 \wedge \frac{1}{2}\alpha$. It then follows by Lemma 11.1 that

$$\Xi_{\mathrm{LP}}(\iota^z) \subset \left\{ \frac{y_1 + z_y}{(\alpha + z_\alpha) \vee (\alpha + z_\beta)}\, (te_1 + (1 - t)e_2) \mid t \in [0,1] \right\} \text{ and}$$

$$\Xi_{\mathrm{LP}}(\iota) = \left\{ \frac{y_1}{\alpha}\, (te_1 + (1 - t)e_2) \mid t \in [0,1] \right\}.$$

Therefore in all possible cases for ι and ι^z with ϵ sufficiently small we must have

$$\mathrm{dist}_\infty \left(\Xi_{\mathrm{LP}}(\iota), \Xi_{\mathrm{LP}}(\iota^z) \right) \leq \max_{\alpha, \beta \in \mathcal{L}} \left| \frac{y_1 + z_y}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{y_1}{\alpha \vee \beta} \right|$$

Moreover, in all the cases above we have $0 \leq y_1 + z_y \leq 2y_1 \leq 4 \cdot 10^{-k} \leq 1$, and $|(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta| \leq \|z\|_\infty$ by Lemma 12.3. Thus, for $\|z\|_\infty$ sufficiently small (in particular insisting that $\|z\|_\infty < 1/8$), the mean value theorem applied to the function $f_{\mathrm{LP}}(u,v) := u/v$ gives

$$\max_{\alpha, \beta \in \mathcal{L}} \left| \frac{y_1 + z_y}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} - \frac{y_1}{\alpha \vee \beta} \right| \leq \max_{\substack{u \in [0,1] \\ v \in [\frac{3}{8}, \frac{5}{8}]}} \|\nabla f_{\mathrm{LP}}(u,v)\|_1 \cdot \|z\|_\infty \leq \left(\frac{8}{3} + \frac{64}{9} \right) \cdot \|z\|_\infty < 10\|z\|_\infty$$

We hence deduce that

$$\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,k}^{\mathrm{LP,s}}) = \sup_{N > m \geq 4} \sup_{\iota \in \Omega_{m,N,k}^{\mathrm{LP,s}}} \limsup_{\epsilon \to 0^+} \sup_{\substack{\iota^z \in \left(\Omega_{m,N,k}^{\mathrm{LP,s}}\right)^{\mathrm{act}} \\ 0 < \|z\|_\infty \leq \epsilon}} \left\{ \frac{\mathrm{dist}_\infty \left(\Xi_{\mathrm{LP}}(\iota), \Xi_{\mathrm{LP}}(\iota^z) \right)}{\|z\|_\infty} \right\} \leq 10.$$

Unconstrained lasso with ℓ^1 regularisation: The active coordinates of $\Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}}$ are $\{(1,1),(1,2)\}$ for the matrix part of the input and $\{1\}$ for the vector part. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{UL},\ell^1,k} + z_y e_1, L(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}$ with $\iota = (y^{\mathrm{UL},\ell^1,k}, L(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{UL},\ell^1,k}$.

For such an ι note that $\alpha \vee \beta = \frac{1}{2} > \frac{\lambda}{2y_1}$. Let $\epsilon \in (0, y_1)$ be small enough so that $\alpha \vee \beta - \epsilon > \frac{\lambda}{2(y_1 + \epsilon)}$ and consider $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \leq \epsilon$. We then have $0 \leq y_1 + z_y \leq 2y_1 \leq 2(1 + \lambda) \leq \frac{8}{3}$ and again $|(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta| \leq \|z\|_\infty$ by Lemma 12.3. Thus an argument analogous to the one presented when analysing $\mathrm{cond}(\Xi_{\mathrm{LP}})$, but employing Lemma 11.2 instead of Lemma 11.1, gives

$$\mathrm{dist}_\infty \left(\Xi_{\mathrm{UL}}(\iota), \Xi_{\mathrm{UL}}(\iota^z) \right)$$

$$\leq \max_{(\alpha, \beta) \in \mathcal{L}} \left| \frac{2(\alpha \vee \beta)y_1 - \lambda}{2(\alpha \vee \beta)^2} - \frac{2\left((\alpha + z_\alpha) \vee (\beta + z_\beta)\right)(y_1 + z_y) - \lambda}{2\left((\alpha + z_\alpha) \vee (\beta + z_\beta)\right)^2} \right|$$

$$\leq \max_{(u,v) \in [0, \frac{8}{3}] \times [\frac{3}{8}, \frac{5}{8}]} \|\nabla f_{\mathrm{UL},\ell^1}(u,v)\|_1 \cdot \|z\|_\infty = \max_{(u,v) \in [0, \frac{8}{3}] \times [\frac{3}{8}, \frac{5}{8}]} \left(\left| \frac{1}{v} \right| + \left| \frac{\lambda}{v^3} - \frac{u}{v^2} \right| \right) \cdot \|z\|_\infty$$

where $f_{\mathrm{UL},\ell^1}(u,v) = \frac{2uv - \lambda}{2v^2}$. Noting that $|\lambda/v^3 - u/v^2| \leq ((\lambda/v^3) \vee (u/v^2))$ and $\lambda \leq 1/3$ we obtain $\mathrm{dist}_\infty(\Xi_{\mathrm{UL}}(\iota), \Xi_{\mathrm{UL}}(\iota^z)) \leq 8/3 + (8^3/3^4 \vee 8^3/3^3) \leq 22$ from which $\mathrm{cond}(\Xi_{\mathrm{UL}}, \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}}) \leq 22$ follows.

Constrained lasso with ℓ^1 regularisation: The active coordinates of $\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}$ are $\{(1,1),(1,2)\}$ for the matrix part of the input and $\{1\}$ for the vector part. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{CL},k} + z_y e_1, L(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}$ with $\iota = (y^{\mathrm{CL},k}, L(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{CL},k}$.

For such an ι let $r = \frac{(\alpha \vee \beta)y_1}{1 + (\alpha \vee \beta)^2} = 2y_1/5 = 4 \cdot 10^{-k}$ and note that $r < 1/2 \leq \tau$ and so Lemma 11.4 applies. Next, let $\epsilon \in (0, y_1 \wedge 1/2)$ be small enough so that $\frac{(\alpha \vee \beta + \epsilon)(y_1 + \epsilon)}{1 + (\alpha \vee \beta + \epsilon)^2} < \tau$ and consider $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \leq \epsilon$. We then have $0 \leq y_1 + z_y \leq 2y_1 \leq 2$ and $|(\alpha + z_\alpha) \vee (\beta + z_\beta) - \alpha \vee \beta| \leq \|z\|_\infty$ by Lemma 12.3. Thus an argument analogous to the one presented when analysing $\mathrm{cond}(\Xi_{\mathrm{LP}})$, but employing Lemma 11.4 instead of Lemma 11.1, gives

$$\mathrm{dist}_\infty\left(\Xi_{\mathrm{CL}}(\iota), \Xi_{\mathrm{CL}}(\iota^z)\right)$$

$$\leq \max_{(\alpha,\beta)\in\mathcal{L}} \left| \left(\tau - \frac{(\alpha \vee \beta)y_1}{1 + (\alpha \vee \beta)^2}\right) - \left(\tau - \frac{((\alpha + z_\alpha) \vee (\beta + z_\beta))(y_1 + z_y)}{1 + ((\alpha + z_\alpha) \vee (\beta + z_\beta))^2}\right) \right|$$

$$\leq \max_{(u,v)\in[0,1]\times[\frac{3}{8},\frac{5}{8}]} \|\nabla f_{\mathrm{CL}}(u,v)\|_1 \cdot \|z\|_\infty = \max_{(u,v)\in[0,1]\times[\frac{3}{8},\frac{5}{8}]} \left(\left|\frac{1}{1+v^2}\right| + \left|\frac{u(1-v^2)}{(1+v^2)^2}\right| \right) \cdot \|z\|_\infty$$

where $f_{\mathrm{CL}}(u,v) = \frac{uv}{1+v^2}$. Noting that $\left|1/(1+v^2)\right| + \left|u(1-v^2)/(1+v^2)^2\right| \leq 2/(1+v^2)^2$ for the specified range of (u,v), we obtain $\mathrm{dist}_\infty(\Xi_{\mathrm{CL}}(\iota), \Xi_{\mathrm{CL}}(\iota^z)) \leq 2/(1 + (3/8)^2)^2 \leq 2$ from which we deduce that $\mathrm{cond}(\Xi_{\mathrm{CL}}, \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}) \leq 2$.

Basis pursuit with ℓ^1 regularisation: As before the active coordinates of $\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}$ are $\{(1,1),(1,2)\}$ for the matrix part of the input and $\{1\}$ for the vector part. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{BP},\ell^1,k} + z_y e_1, L(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}$ with $\iota = (y^{\mathrm{BP},\ell^1,k}, L(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{BP},\ell^1,k}$.

For such an ι note that $\delta < \delta + 10^{-k} \leq y_1 \leq 2$ so for $\epsilon \in (0, y_1 - \delta)$ and $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \leq \epsilon$, an argument analogous to the analysis of $\mathrm{cond}(\Xi_{\mathrm{LP}})$ except employing Lemma 11.3 instead of Lemma 11.1 shows that

$$\mathrm{dist}_\infty\left(\Xi_{\mathrm{BPDN}}(\iota)), \Xi_{\mathrm{BPDN}}(\iota^z))\right) \leq \left| \frac{y_1 - \delta}{\alpha \vee \beta} - \frac{y_1 + z_y - \delta}{(\alpha + z_\alpha) \vee (\beta + z_\beta)} \right|$$

$$\leq \max_{(u,v)\in[0,2]\times[\frac{3}{8},\frac{5}{8}]} \|\nabla f_{\mathrm{BP},\ell^1}(u,v)\|_1 \cdot \|z\|_\infty$$

$$\leq \left(\frac{1}{v} + \left|\frac{u-\delta}{v^2}\right| \right) \cdot \|z\|_\infty \leq \left(\frac{8}{3} + \frac{2}{(3/8)^2} \right) < 17\|z\|_\infty,$$

where $f_{\mathrm{BP},\ell^1}(u,v) = \frac{u-\delta}{v}$, from which we deduce that $\mathrm{cond}(\Xi_{\mathrm{BPDN}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}) \leq 17$.

Basis pursuit with TV regularisation: The active coordinates of $\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{s}}$ are $\{(1,1),(1,N)\}$ for the matrix and $\{1\}$ for the vector. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{BP},\mathrm{TV},k} + z_y e_1, T(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{BP},\mathrm{TV},\mathrm{s}}$ with $\iota = (y^{\mathrm{BP},\mathrm{TV},k}, T(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{BP},\mathrm{TV},k}$.

Suppose that $\alpha \neq \beta$. For $\epsilon \in \left(0, \frac{1}{2}|\alpha - \beta| \wedge \frac{1}{8} \wedge \frac{1}{2}\left(y_1 - \frac{\delta\theta(\alpha,\beta,m)}{m-1}\right)\right)$ and $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \leq \epsilon$ we have $\alpha + z_\alpha \neq \beta + z_\beta$ and, by Lemma 12.4, $y_1 + z_y > \frac{\delta\theta(\alpha+z_\alpha,\beta+z_\beta,m)}{m-1}$. Suppose for now that $\alpha < \beta$. Then, by Lemma 11.5, we have $\Xi_{\mathrm{BPTV}}(\iota^z) = \eta(y + z_y, \alpha + z_\alpha, \beta + z_\beta)$ and thus by Lemma 12.4 we find $\mathrm{dist}_\infty(\Xi_{\mathrm{BPTV}}(\iota), \Xi_{\mathrm{BPTV}}(\iota^z)) = \|\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) - \eta(y_1, \alpha, \beta)\|_\infty \leq 14(y_1+1)\|z\|_\infty < 35\|z\|_\infty$ where the final inequality follows from $y_1 \leq \|y\|_\infty \leq 3/2$, proven in Lemma 12.1. In the case $\alpha > \beta$, the same logic and an application of Lemma 11.5 yields

$\mathrm{dist}_\infty\left(\Xi_{\mathrm{BPTV}}(\iota), \Xi_{\mathrm{BPTV}}(\iota^z)\right) = \|P_{\mathrm{flip}}\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) - P_{\mathrm{flip}}\eta(y_1, \alpha, \beta)\|_\infty \le 14(y_1 + 1)\|z\|_\infty < 35\|z\|_\infty$, as the flipping operator P_{flip} is an isometry.

Next, assume $\alpha = \beta$ and let $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ be such that $0 < \|z\|_\infty \le \frac{1}{8} \wedge \frac{1}{2}\left(y_1 - \frac{\delta\theta(\alpha,\beta)}{m-1}\right)$. Then, by Lemma 11.5, we have both

$$\Xi_{\mathrm{BPTV}}(\iota^z) \cap \{\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta), P_{\mathrm{flip}}\eta(y_1 + z_y, \beta + z_\beta, \alpha + z_\alpha)\} \neq \varnothing$$

and $\Xi_{\mathrm{BPTV}}(\iota) = \{t\eta(y_1, \alpha, \beta) + (1 - t)P_{\mathrm{flip}}\eta(y_1, \beta, \alpha) \mid t \in [0, 1]\}$. Therefore,

$$\mathrm{dist}_\infty\left(\Xi_{\mathrm{BPTV}}(\iota)), \Xi_{\mathrm{BPTV}}(\iota^z)\right)$$
$$\le \|\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) - \eta(y_1, \alpha, \beta)\|_\infty \vee \|P_{\mathrm{flip}}\eta(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) - P_{\mathrm{flip}}\eta(y_1, \alpha, \beta)\|_\infty$$

and thus using Lemma 12.4 we obtain $\mathrm{dist}_\infty\left(\Xi_{\mathrm{BPTV}}(\iota^z), \Xi_{\mathrm{BPTV}}(\iota)\right) \le 14(y_1 + 1)\|z\|_\infty < 35\|z\|_\infty$. Since we have now shown that this bound holds in the cases $\alpha > \beta$, $\alpha < \beta$ and $\alpha = \beta$ we can conclude that $\mathrm{cond}(\Xi_{\mathrm{BPTV}}, \Omega_{m,N,K}^{\mathrm{BP,TV,s}}) \le 35$.

Unconstrained lasso with TV regularisation: The active coordinates of $\Omega_{m,N,k}^{\mathrm{UL,TV,s}}$ are $\{(1, 1), (1, N)\}$ for the matrix and $\{1\}$ for the vector. Thus ι^z is in the active set $\left(\Omega_{m,N,k}^{\mathrm{UL,TV,s}}\right)^{\mathrm{act}}$ only if $\iota^z = (y^{\mathrm{UL,TV},k} + z_y e_1, T(\alpha + z_\alpha, \beta + z_\beta, m, N))$ for some suitable z and α, β such that there is an $\iota \in \Omega_{m,N,k}^{\mathrm{UL,TV,s}}$ with $\iota = (y^{\mathrm{UL,TV},k}, T(\alpha, \beta, m, N))$. For the sake of brevity, we will let $y_1 = y_1^{\mathrm{UL,TV},k}$.

For such an ι we consider $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$ such that $0 < \|z\|_\infty \le \frac{1}{8} \wedge \frac{1}{10}\varsigma(y_1, \alpha, \beta)$, where $\varsigma(y_1, \alpha, \beta) = y_1 - \lambda\theta(\alpha,\beta)^2\left[2(m-1)^2(\alpha \vee \beta)\right]^{-1}$. By Lemma 12.5 we conclude that $\varsigma(y_1 + z_y, \alpha + z_\alpha, \beta + z_\beta) > 0$, and so Lemma 11.6 applies to the input $(y + z_y e_1, T(\alpha + z_\alpha, \beta + z_\beta, m, N))$. An argument analogous to the one presented for Basis Pursuit with TV regularisation employing Lemma 12.5 instead of Lemma 12.4 then yields

$$\mathrm{dist}_\infty\left(\Xi_{\mathrm{ULTV}}(\iota^z), \Xi_{\mathrm{ULTV}}(\iota)\right) \le 112(y_1 + 1) \le 112\left(\frac{107}{180} + 1\right) \le 179,$$

where the penultimate inequality follows from Lemma 12.1. Hence $\mathrm{cond}(\Xi_{\mathrm{ULTV}}, \Omega_{m,N,k}^{\mathrm{UL,TV,s}}) \le 179$, as desired.

This establishes each of the claims in (12.17). It remains to prove (12.8). We do this for linear programming only as the argument for the other cases is analogous. If $K = 1$ there is nothing to prove, so assume w.l.o.g. that $K \ge 2$. Note that the active coordinates for $\Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}}$ are the same as the active coordinates for $\Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}}$. Thus

$$\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}} \cup \Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}}) = \mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}}) \vee \mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}})$$

and (12.17) implies that $\mathrm{cond}(\Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}})$ and $\mathrm{cond}(\Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}})$ are both bounded above by 10 since $K \ge 2$. Next, note that $\Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{w}} \subset \Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}}$. This allows us to conclude that $\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,K}^{\mathrm{BP},\ell^1}) = \mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega_{m,N,K-1}^{\mathrm{BP},\ell^1,\mathrm{s}} \cup \Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}}) \le 10$, and hence

$$\mathrm{cond}(\Xi_{\mathrm{LP}}, \Omega^{\mathrm{BP},\ell^1}) = \sup_{N > m \ge 4} \mathrm{cond}(\Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{s}} \cup \Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{w}}) \le 10.$$

$\square$

Lemma 12.6. *For any natural numbers m, N, and k with $k \ge 1$ and $4 \le m < N$ the FP condition number of every input to any of the problems LP, ℓ^1 BP, ℓ^1 UL, CL, TV BP and TV UL with respective input sets Ω^{LP}, $\Omega^{\mathrm{BP},\ell^1}$, $\Omega^{\mathrm{UL},\ell^1}$, $\Omega^{\mathrm{CL},\ell^1}$, $\Omega^{\mathrm{BP,TV}}$, and $\Omega^{\mathrm{UL,TV}}$ is bounded by 4.*

Proof of Lemma 12.6. Suppose for now that the input is in one of the "strong" input sets, and consider first an input $(y, A) = (y, A(\alpha, \beta, m, N)) \in \Omega_{m,N,k}^{\mathrm{LP,s}}$. Recall that $C_{\mathrm{FP}}(y, A) = \frac{\|y\|_\infty \vee \|A\|_{\max}}{\rho(y,A)}$, where

$$\rho(y, A) = \sup\{\epsilon \mid \exists z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3, \|z\|_\infty \leq \epsilon$$
$$\text{s.t. } (y + z_y e_1, A(\alpha + z_\alpha, \beta + z_\beta, m, N)) \text{ is feasible for LP}\}.$$

Concretely, by recalling that c is the vector of ones of length N and inspecting the explicit form of $A(\alpha + z_\alpha, \beta + z_\beta, m, N)$, we see that it is necessary and sufficient for at least one of $\alpha + z_\alpha$ and $\beta + z_\beta$ to be positive for the LP problem to be feasible. As $\alpha \vee \beta = \frac{1}{2}$, we deduce that $\rho(y, A) = \frac{1}{2}$, and therefore $C_{\mathrm{FP}}(y, A) = \frac{(2 \cdot 10^{-k}) \vee 1}{1/2} = 2$.

Next, note that, for $z = (z_y, z_\alpha, z_\beta) \in \mathbb{R}^3$, the matrices $L(\alpha + z_\alpha, \beta + z_\beta, m, N)$ and $T(\alpha + z_\beta, \beta + z_\beta, m, N)$ are onto as long as at least one of $\alpha + z_\alpha$ and $\beta + z_\beta$ is nonzero, and therefore $\rho(y, A) \geq \frac{1}{2}$ for the problems ℓ^1 BP, CL, and TV BP with inputs (y, A) in $\Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}$, $\Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}$, and $\Omega_{m,N,k}^{\mathrm{BP,TV,s}}$, respectively. By Lemma 12.1 $\|y\|_\infty \leq 2$ and $\|A\|_{\max} \leq 1$, we deduce that $C_{\mathrm{FP}}(y, A) \leq \frac{2}{1/2} = 4$ in the case of ℓ^1 BP, CL, and TV BP.

Finally, inputs $(y, L(\alpha, \beta, m, N))$ and $(y, T(\alpha, \beta, m, N))$ are feasible for the ℓ^1 UL and TV UL problems for any values of y, α, and β, and therefore $C_{\mathrm{FP}}(y, A) = 0$ in the case of ℓ^1 UL and TV UL for any input set. The proof is entirely analogous in the case when the input is in one of the "weak" input sets, and thus the proof of the lemma is complete. $\square$

Lemma 12.7. *Let $(\alpha, \beta) \in \mathcal{L}$. Then $\mathrm{cond}(AA^*), \mathrm{cond}(LL^*) \leq 16/5$. In addition, for $k \geq 1$ and $(\alpha, \beta) \in \mathcal{L}^{BP,TV,k}$ or $(\alpha, \beta) \in \mathcal{L}^{UL,TV,k}$, we have $\mathrm{cond}(TT^*) \leq \frac{16}{5}$.*

Proof of Lemma 12.7. Note that $M := AA^* = LL^* = TT^* = (\alpha^2 + \beta^2) \oplus I_{m-1}$. Therefore $\mathrm{cond}(M) = \|M\|_2 \|M^{-1}\|_2 = ((\alpha^2 + \beta^2) \vee 1) \cdot ((\alpha^2 + \beta^2)^{-1} \vee 1) \leq 1 \cdot \frac{16}{5} = \frac{16}{5}$. $\square$

12.4. Inequalities to control the breakdown epsilons for the TV problems.

Lemma 12.8. *Let $p \in [1, \infty]$, $n \in \mathbb{N}$, and set*

$$y_1 = \delta m[\theta(1/2, 1/2, m)]^{-1} + [7 - 3(\theta(1/2, 1/2, m))^{-1}]10^{-n}/4.$$

Then

$$\min_{\substack{(\alpha,\beta) \in \mathcal{L}^{BP,TV,n} \\ (\alpha',\beta') \in \mathcal{L}^{BP,TV,n}}} \|\eta(y_1, \alpha, \beta) - P_{\mathit{flip}}\eta(y_1, \alpha', \beta')\|_p > 2 \cdot 10^{-n} \tag{12.18}$$

and

$$\max_{(\alpha,\beta) \in \mathcal{L}^{BP,TV,n}} |\eta(y_1, \alpha, \beta)_1 - \eta(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-n+1} 2^{-1/p}/6. \tag{12.19}$$

Proof. We first prove the following inequalities for general $y_1 \geq 0$ and then specialise to the value of y_1 in the statement of the lemma:

$$\min_{\substack{(\alpha,\beta) \in \mathcal{L}^{BP,TV,n} \\ (\alpha',\beta') \in \mathcal{L}^{BP,TV,n}}} \|\eta(y_1, \alpha, \beta) - P_{\mathrm{flip}}\eta(y_1, \alpha', \beta')\|_p \geq 2^{1/p}\left[2y_1 - \frac{2\delta m}{\theta(1/2, 1/2, m)}\right] \tag{12.20}$$

(with the convention $2^1/p = 1$ when $p = \infty$) and, recalling r_n from (12.2),

$$\max_{(\alpha,\beta) \in \mathcal{L}^{BP,TV,n}} |\eta(y_1, \alpha, \beta)_1 - \eta(y_1, \alpha, \beta)_N| \leq 2y_1 - \frac{2\delta[4r_n^2 + (m-1)]}{\theta(1/2, 1/2, m)}. \tag{12.21}$$

To this end, we have for $(\alpha, \beta), (\alpha', \beta') \in \mathcal{L}^{\mathrm{BP,TV},n}$ and $p < \infty$

$$\|\eta(y_1, \alpha, \beta) - P_{\mathrm{flip}}\eta(y_1, \alpha', \beta')\|_p \geq \Big[\big|\eta(y_1, \alpha, \beta)_1 - (P_{\mathrm{flip}}\eta(y_1, \alpha', \beta'))_1\big|^p +$$
$$+ \big|\eta(y_1, \alpha, \beta)_N - (P_{\mathrm{flip}}\eta(y_1, \alpha', \beta'))_N\big|^p \Big]^{1/p}$$
$$= \Big[\big|\eta(y_1, \alpha, \beta)_1 - \eta(y_1, \alpha', \beta')_N\big|^p + \big|\eta(y_1, \alpha, \beta)_N - \eta(y_1, \alpha', \beta')_1\big|^p \Big]^{1/p}$$
$$\geq \Big[2 \cdot \min_{\substack{(u,v)\in\mathcal{L}^{\mathrm{BP,TV},n} \\ (u',v')\in\mathcal{L}^{\mathrm{BP,TV},n}}} |\eta(y_1, u, v)_N - \eta(y_1, u', v')_1|^p \Big]^{1/p},$$

and therefore

$$\min_{\substack{(\alpha,\beta)\in\mathcal{L}^{\mathrm{BP,TV},n} \\ (\alpha',\beta')\in\mathcal{L}^{\mathrm{BP,TV},n}}} \|\eta(y_1, \alpha, \beta) - P_{\mathrm{flip}}\eta(y_1, \alpha', \beta')\|_p \geq \min_{\substack{(\alpha,\beta)\in\mathcal{L}^{\mathrm{BP,TV},n} \\ (\alpha',\beta')\in\mathcal{L}^{\mathrm{BP,TV},n}}} 2^{1/p}|\eta(y_1, \alpha, \beta)_N - \eta(y_1, \alpha', \beta')_1|,$$

where this inequality also holds for $p = \infty$ by an analogous argument. Next, by the definition of η we have, for $(\alpha, \beta), (\alpha', \beta') \in \mathcal{L}^{\mathrm{BP,TV},n}$,

$$\eta(y_1, \alpha, \beta)_N - \eta(y_1, \alpha', \beta')_1 \geq 2y_1 + \frac{\delta(\alpha + \beta)}{\theta(\alpha, \beta)} - \frac{2\delta\theta(\alpha, \beta)}{m-1} - \frac{\delta(\alpha' + \beta')}{\theta(\alpha', \beta')} = 2y_1 + \delta g(\alpha + \beta) + \delta h(\alpha' + \beta')$$

where $g(u) = \frac{u}{\sqrt{(m-1)u^2 + (m-1)^2}} - \frac{2\sqrt{(m-1)u^2 + (m-1)^2}}{m-1}$ and $h(u) = \frac{-1}{\sqrt{(m-1) + (m-1)^2/u^2}}$. A simple calculation yields, for $u \in [1/2, 1]$,

$$g'(u) = \frac{-(m-1)\left[(2u-1)(m-1) + 2u^3\right]}{[(m-1)^2 + u^2(m-1)]^{3/2}} \leq 0,$$

so that $g(u) \geq g(1)$. It is also clear that $h(u)$ is decreasing on $[1/2, \infty)$ and that if $(\alpha, \beta), (\alpha', \beta') \in \mathcal{L}^{\mathrm{BP,TV},n}$ then $(\alpha + \beta), (\alpha' + \beta') \in [3/4, 1]$. Thus

$$\min_{\substack{\alpha,\beta\in\mathcal{L}^{\mathrm{BP,TV},n} \\ \alpha',\beta'\in\mathcal{L}^{\mathrm{BP,TV},n}}} 2^{1/p}|\eta(y_1, \alpha, \beta)_N - \eta(y_1, \alpha', \beta')_1| \geq 2^{1/p}\left(2y_1 + \delta g(1) + \delta h(1)\right).$$

Noting that $g(1) + h(1) = -2\sqrt{m-1 + (m-1)^2}/(m-1) = -2m/\theta(1/2, 1/2, m)$ gives (12.20).

To obtain (12.21), we note that $\eta(y_1, \alpha, \beta)_1 \leq \eta(y_1, \alpha, \beta)_N$ and $\theta(\alpha, \beta, m)$ is increasing in its arguments provided $(\alpha, \beta) \in [0, \infty)^2$. Thus

$$\max_{(\alpha,\beta)\in\mathcal{L}^{\mathrm{BP,TV},n}} |\eta(y_1, \alpha, \beta)_1 - \eta(y_1, \alpha, \beta)_N| = \max_{(\alpha,\beta)\in\mathcal{L}^{\mathrm{BP,TV},n}} \left[2y_1 - 2\delta\frac{\theta(\alpha, \beta, m)}{m-1}\right] = 2y_1 - 2\delta\frac{\theta(r_n, r_n, m)}{m-1}.$$

We conclude (12.21) by noting that $\theta(r_n, r_n, m)/(m-1) \geq \theta(r_n, r_n, m)^2\left[\theta(1/2, 1/2, m)(m-1)\right]^{-1} = [4r_n^2 + (m-1)][\theta(1/2, 1/2, m)]^{-1}$.

With the specific choice $y_1 = \delta m[\theta(1/2, 1/2, m)]^{-1} + [7 - 3(\theta(1/2, 1/2, m))^{-1}]10^{-n}/4$ and using $m \geq 4$, we calculate

$$2^{1/p}\left[2y_1 - \frac{2\delta m}{\theta(1/2, 1/2, m)}\right] \geq \left(\frac{7}{2} - \frac{3}{2\theta(1/2, 1/2, m)}\right)10^{-n} > \left(\frac{7}{2} - \frac{3}{2(m-1)}\right)10^{-n} \geq 2 \cdot 10^{-n},$$

from which we conclude (12.18), as well as

$$2y_1 - \frac{2\delta\left[4r_n^2 + (m-1)\right]}{\theta(1/2, 1/2, m)} = \frac{2\delta(1 - 4r_n^2)}{\theta(1/2, 1/2, m)} + \left(\frac{7}{2} - \frac{3}{2\theta(1/2, 1/2, m)}\right)10^{-n}$$
$$\leq \frac{6\delta \cdot 10^{-n}}{4\delta\theta(1/2, 1/2, m)} + \left(\frac{7}{2} - \frac{3}{2\theta(1/2, 1/2, m)}\right)10^{-n}$$
$$= \frac{7 \cdot 10^{-n}}{2} < \frac{5 \cdot 10^{-n+1}}{12} \leq \frac{5 \cdot 10^{-n+1}2^{-1/p}}{6},$$

where we used $1 - 4r_n^2 \leq 3 \cdot 10^{-n}/(4\delta)$, which is by the definition (12.2). This concludes the proof of (12.19). □

Lemma 12.9. *Let $p \in [1, \infty]$, $n \in \mathbb{N}$, and*

$$y_1 = 7 \cdot 10^{-n}/4 + \lambda(1 + 1/(m-1)) - 3 \cdot 10^{-n}/(4(m-1)).$$

Then

$$\min_{\substack{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n} \\ (\alpha',\beta')\in\mathcal{L}^{UL,TV,n}}} \|\psi(y_1, \alpha, \beta) - P_{flip}\psi(y_1, \alpha', \beta')\|_p > 2 \cdot 10^{-n} \tag{12.22}$$

and

$$\max_{\alpha,\beta\in\mathcal{L}^{UL,TV,n}} |\psi(y_1, \alpha, \beta)_1 - \psi(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-n+1} 2^{-1/p}/6. \tag{12.23}$$

Proof. As in the proof of Lemma 12.8, we will first prove the following inequalities for general $y_1 \geq 0$, and then specialise to y_1 in the statement of the lemma:

$$\min_{\substack{\alpha,\beta\in\mathcal{L}^{UL,TV,n} \\ \alpha',\beta'\in\mathcal{L}^{UL,TV,n}}} \|\psi(y_1, \alpha, \beta) - P_{flip}\psi(y_1, \alpha', \beta')\|_p \geq 2^{1/p}\left[2y_1 - 2\lambda\left(1 + \frac{1}{m-1}\right)\right] \tag{12.24}$$

and, recalling s_n from (12.2),

$$\max_{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n}} |\psi(y_1, \alpha, \beta)_1 - \psi(y_1, \alpha, \beta)_N| \leq 2y_1 - 2\lambda\frac{\theta(s_n, s_n, m)^2}{(m-1)^2}. \tag{12.25}$$

To this end, note that

$$\min_{\substack{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n} \\ (\alpha',\beta')\in\mathcal{L}^{UL,TV,n}}} \|\psi(y_1, \alpha, \beta) - P_{flip}\psi(y_1, \alpha', \beta')\|_p \geq \min_{\substack{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n} \\ (\alpha',\beta')\in\mathcal{L}^{UL,TV,n}}} 2^{1/p}|\psi(y_1, \alpha, \beta)_N - \psi(y_1, \alpha', \beta')_1|. \tag{12.26}$$

Next, a simple calculation and the definition of ψ yields, for $(\alpha, \beta), (\alpha', \beta') \in \mathcal{L}^{UL,TV,n}$,

$$\psi(y_1, \alpha, \beta)_N - \psi(y_1, \alpha', \beta')_1 = 2y_1 - \lambda\left(\frac{2(\alpha+\beta)^2 + 2(m-1) - (\alpha+\beta)}{(m-1)} + \frac{(\alpha'+\beta')}{m-1}\right)$$

$$= 2y_1 - \lambda\left(\frac{2(\alpha+\beta-1/4)^2 + 2(m-1) - 1/8}{(m-1)} + \frac{(\alpha'+\beta')}{m-1}\right) \tag{12.27}$$

By noting that $\alpha + \beta, \alpha' + \beta' \in [1/2, 1]$ we see that the right hand side of this expression is minimised when $\alpha + \beta = \alpha' + \beta' = 1$. Applying these values of $\alpha + \beta$ and $\alpha' + \beta'$ in (12.27) and combining with (12.26) yields (12.24).

For (12.25), we note that $\psi(y_1, \alpha, \beta)_1 \leq \psi(y_1, \alpha, \beta)_N$ and $\theta(\alpha, \beta, m)$ is increasing in its arguments provided $(\alpha, \beta) \in [0, \infty)^2$. Thus

$$\max_{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n}} |\psi(y_1, \alpha, \beta)_1 - \psi(y_1, \alpha, \beta)_N| = \max_{(\alpha,\beta)\in\mathcal{L}^{UL,TV,n}}\left[2y_1 - 2\lambda\frac{\theta(\alpha,\beta,m)^2}{(m-1)^2}\right] \leq 2y_1 - 2\lambda\frac{\theta(s_n, s_n, m)^2}{(m-1)^2}.$$

Now, with the specific choice $y_1 = 7 \cdot 10^{-n}/4 + \lambda(1 + 1/(m-1)) - 3 \cdot 10^{-n}/(4(m-1))$ and using $m \geq 4$, we calculate

$$2^{1/p}\left[2y_1 - 2\lambda\left(1 + \frac{1}{m-1}\right)\right] \geq \left(\frac{7}{2} - \frac{3}{2(m-1)}\right)10^{-n} > 2 \cdot 10^{-n},$$

as well as

$$2y_1 - 2\lambda\frac{\theta(s_n, s_n, m)^2}{(m-1)^2} = \frac{7 \cdot 10^{-n}}{2} + 2\lambda\left(\frac{1}{m-1} - \frac{4s_n^2}{m-1}\right) - \frac{3 \cdot 10^{-n}}{2(m-1)}$$

$$\leq \frac{7 \cdot 10^{-n}}{2} < \frac{5 \cdot 10^{-n+1}}{12} \leq \frac{5 \cdot 10^{-n+1}2^{-1/p}}{6}$$

since $1 - 4s_n^2 \leq 3 \cdot 10^{-n}/(4\lambda)$ by the definition (12.2). This concludes the proof of (12.23). □

12.5. **The strong breakdown epsilons.** Our aim will be to prove that, for each of the computational problems under consideration, we have $\epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) > 10^{-k}$ for $\mathrm{p} \in [0, 1/2)$ and $\epsilon_{\mathbb{P}\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) > 10^{-k}$ for $\mathrm{p} \in [0, 1/3)$. We do so in the following lemma:

Lemma 12.10. *Let k, m, and N be natural numbers with $N > m \geq 4$ and $k \geq 1$, and consider the computational problem $\{\Xi, \Omega_{m,N,k}^{\mathrm{s}}, \mathcal{M}, \Lambda_{m,N}\}$, where $\Omega_{m,N,k}^{\mathrm{s}}$ is one of (12.3) and Ξ is the appropriate solution map. Then there exists a $\hat{\Lambda} \in \mathcal{L}^1(\Lambda_{m,N})$ such that, for the computational problem $\{\Xi, \Omega_{m,N,k}^{\mathrm{s}}, \mathcal{M}, \hat{\Lambda}\}$, we have $\epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) > 10^{-k}$ for $\mathrm{p} \in [0, 1/2)$ and $\epsilon_{\mathbb{P}\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) > 10^{-k}$ for $\mathrm{p} \in [0, 1/3)$.*

Proof. The proof is based on using Proposition 10.5. Concretely, for linear programming, we will construct input sequences $\{\iota_n^1\}_{n=1}^{\infty} \subset \Omega_{m,N,k}^{\mathrm{s}}$, $\{\iota_n^2\}_{n=1}^{\infty} \subset \Omega_{m,N,k}^{\mathrm{s}}$, an input $\iota^0 \in \Omega_{m,N,k}^{\mathrm{s}}$, as well as sets $S^1 \subset \mathbb{R}^m$ and $S^2 \subset \mathbb{R}^m$ such that the following hold:

(i) $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_p > 2 \cdot 10^{-k}$.

(ii) $\iota^0 = (y^0, A^0) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}$ and $\iota_n^j = (y^{j,n}, A^{j,n}) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}$, for $j = 1, 2$ and $n \in \mathbb{N}$, satisfy

$$\|y^{j,n} - y^0\|_\infty, \|A^{j,n} - A^0\|_{\max} \leq 4^{-n}. \tag{12.28}$$

(iii) $\Xi(\iota_n^1) \subset S^1$ and $\Xi(\iota_n^2) \subset S^2$, for all $n \in \mathbb{N}$.

For the ℓ^1 and TV problems we will do the same, except that we respectively write L and T instead of A. Once we have done this, the result will follow by part (ii) of Proposition 10.5. We thus work through each of the computational problems $\{\Xi_{\mathrm{LP}}, \Omega_{m,N,k}^{\mathrm{LP,s}}\}$, $\{\Xi_{\mathrm{BP}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}\}$, $\{\Xi_{\mathrm{UL}}, \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}}\}$, $\{\Xi_{\mathrm{CL}}, \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}\}$, $\{\Xi_{\mathrm{BPTV}}, \Omega_{m,N,k}^{\mathrm{BP,TV,s}}\}$, and $\{\Xi_{\mathrm{ULTV}}, \Omega_{m,N,k}^{\mathrm{UL,TV,s}}\}$ in turn.

Case $\{\Xi_{\mathrm{LP}}, \Omega_{m,N,k}^{\mathrm{LP,s}}\}$: We set

$$\iota_n^0 = (y^{\mathrm{LP},k}, L(1/2, 1/2, m, N)),$$
$$\iota_n^1 = (y^{\mathrm{LP},k}, L(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{LP},k}, L(1/2 - 4^{-n}, 1/2, m, N)),$$
$$S^1 = \{4 \cdot 10^{-k} e_1\}, \quad \text{and} \quad S^2 = \{4 \cdot 10^{-k} e_2\},$$

where e_i is the i-th canonical basis vector of $\mathbb{R}^m$. We note that (12.28) immediately holds, hence establishing (ii). Next, we have $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_p = \|4 \cdot 10^{-k} e_1 - 4 \cdot 10^{-k} e_2\|_p > 2 \cdot 10^{-k}$, yielding (i). Finally, to see (iii), note that Lemma 11.1 implies $\Xi_{\mathrm{LP}}(\iota_n^1) = \{4 \cdot 10^{-k} e_1\}$ and $\Xi_{\mathrm{LP}}(\iota_n^2) = \{4 \cdot 10^{-k} e_2\}$, for $j = 1, 2$ and $n \in \mathbb{N}$.

Case $\{\Xi_{\mathrm{BP}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}\}$: We set

$$\iota^0 = (y^{\mathrm{BP},\ell^1,k}, L(1/2, 1/2, m, N)),$$
$$\iota_n^1 = (y^{\mathrm{BP},\ell^1,k}, L(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{BP},\ell^1,k}, L(1/2 - 4^{-n}, 1/2, m, N)),$$
$$S^1 = \{4 \cdot 10^{-k} e_1\}, \quad \text{and} \quad S^2 = \{4 \cdot 10^{-k} e_2\},$$

from which (12.28) immediately holds, hence establishing (ii). Next, we again have $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_p = \|4 \cdot 10^{-k} e_1 - 4 \cdot 10^{-k} e_2\|_p > 2 \cdot 10^{-k}$, yielding (i). Finally, to see (iii), we use Lemma 11.3 to find that $\Xi_{\mathrm{BPDN}}(\iota_n^1) = \{2(\delta + 2 \cdot 10^{-k} - \delta)e_1\} = \{4 \cdot 10^{-k} e_1\} = S^1$ and $\Xi_{\mathrm{BPDN}}(\iota_n^2) = \{2(\delta + 2 \cdot 10^{-k} - \delta)e_2\} = \{4 \cdot 10^{-k} e_2\} = S^2$.

Case $\{\Xi_{\mathrm{UL}}, \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{s}}\}$: The argument is almost identical to that for $\{\Xi_{\mathrm{BP}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}\}$, except that $y^{\mathrm{UL},\ell^1,k}$ replaces $y^{\mathrm{BP},\ell^1,k}$, and we use Lemma 11.2 to obtain

$$\Xi_{\mathrm{UL}}(\iota_n^1) = \{(\lambda + 2 \cdot 10^{-k} - \lambda)[2(1/2)^2]^{-1} e_1\} = \{4 \cdot 10^{-k} e_1\} = S^1$$

and similarly $\Xi_{\mathrm{UL}}(\iota_n^2) = \{4 \cdot 10^{-k} e_2\} = S^2$.

Case $\{\Xi_{\mathrm{CL}}, \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{s}}\}$: Here, the argument differs slightly from the argument for $\{\Xi_{\mathrm{BP}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}\}$. Again, we replace $y^{\mathrm{BP},\ell^1,k}$ with $y^{\mathrm{CL},k}$. As before, we immediately obtain (ii). We now define $S^1 =$

$\{4 \cdot 10^{-k}e_1 + (\tau - 4 \cdot 10^{-k})e_3\}$ and $S^2 = \{4 \cdot 10^{-k}e_2 + (\tau - 4 \cdot 10^{-k})e_3\}$ and note that $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_p = \|4 \cdot 10^{-k}e_1 - 4 \cdot 10^{-k}e_2\|_p > 2 \cdot 10^{-k}$, and thus (i) is satisfied. To establish (iii), we note that for $(\alpha, \beta) \in \mathcal{L}$ we have $\alpha \vee \beta = 1/2$ and thus the parameter r given in 11.4 satisfies

$$r = (\alpha \vee \beta)y_1^{\mathrm{CL},k}/[1 + (\alpha \vee \beta)^2] = 2 \cdot 10^{-k+1}/5 = 4 \cdot 10^{-k} \leq \tau.$$

Therefore Lemma 11.4 applies and so $\Xi_{\mathrm{CL}}(\iota_n^1) = \{re_1 + (\tau - r)e_3\} = \{4 \cdot 10^{-K}e_1 + (\tau - 4 \cdot 10^{-K})e_3\}$ and similarly $\Xi_{\mathrm{CL}}(\iota_n^2) = \{4 \cdot 10^{-K}e_2 + (\tau - 4 \cdot 10^{-K})e_3\}$, as desired.

Case $\{\Xi_{\mathrm{BPTV}}, \Omega_{m,N,k}^{\mathrm{BP,TV,s}}\}$: We choose n_0 so that $1/2 - 4^{-n_0} \in [r_k, 1/2]$ and, writing $y_1 = y_1^{\mathrm{BP},\ell^1,k}$, we set

$\iota^0 = (y^{\mathrm{BP,TV},k}, T(1/2, 1/2, m, N)),$

$\iota_n^1 = (y^{\mathrm{BP,TV},k}, T(1/2, 1/2 - 4^{-n-n_0}), m, N)), \qquad \iota_n^2 = (y^{\mathrm{BP,TV},k}, T(1/2 - 4^{-n-n_0}), 1/2, m, N)),$

$S^1 = \{P_{\mathrm{flip}}\eta(y_1, \alpha, \beta) \,|\, (\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV},k}\}, \quad \text{and} \quad S^2 = \{\eta(y_1, \alpha, \beta) \,|\, (\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV},k}\}.$

It then follows by the choice of n_0 that ι^0, ι_n^1 and ι_n^2 are all in $\Omega_{m,N,K}^{\mathrm{BP,TV,s}}$. We also immediately have (12.28) and hence (ii) holds. Now, by Lemma 12.8 (where we choose $n = k$ in Lemma 12.8) and more specifically equation (12.18) we have

$$\min_{\substack{x^1 \in S^1 \\ x^2 \in S^2}} \|x^1 - x^2\|_p = \min_{\substack{(\alpha,\beta) \in \mathcal{L}^{\mathrm{BP,TV},K} \\ (\alpha',\beta') \in \mathcal{L}^{\mathrm{BP,TV},K}}} \|\eta(y_1, \alpha, \beta) - P_{\mathrm{flip}}(y_1, \alpha', \beta')\|_p > 2 \cdot 10^{-k},$$

and thus (i) follows. Finally, using Lemma 11.5 we immediately obtain $\Xi_{\mathrm{BPTV}}(\iota_n^j) \subseteq S^j$, for $j = 1, 2$ and $n \in \mathbb{N}$, establishing (iii).

Case $\{\Xi_{\mathrm{ULTV}}, \Omega_{m,N,k}^{\mathrm{UL,TV,s}}\}$: This time the argument is very similar to that for $\{\Xi_{\mathrm{BPTV}}, \Omega_{m,N,k}^{\mathrm{BP,TV,s}}\}$. We list only the differences: first, we replace $y^{\mathrm{BP,TV},k}$ with $y^{\mathrm{UL,TV},k}$. Again, it follows immediately that 12.29 holds. The definitions of S^1 and S^2 are also identical with one difference – we replace η with ψ defined in Lemma 11.6. The argument for (i) is the same except we replace all mentions of Lemma 12.8 with Lemma 12.9 and inequality (12.18) with (12.22). Finally, part (iii) is identical except now we use Lemma 11.6 instead of Lemma 11.5. $\qquad\square$

12.6. **The weak breakdown epsilons.** Similarly to the previous subsection, our aim will be to prove that, for each of the computational problems under consideration, we have both $\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) > 10^{-k+1}$, for $\mathrm{p} \in [0, 1/2)$, and $\epsilon_{\mathrm{B}}^{\mathrm{w}} > 10^{-k+1}$.

Lemma 12.11. *Let k, m, and N be natural numbers with $N > m \geq 4$ and $k \geq 2$, and consider the computational problem $\{\Xi, \Omega_{m,N,k}^{\mathrm{w}}, \mathcal{M}, \Lambda_{m,N}\}$, where $\Omega_{m,N,k}^{\mathrm{w}}$ is one of (12.5) and Ξ is the appropriate solution map. Then there exists a $\hat{\Lambda} \in \mathcal{L}^1(\Lambda_{m,N})$ such that, for the computational problem $\{\Xi, \Omega_{m,N,k}^{\mathrm{s}}, \mathcal{M}, \hat{\Lambda}\}$, we have $\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) > 10^{-k+1}$, for $\mathrm{p} \in [0, 1/2)$, and $\epsilon_{\mathrm{B}}^{\mathrm{w}} > 10^{-k+1}$.*

Proof. The proof reads almost identically to that of Lemma 12.10, so we will only list the differences. Concretely, for linear programming, the proof proceeds by constructing input sequences $\{\iota_n^1\}_{n=1}^{\infty} \subset \Omega$, $\{\iota_n^2\}_{n=1}^{\infty} \subset \Omega$, an $\iota^0 = (y^0, A^0) \in \mathbb{R}^m \times \mathbb{R}^{m \times N}$ (a crucial difference from the proof of Lemma 12.10 is that we no longer require $\iota^0 \in \Omega$) as well as sets $S^1 \subset \mathbb{R}^N$ and $S^2 \subset \mathbb{R}^N$ such that the following hold:

(i) $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_{\infty} > 2 \cdot 10^{-k+1}$.

(ii) For $j = 1, 2$ and each natural n, ι_n^j is a tuple $(y^{j,n}, A^{j,n})$ for $y^{j,n} \in \mathbb{R}^m$ and $A^{j,n}$. These tuples will satisfy

$$\|y^{j,n} - y^0\|_{\infty}, \|A^{j,n} - A^0\|_{\max} \leq 4^{-n}. \tag{12.29}$$

(iii) For each n, both $\Xi(\iota_n^1) \subset S^1$ and $\Xi(\iota_n^2) \subset S^2$.

Again, for the ℓ^1 and TV problems we do the same, except that we respectively write L and T instead of A.

These are the conditions required for us to apply part (i) of Proposition 10.5 and conclude that $\epsilon_{\mathbb{P}\mathrm{B}}^{\mathrm{w}}(\mathrm{p}) > 10^{-k+1}$ for $\mathrm{p} \in [0, 1/2)$ and that $\epsilon_{\mathrm{B}}^{\mathrm{w}} > 10^{-K+1}$. We go through the differences to the proof of Lemma 12.10 for each of the computational problems in turn.

Case $\{\Xi_{\mathrm{LP}}, \Omega_{m,N,k}^{\mathrm{LP,w}}\}$: in the definitions of ι^0, ι_n^1 and ι_n^2 we replace $y^{\mathrm{LP},k}$ with $y^{\mathrm{LP},k-1}$, and every instance of 10^{-k} is replaced by 10^{-k+1}.

Case $\{\Xi_{\mathrm{BP}}, \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{w}}\}$: we replace $y^{\mathrm{BP},\ell^1,k}$ with $y^{\mathrm{BP},\ell^1,k-1}$ and every instance of 10^{-k} is replaced by 10^{-k+1}.

Case $\{\Xi_{\mathrm{UL}}, \Omega_{m,N,k}^{\mathrm{UL},\ell^1,\mathrm{w}}\}$: we replace $y^{\mathrm{UL},\ell^1,k}$ with $y^{\mathrm{UL},\ell^1,k-1}$ and every instance of 10^{-k} is replaced by 10^{-k+1}.

Case $\{\Xi_{\mathrm{CL}}, \Omega_{m,N,k}^{\mathrm{CL},\ell^1,\mathrm{w}}\}$: we replace $y^{\mathrm{CL},k}$ with $y^{\mathrm{CL},k-1}$. We replace $4 \cdot 10^{-k}$ in the definitions of S^1 and S^2 by $4 \cdot 10^{-k+1}$ so that $\inf_{x^1 \in S^1, x^2 \in S^2} \|x^1 - x^2\|_p > 2 \cdot 10^{-K+1}$. Finally, the value of r now becomes $2 \cdot 10^{-k+2}/5 = 4 \cdot 10^{-k+1}$.

Case $\{\Xi_{\mathrm{BPTV}}, \Omega_{m,N,k}^{\mathrm{BP,TV,w}}\}$: we replace $y^{\mathrm{BP,TV},k}$ with $y^{\mathrm{BP,TV},k-1}$ and every instance of 10^{-k} is replaced by 10^{-k+1}. In addition, Lemma 12.8 is applied by setting n to $k-1$ instead of k.

Case $\{\Xi_{\mathrm{ULTV}}, \Omega_{m,N,k}^{\mathrm{UL,TV,w}}\}$: we replace $y^{\mathrm{UL,TV},k}$ with $y^{\mathrm{UL,TV},k-1}$ and every instance of 10^{-k} is replaced by 10^{-k+1}. In addition, Lemma 12.9 is applied by setting n to $k-1$ instead of k.

$\square$

Note that in none of the cases in the proof of Lemma 12.6 is ι^0 contained in Ω, so these arguments are not sufficient to prove that the strong breakdown epsilons are greater than or equal to 10^{-k+1} (indeed, this statement is not true).

13. Proof of Theorem 3.4 – Preliminaries: constructing the subroutines

To construct the algorithms required to prove Proposition 9.33 and part (ii) of Proposition 9.32 and, we require various subroutines. Throughout this section, we consider the computational problem $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}^{\Delta_1} = \{\tilde{\Xi}, \tilde{\Omega}_{m,N}, \mathcal{M}_N, \tilde{\Lambda}_{m,N}\}$, where $\Omega_{m,N}$ is one of (12.6) and Ξ is the corresponding solution map. For the linear programming case, we fix the notation for an element of $\tilde{\Omega}_{m,N}$ by writing $\tilde{\iota} = (\{y_j^{(n)}\}_{n=0}^\infty, \{A_{j,k}^{(n)}\}_{n=0}^\infty)_{j,k}$, corresponding to an $\iota = (y, A) \in \Omega$. For the ℓ^1 and TV problems the notation is analogous, except that we respectively write L and T instead of A.

We begin with a subroutine termed *OutputEta* which applies only to the basis pursuit with TV regularisation problem. Informally, the purpose of this subroutine is to approximate the function η defined in §11.3. The exact specification of the subroutine is given below and a proof of its correctness and complexity is given in Lemma 13.1.

Subroutine OutputEta:

Inputs: Dimensions m, N, and natural numbers k_K and a k_ϵ.

Oracles: $\mathscr{O}_{\mathrm{vec}}$ and $\mathscr{O}_{\mathrm{mat}}$ providing access to the components $y_j^{(n)}$ and $T_{j,k}^{(n)}$ of an input $\tilde{\iota}$.

Output: $\eta^* \in \mathbb{D}^N$ (in the Turing case) or $\eta^* \in \mathbb{R}^N$ (in the BSS case).

1. Set $n_1 = (4k_\epsilon + \mathrm{Len}(N) + 7) \vee (4k_K + 1)$ and use the oracle $\mathscr{O}_{\mathrm{vec}}$ to obtain $y_1' := y_1^{(n_1)}$ and the oracle $\mathscr{O}_{\mathrm{mat}}$ to obtain $\alpha' := T_{1,1}^{(n_1)}$ and $\beta' := T_{1,N}^{(n_1)}$. We then compute a $w \in \mathbb{D}$ so that

$$w \approx \left(\frac{m - 1 + (\alpha' + \beta')^2}{m - 1} \right)^{-1/2} \tag{13.1}$$

to $k_\epsilon + \mathrm{Len}(N) + 4$ bits of precision.

2. Next, set $n_2 = k_\epsilon + \text{Len}(N) + 2$ and compute $\eta_1^* = \eta_2^* = \cdots = \eta_{N-1}^* \in \mathbb{D}$ (or $\mathbb{R}$, in the BSS case) so that

$$\eta_1^* \approx \frac{\delta(\alpha' + \beta')w}{m - 1} \tag{13.2}$$

to n_2 bits of precision, incurred by converting a rational to a dyadic (in the BSS case, we simply assign the right hand side to η_1^*).

3. Similarly we compute $\eta_N^* \in \mathbb{D}$ (or $\mathbb{R}$, in the BSS case)

$$\eta_N^* \approx \frac{\delta(\alpha' + \beta')w}{m - 1} + \frac{y_1'}{(\alpha' \vee \beta')} - \frac{\delta w}{(\alpha' \vee \beta')}\left(\frac{m - 1 + (\alpha' + \beta')^2}{m - 1}\right) \tag{13.3}$$

to n_2 bits of precision, again incurred by converting a rational to a dyadic (in the BSS case, we assign the right hand side to η_N^*).

Lemma 13.1. *Assume that the subroutine 'OutputEta' is applied to an input $\tilde{\iota}$ corresponding to $\iota = (y, T) = (y^{\text{BP,TV},k}, T(\alpha, \beta, m, N)) \in \Omega_{m,N,k}^{\text{BP},\ell^1,\text{s}}$, where $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},k}$ for some natural number k with $k \leq k_K$. Then the output η^* satisfies $\|\eta^* - \eta(y_1, \alpha, \beta)\|_p \leq 10^{-k_\epsilon}$, and the number of digits needed by the oracles as well as the BSS and the Turing runtime are all bounded above by some polynomial in k_K, k_ϵ, and $\log(N)$.*

Proof. Let α and β be as in the statement of the lemma. For ease of notation we also write $z_y = y_1' - y_1$, $z_\alpha = \alpha' - \alpha$ and $z_\beta = \beta' - \beta$ as well as $z = (z_y, z_\alpha, z_\beta)$. Note that then $\|z\|_\infty \leq 2^{-n_1} \leq 10^{-k_\epsilon}/(70N) \wedge 10^{-k_K}/2$, by definition of n_1.

Our argument to prove the correctness of *OutputEta* will rely on bounding $\|\eta^* - \eta(y_1', \alpha', \beta')\|_1$ and $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_1$ separately. For the first of these two terms, we note that $\eta(y_1', \alpha', \beta')$ can be written as

$$\eta(y_1', \alpha', \beta')_1 = \eta(y_1', \alpha', \beta')_2 = \cdots = \eta(y_1', \alpha', \beta')_{N-1} = \frac{\delta(\alpha' + \beta')}{m - 1}\left(\frac{m - 1 + (\alpha' + \beta')^2}{m - 1}\right)^{-\frac{1}{2}}$$

and

$$\eta(y_1', \alpha', \beta')_N = \eta(y_1', \alpha', \beta')_1 + \frac{y_1'}{\alpha' \vee \beta'} - \delta\left(\frac{m - 1 + (\alpha' + \beta')^2}{(\alpha' \vee \beta')(m - 1)}\right)\left(\frac{m - 1 + (\alpha' + \beta')^2}{m - 1}\right)^{-\frac{1}{2}}$$

Thus, using the triangle inequality, we obtain

$$\|\eta^* - \eta(y_1', \alpha', \beta')\|_1 \leq \delta\left(\frac{N(\alpha' + \beta')}{m - 1} + \frac{m - 1 + (\alpha' + \beta')^2}{(\alpha' \vee \beta')(m - 1)}\right)\left|w - \left(1 + \frac{(\alpha' + \beta')^2}{m - 1}\right)^{-\frac{1}{2}}\right| + N2^{-n_2}$$

Since $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},k}$ we must have $\alpha + \beta \leq 1$ and $\alpha \vee \beta = 1/2$. Thus because $z_\alpha, z_\beta \leq 10^{-k_K} \leq 1/4$, we note that both $\alpha' + \beta' \leq 3/2$ and, using Lemma 12.3, $\alpha' \vee \beta' \geq 1/4$. Therefore

$$\delta\left(\frac{N(\alpha' + \beta')}{m - 1} + \frac{m - 1 + (\alpha' + \beta')^2}{(\alpha' \vee \beta')(m - 1)}\right) \leq \delta\left[\frac{3N}{2(m - 1)} + 4\left(1 + \frac{9}{4(m - 1)}\right)\right].$$

Furthermore, as $\delta \leq 1$ and $N > m \geq 4$ we obtain

$$\delta\left[\frac{3N}{2(m - 1)} + 4\left(1 + \frac{9}{4(m - 1)}\right)\right] \leq \frac{N}{2} + 4 + 3 \leq \frac{5N}{2}.$$

Therefore, as w is computed to precision $k_\epsilon + \text{Len}(N) + 4$ so that $2^{-(k_\epsilon + \text{Len}(N) + 4)} \leq 10^{-k_\epsilon}/(10N)$ and $2^{-n_2} \leq 10^{-k_\epsilon}/(10N)$ by definition of n_2, we obtain

$$\|\eta^* - \eta(y_1', \alpha', \beta')\|_1 \leq \frac{5N}{2}\left|w - \left(1 + \frac{(\alpha' + \beta')^2}{m - 1}\right)^{-\frac{1}{2}}\right| + N2^{-n_2} \leq \frac{5N \cdot 10^{-k_\epsilon}}{20N} + \frac{N \cdot 10^{-k_\epsilon}}{4N} = \frac{10^{-k_\epsilon}}{2}.$$

Next, we will bound $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_\infty$ using Lemma 12.4. To apply Lemma 12.4 we must show that $\|z\|_\infty \leq (1/8) \wedge (\mu(y_1, \alpha, \beta)/2)$ where the function μ is defined within the statement of Lemma 12.4. Because $(y^{\mathrm{TV}}, T) \in \Omega_{m,N,k}^{\mathrm{BP},\ell^1,\mathrm{s}}$ with $k \leq k_K$ and $\theta(1/2, 1/2, m) \geq m - 1 \geq 1$ we have

$$y_1 = \frac{\delta m}{\theta(1/2, 1/2, m)} + \left(7 - \frac{3}{\theta(1/2, 1/2, m)}\right)\frac{10^{-k}}{4} \geq \frac{\delta m}{\theta(1/2, 1/2, m)} + \frac{(7-3)\cdot 10^{-k_K}}{4}$$

and thus, because $\theta(\alpha, \beta, m)$ is increasing in both α and β and $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP},\mathrm{TV},k} \subset [1/4, 1/2] \times [1/4, 1/2]$,

$$\mu(y_1, \alpha, \beta) \geq \frac{\delta m}{\theta(1/2, 1/2, m)} - \frac{\delta\theta(1/2, 1/2, m)}{m-1} + \frac{(7-3)\cdot 10^{-k_K}}{4} = 10^{-k_K}$$

Hence, as $\|z\|_\infty \leq 10^{-k_K}/2 \leq (1/8) \wedge (\mu(y_1, \alpha, \beta)/2)$, the conditions of Lemma 12.4 are met.

We conclude, this time using $\|z\|_\infty \leq 10^{-k_\epsilon}/(70N)$, that $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_\infty \leq 14(y_1 + 1)\|z\|_\infty \leq 10^{-k_\epsilon}(y_1+1)/(5N)$. By Lemma 12.1 we have $y_1 \leq 3/2$ so $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_\infty \leq 10^{-k_\epsilon}/(2N)$. Therefore $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_p \leq 10^{-k_\epsilon}/2$.

The proof of correctness completed by combining the already established inequalities $\|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_1 \leq 10^{-k_\epsilon}/2$ and $\|\eta^* - \eta(y_1', \alpha', \beta')\|_1 \leq 10^{-k_\epsilon}/2$. Indeed, we have $\|\eta^* - \eta(y_1', \alpha', \beta')\|_p \leq \|\eta^* - \eta(y_1', \alpha', \beta')\|_1 \leq \|\eta(y_1, \alpha, \beta) - \eta(y_1', \alpha', \beta')\|_1 + \|\eta^* - \eta(y_1', \alpha', \beta')\|_1 \leq 10^{-k_\epsilon}$.

Next, we note that number of digits $n_1 = (4k_\epsilon + \mathrm{Len}(N) + 7) \vee (4k_K + 1)$ needed by the oracles is polynomial in k_K, k_ϵ, and $\log(N)$. All that remains is to bound the complexities. Note that it suffices to show that the Turing runtime is polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$, as this will then imply the desired polynomial bound on the BSS runtime.

To this end, as noted in [58, Page 92-93], we recall that it is possible to use Newton-Raphson iteration to compute the reciprocal square root of an u-bit number to v bits of precision at the same cost (up to constants and asymptotically in u and v) as a multiplication operation on two integers each with number of bits bounded above by $u \vee v$. Since we have already shown that $(\alpha' + \beta')^2 \leq 4$, the number of bits of the numerator of $\frac{m-1+(\alpha'+\beta')^2}{m-1}$ is bounded above by a polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$ and the number of bits of the denominator is $\mathcal{O}(\log(m))$. The length of $k_\epsilon + \mathrm{Len}(N) + 4$ is clearly polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$. Hence (since multiplication of b-bit integers can be done in $\mathcal{O}(b\log(b))$, the runtime of computing w in the Turing model is polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$. In particular, $\mathrm{Len}(w)$ must be bounded above by a polynomial in k_ϵ, k_K and $\mathrm{Len}(N)$.

The right hand sides of both (13.2) and (13.3) involve finitely many additions, multiplications and subtractions of rational numbers each with lengths bounded above by a polynomial in k_ϵ, k_K, $\mathrm{Len}(N)$, and $\mathrm{Len}(\delta)$. Since δ is assumed to be fixed, the numerator and denominator of the right hand side of both (13.2) and (13.3) can be computed in Turing runtime polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$.

For each of equations (13.2) and (13.3) the conversion between the rational (say, q_1/q_2) right hand side to a dyadic is done through integer division to n_2 bits of precision. Again, as noted in [58, Page 92-93] this can be done in Turing runtime polynomial in $\mathrm{Len}(q_1/q_2)$ and n_2. But we have already established that both of these quantities are bounded above by some polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$, and thus the Turing runtime of this step is also polynomial in the same quantities.

We have therefore bounded each of the steps of the subroutine by polynomials in k_ϵ, k_K, and $\mathrm{Len}(N)$. But the subroutine itself only performs finitely many steps and thus the overall Turing runtime is also bounded by a polynomial in k_ϵ, k_K, and $\mathrm{Len}(N)$. The proof is complete by noting that $\mathrm{Len}(N) = \mathcal{O}(\log(N))$. $\qquad\square$

Similarly, we formulate *OutputPsi*, which applies only to the unconstrained lasso with TV regularisation problem. Informally, the purpose of this subroutine is to approximate the function ψ defined in

§11.3. Its exact specification is given below and a proof of its correctness and complexity is given in Lemma 13.2.

Subroutine OutputPsi:

Inputs: Dimensions m, N, and natural numbers k_K and a k_ϵ.

Oracles: $\mathscr{O}_{\text{vec}}$ and $\mathscr{O}_{\text{mat}}$ providing access to the components $y_j^{(n)}$ and $T_{j,k}^{(n)}$ of an input $\tilde{\iota}$.

Output: $\psi^* \in \mathbb{D}^N$ (in the Turing case) or $\psi^* \in \mathbb{R}^N$ (in the BSS case).

1. Set $n_1 = (4k_\epsilon + \text{Len}(N) + 9) \vee (4k_K + 3)$ and use the oracle $\mathscr{O}_{\text{vec}}$ to obtain $y_1' := y_1^{(n_1)}$ and the oracle $\mathscr{O}_{\text{mat}}$ to obtain $\alpha' := T_{1,1}^{(n_1)}$ and $\beta' := T_{1,N}^{(n_1)}$.

2. Next, set $n_2 = 4k_\epsilon + \text{Len}(N) + 1$ and compute $\psi_1^* = \psi_2^* = \cdots = \psi_{N-1}^* \in \mathbb{D}$ (or $\mathbb{R}$, in the BSS case) so that

$$\psi_1^* \approx \frac{\lambda(\alpha' + \beta')}{2(m-1)(\alpha' \vee \beta')} \tag{13.4}$$

to n_2 bits of precision, incurred by converting a rational to a dyadic (in the BSS case, we assign the right hand side to ψ_1^*).

3. Similarly, compute $\psi_N^* \in \mathbb{D}$ (or $\mathbb{R}$, in the BSS case) so that

$$\psi_N^* \approx \frac{\lambda(\alpha' + \beta')}{2(m-1)(\alpha' \vee \beta')} + \frac{1}{\alpha' \vee \beta'} \left(y_1' - \frac{\lambda \left[(\alpha' + \beta')^2 + (m-1) \right]}{2(m-1)(\alpha' \vee \beta')} \right) \tag{13.5}$$

to n_2 bits of precision, again incurred by converting a rational to a dyadic (in the BSS case, we assign the right hand side to ψ_N^*).

Lemma 13.2. *Assume that the subroutine 'OutputPsi' is applied to an input $\tilde{\iota}$ corresponding to $\iota = (y,T) = (y^{\text{UL,TV},k}, T(\alpha, \beta, m, N)) \in \Omega_{m,N,k}^{\text{UL},\ell^1,\text{s}}$, where $(\alpha, \beta) \in \mathcal{L}^{UL,TV,k}$ for some natural number k with $k \leq k_K$. Then the output ψ^* satisfies $\|\psi^* - \psi(y_1, \alpha, \beta)\|_p \leq 10^{-k_\epsilon}$, and the number of digits needed by the oracles as well as the BSS and the Turing runtime are all bounded above by some polynomial in k_K, k_ϵ, and $\log(N)$.*

Proof. We start by proving correctness. Let α and β be as in the statement of the lemma. For ease of notation we also write $z_y = y_1' - y_1$, $z_\alpha = \alpha' - \alpha$ and $z_\beta = \beta' - \beta$ as well as $z = (z_y, z_\alpha, z_\beta)$. Note that then $\|z\|_\infty \leq 2^{-n_1} \leq (10^{-k_\epsilon}/(512N)) \wedge (10^{-k_K}/8)$, by definition of n_1. Next, by definition of n_2, we have $2^{-n_2} \leq 10^{-k_\epsilon}/(2N)$, and thus $\|\psi^* - \psi(y_1', \alpha', \beta')\|_p \leq N \cdot 2^{-n_2} \leq 10^{-k_\epsilon}/2$. It thus suffices to show that $\|\psi(y_1, \alpha, \beta) - \psi(y_1', \alpha', \beta')\|_p \leq 10^{-k_\epsilon}/2$. We will accomplish this by using Lemma 12.5.

As in the proof of Lemma 13.1, let $z_y = y_1' - y_1$, $z_\alpha = \alpha' - \alpha$, and $z_\beta = \beta' - \beta$. To apply Lemma 12.5 we must verify the condition $\|z\|_\infty \leq (1/8) \wedge (\varsigma(y_1, \alpha, \beta)/10)$ where the function ς is defined in Lemma 12.5. Because $(y,T) \in \Omega_{m,N,K}^{\text{UL},\ell^1,\text{s}}$ as well as $m \geq 4$ and $k \leq k_K$

$$y_1 = \frac{7 \cdot 10^{-k}}{4} + \lambda \left(1 + \frac{1}{m-1} \right) - \frac{3 \cdot 10^{-k}}{4(m-1)} \geq \frac{3 \cdot 10^{-k_K}}{2} + \lambda \left(1 + \frac{1}{m-1} \right).$$

Thus the definition of r from 12.5 implies that

$$\varsigma(y_1, \alpha, \beta) \geq \frac{3 \cdot 10^{-k_K}}{2} + \lambda \left(1 + \frac{1}{m-1} \right) - \lambda \frac{\theta(\alpha, \beta, m)^2}{(m-1)^2}$$

$$= \frac{3 \cdot 10^{-k_K}}{2} + \lambda \frac{\theta(1/2, 1/2, m)^2}{(m-1)^2} - \lambda \frac{\theta(\alpha, \beta, m)^2}{(m-1)^2} \geq \frac{3 \cdot 10^{-k_K}}{2}$$

since once again $\theta(\alpha, \beta, m)$ is increasing in α and β, and $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},K} \cup \mathcal{L}^{\text{BP,TV},K-1} \subset [1/4, 1/2] \times [1/4, 1/2]$. Hence $\|z\|_\infty \leq 10^{-k_K}/8 < 3 \cdot 10^{-k_K}/20 \leq (1/8) \wedge (\varsigma(y_1, \alpha, \beta)/10)$ and thus the conditions of Lemma 12.5 are met.

We conclude that $\|\psi(y_1, \alpha, \beta) - \psi(y_1', \alpha', \beta')\|_\infty \leq 112(y_1 + 1)\|z\|_\infty \leq 112(y_1 + 1) \cdot 10^{-k_\epsilon}/(512N)$ where we used $\|z\|_\infty \leq 10^{-k_\epsilon}/(512N)$. By Lemma 12.1, $y_1 \leq 107/180$ so $112(y_1 + 1) \leq 112 \cdot$

$287/180 \leq 256$. Therefore $\|\psi(y_1, \alpha, \beta) - \psi(y_1', \alpha', \beta')\|_\infty \leq 10^{-k_\epsilon}/(2N)$ and thus $\|\psi(y_1, \alpha, \beta) - \psi(y_1', \alpha', \beta')\|_p \leq \|\psi(y_1, \alpha, \beta) - \psi(y_1', \alpha', \beta')\|_\infty \cdot N^{1/p} \leq 10^{-k_\epsilon}/2$, as desired.

Next, we note that number of digits $n_1 = (4k_\epsilon + \text{Len}(N) + 9) \vee (4k_K + 3)$ needed by the oracles is polynomial in k_K, k_ϵ, and $\log(N)$. All that remains is to bound the complexities. Note that it suffices to show that the Turing runtime is polynomial in k_ϵ, k_K, and $\text{Len}(N)$, as this will then imply the desired polynomial bound on the BSS runtime.

The right hand sides of both (13.4) and (13.5) involve finitely many additions, multiplications and subtractions of rational numbers each with lengths bounded above by a polynomial in k_ϵ, k_K, $\text{Len}(N)$, and $\text{Len}(\lambda)$. Since λ is assumed to be fixed, numerator and thee denominator of the right hand side of both (13.2) and (13.3) can thus be computed in Turing runtime complexity polynomial in k_ϵ, k_K, and $\text{Len}(N)$.

For each of equations (13.2) and (13.3) the conversion between the rational (say, q_1/q_2) right hand side to a dyadic is done through integer division to n_2 bits of precision. Again, as noted in [58, Page 92-93] this can be done with complexity polynomial in $\text{Len}(q_1/q_2)$ and n_2. But we have already established that both of these quantities are bounded above by some polynomial in k_ϵ, k_K, and $\text{Len}(N)$, and thus the complexity of this step is also polynomial in the same quantities.

We have therefore bounded each of the steps of the subroutine by polynomials in k_ϵ, k_K, and $\text{Len}(N)$, and, as the subroutine itself only performs finitely many steps, the overall Turing runtime is also bounded by a polynomial in k_ϵ, k_K, and $\text{Len}(N)$. Noting that $\text{Len}(N) = \mathcal{O}(\log(N))$ concludes the proof. $\qquad\square$

Next, we need the following subroutine which operates in the case when the input $\tilde{\iota}$ corresponds to an ι in one of $\Omega_{m,N,k}^{\text{LP,w}}$, $\Omega_{m,N,k}^{\text{BP},\ell^1,\text{w}}$ $\Omega_{m,N,k}^{\text{UL},\ell^1,\text{w}}$, $\Omega_{m,N,k}^{\text{CL},\ell^1,\text{w}}$, $\Omega_{m,N,k}^{\text{BP,TV,w}}$ or $\Omega_{m,N,k}^{\text{UL,TV,w}}$. We title this subroutine *Weak* as it is used in §12.6 to give lower bounds on the weak breakdown epsilons. The subroutine is defined as follows:

Subroutine Weak:

Inputs: Dimensions m, N, and a natural number k_ϵ.

Oracles: $\mathscr{O}_{\text{vec}}$ and $\mathscr{O}_{\text{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$) of an input $\tilde{\iota}$.

Output: $x \in \mathbb{D}^N$ (in the Turing case) or $x \in \mathbb{R}^N$ (in the BSS case).

1. We execute a loop that proceeds as follows – at each iteration, we increase n, starting with $n = 1$. What we do now depends on the problem at hand. In the linear programming case we use the oracle $\mathscr{O}_{\text{mat}}$ to read $A_{1,1}^{(n)}$ and $A_{1,2}^{(n)}$ and set $d = A_{1,1}^{(n)} - A_{1,2}^{(n)}$. For the ℓ^1 problems we likewise read $L_{1,1}^{(n)}$ and $L_{1,2}^{(n)}$ and set $d = L_{1,1}^{(n)} - L_{1,2}^{(n)}$. For the TV problems we read $T_{1,1}^{(n)}$ and $T_{1,N}^{(n)}$ and set $d = T_{1,1}^{(n)} - T_{1,N}^{(n)}$. Next, we branch depending on the value of d:
   a. If $d > 2^{-n+1}$ then we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K+1}e_1\|_p \leq 10^{-k_\epsilon}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K+1}e_1 - (\tau - 4 \cdot 10^{-K+1})e_3\|_p \leq 10^{-k_\epsilon}$. For basis pursuit TV we apply the subroutine *OutputEta* to obtain $\eta^* = OutputEta(m, N, k_K = K - 1, k_\epsilon) \in \mathbb{D}^N$, to which we apply P_{flip} and output as x (so that $x = P_{\text{flip}}\eta^*$). Finally, for unconstrained lasso TV we apply the subroutine *OutputPsi* to obtain $\psi^* = OutputPsi(m, N, k_K = K - 1, k_\epsilon) \in \mathbb{D}^N$, to which we apply P_{flip} and output the result as x (so that $x = P_{\text{flip}}\psi^*$). In all of the above cases we terminate execution after outputting x.
   b. Alternatively, if $d < -2^{-n+1}$ then we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K+1}e_2\|_p \leq 10^{-k_\epsilon}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K+1}e_2 - (\tau - 4 \cdot$

$10^{-K+1})e_3\|_p \leq 10^{-k_\epsilon}$. For basis pursuit TV we apply the subroutine *OutputEta* to obtain $\eta^* = OutputEta(m, N, k_K = K - 1, k_\epsilon) \in \mathbb{D}^N$, which we output as x. Finally, for unconstrained lasso TV we apply the subroutine *OutputPsi* to obtain $\psi^* = OutputPsi(m, N, k_K = K - 1, k_\epsilon) \in \mathbb{D}^N$, which we output as x. In all of the above cases we terminate execution after outputting x.

> If neither of these conditions are met then the loop continues by executing the next iteration.

We have presented the Turing version of the subroutine. For the BSS version, all instances of $\mathbb{D}$ are replaced by $\mathbb{R}$.

Lemma 13.3. *Assume that the subroutine 'Weak' is applied to an input $\tilde{\iota}$ corresponding to ι in one of $\Omega^{\mathrm{LP,w}}_{m,N,K}$, $\Omega^{\mathrm{BP},\ell^1,\mathrm{w}}_{m,N,K}$, $\Omega^{\mathrm{UL},\ell^1,\mathrm{w}}_{m,N,K}$, $\Omega^{\mathrm{CL},\ell^1,\mathrm{w}}_{m,N,K}$, $\Omega^{\mathrm{BP,TV,w}}_{m,N,K}$ or $\Omega^{\mathrm{UL,TV,w}}_{m,N,K}$. Then the subroutine always terminates with an output x such that $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-k_\epsilon}$, for the solution map Ξ corresponding to the problem at hand.*

Proof. We start the proof by analysing the case where the problem is either linear programming or an ℓ^1 regularisation problem. By the assumption that ι is in one of the "weak" input sets, it must be of one of the following forms: $(y^{\mathrm{LP},K-1}, A(\alpha, \beta, m, N))$ for linear programming, $(y^{\mathrm{BP},\ell^1,K-1}, L(\alpha, \beta, m, N))$ for basis pursuit denoising with ℓ^1 regularisation, $(y^{\mathrm{UL},\ell^1,K-1}, L(\alpha, \beta, m, N))$ for unconstrained lasso with ℓ^1 regularisation or $(y^{\mathrm{CL},K-1}, L(\alpha, \beta, m, N))$ for constrained lasso with ℓ^1 regularisation, where $(\alpha, \beta) \in \mathcal{L}, \alpha \neq \beta$. Our argument for each of the cases $\alpha > \beta$ and $\alpha < \beta$ is identical, so we will start by assuming that $\alpha < \beta$. First, we fix an n and assume that the values of each of the variables accessed in the statement of the loop are given at the n-th iteration. In particular, $d \leq \alpha + 2^{-n} - (\beta - 2^{-n}) \leq 2^{-n+1}$ and hence the subroutine never exits at a..

By contrast, for n sufficiently large $d \leq \alpha - \beta + 2^{-n+1} < -2^{-n+1}$ since α is strictly smaller than β and thus eventually the subroutine does exit at b.. Hence the output x of the subroutine satisfies $\|x - 4 \cdot 10^{-K+1}e_2\|_1 \leq 10^{-k_\epsilon}$ (or $\|x - 4 \cdot 10^{-K+1}e_2 - (\tau - 4 \cdot 10^{-K+1})e_3\|_1 \leq 10^{-k_\epsilon}$ if the problem is constrained lasso). We now use Lemma 11.1 in the case of LP, Lemma 11.3 in the case of BP with ℓ^1 regularisation, Lemma 11.2 in the case of unconstrained lasso with ℓ^1 regularisation, or Lemma 11.4 in the case of constrained lasso with ℓ^1 regularisation) to conclude that the subroutine is correct in the case $\alpha < \beta$.

The case $\alpha > \beta$ is identical except now the value d will never be smaller than -2^{-n+1} and instead we will eventually (after sufficiently many iterations, depending on ι) have $d > 2^{-n+1}$. Thus the subroutine always outputs x with $\|x - 4 \cdot 10^{-K+1}e_1\|_1 \leq 10^{-k_\epsilon}$ (or $\|x - 4 \cdot 10^{-K+1}e_1 - (\tau - 4 \cdot 10^{-K+1})e_3\|_1 \leq 10^{-k_\epsilon}$ for constrained lasso), which for $\alpha > \beta$ is the correct output by one of Lemma 11.1, Lemma 11.3, Lemma 11.2 or Lemma 11.4 depending on the computational problem being analysed.

The argument that the subroutine is correct for the TV problems is similar. This time ι is either $(y^{\mathrm{BP,TV},K-1}, T(\alpha, \beta, m, N))$ for basis pursuit or $(y^{\mathrm{UL,TV},K-1}, T(\alpha, \beta, m, N))$ for unconstrained lasso. Once again, if $\alpha < \beta$ the subroutine will exit at b. (and never at a.). This time, however, the subroutine *OutputEta* for basis pursuit denoising is called to find η^* and then outputs $x = \eta^*$ (similarly, for unconstrained lasso, *OutputPsi* is called to find ψ^* and output $x = \psi^*$). Note that in this case, for basis pursuit denoising, by Lemma 11.5 we have $\Xi(\iota) = \{\eta(y_1, \alpha, \beta)\}$ (or for unconstrained lasso we use Lemma 11.6 to obtain $\Xi(\iota) = \{\psi(y_1, \alpha, \beta)\}$), where y_1 is the first coordinate of $y^{\mathrm{BP,TV,w}}$ or $y^{\mathrm{UL,TV,w}}$, as appropriate. Lemma 13.1 for basis pursuit denoising (correspondingly Lemma 13.2 for unconstrained lasso) as well as the inclusion $\Omega^{\mathrm{BP},\ell^1,\mathrm{w}}_{m,N,K} \subset \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,K-1}$ (correspondingly $\Omega^{\mathrm{UL},\ell^1,\mathrm{w}}_{m,N,K} \subset \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,K-1}$ for unconstrained lasso) now yields $\|x - \eta(y_1, \alpha, \beta)\|_p = \|\eta^* - \eta(y_1, \alpha, \beta)\|_p \leq \|\eta^* - \eta(y_1, \alpha, \beta)\|_1 \leq 10^{-k_\epsilon}$ (correspondingly $\|x - \psi(y_1, \alpha, \beta)\|_p \leq 10^{-k_\epsilon}$). Thus for the TV problems the subroutine returns x with $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-k_\epsilon}$ in the case $\alpha < \beta$.

The case $\alpha > \beta$ for the TV problems is identical to the above except now the subroutine will execute a. and never b., giving $x = P_{\text{flip}}\eta^*$ or $x = P_{\text{flip}}\psi^*$. The same argument as before shows that $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \le 10^{-k_\epsilon}$. $\square$

Finally, we need a subroutine 'IdentifyStrongOrWeak' which determines whether the input $\tilde{\iota}$ corresponds to an ι in one of

$$\Omega^{\text{LP,s}}_{m,N,K}, \Omega^{\text{BP},\ell^1,\text{s}}_{m,N,K}, \Omega^{\text{UL},\ell^1,\text{s}}_{m,N,K}, \Omega^{\text{CL},\ell^1,\text{s}}_{m,N,K}, \Omega^{\text{BP,TV,s}}_{m,N,K}, \text{ or } \Omega^{\text{UL,TV,s}}_{m,N,K},$$

(which we call the 'InputStrong' case) or an ι in of

$$\Omega^{\text{LP,w}}_{m,N,K}, \Omega^{\text{BP},\ell^1,\text{w}}_{m,N,K}, \Omega^{\text{UL},\ell^1,\text{w}}_{m,N,K}, \Omega^{\text{UL},\ell^1,\text{w}}_{m,N,K}, \Omega^{\text{BP,TV,w}}_{m,N,K}, \text{ or } \Omega^{\text{UL,TV,w}}_{m,N,K},$$

(which we call the 'InputWeak' case), provided either of these two cases occurs.

Subroutine IdentifyStrongOrWeak:

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\text{vec}}$ providing access to the components $y_j^{(n)}$ of an input $\tilde{\iota}$.

Output: Either 'InputStrong' or 'InputWeak'.

1. We set two values, t_0 and t_1, depending on the problem at hand:
   a. For linear programming, we set $t_0 = 2 \cdot 10^{-K}$ and $t_1 = 2 \cdot 10^{-K+1}$.
   b. For basis pursuit with ℓ^1 regularisation we set $t_0 = 2 \cdot 10^{-K} + \delta$ and $t_1 = 2 \cdot 10^{-K+1} + \delta$.
   c. For unconstrained lasso with ℓ^1 regularisation we set $t_0 = 2 \cdot 10^{-K} + \lambda$ and $t_1 = 2 \cdot 10^{-K+1} + \lambda$.
   d. For constrained lasso we set $t_0 = 10^{-K+1}$ and $t_1 = 10^{-K+2}$.
   e. For basis pursuit with TV we compute $[(m-1)/m]^{-1/2}$ to $4K$ bits of precision, yielding some value w. We then set
   $$t_0 = \delta w + \left(7 - \frac{3}{m}\right)\frac{10^{-K}}{4}, \quad t_1 = \delta w + \left(7 - \frac{3}{m-1}\right)\frac{10^{-K+1}}{4}.$$
   f. For unconstrained lasso with TV we set $t_0 = 7 \cdot 10^{-K}/4 + \lambda(1 + 1/(m-1))$ and $t_1 = 3 \cdot 10^{-K+1}/2 + \lambda(1 + 1/(m-1))$.

2. Using the oracle $\mathscr{O}_{\text{vec}}$, we read $y_1' := y_1^{(4K)}$. If $y_1' \le t_0 + 2 \cdot 2^{-4K}$ we output 'InputStrong' and terminate the subroutine. Else if $y_1' \ge t_1 - 2 \cdot 2^{-4K}$ we output 'InputWeak'. The subroutine then terminates.

Lemma 13.4. *The subroutine 'IdentifyStrongOrWeak' correctly identifies the cases 'InputStrong' and 'InputWeak' as described above. Moreover, the number of digits needed by the oracles as well as the BSS and the Turing runtime are all bounded above by some polynomial in $\log(m)$.*

Proof. Suppose first that $\iota = (y, \cdot)$ is in one of $\Omega^{\text{LP,s}}_{m,N,K}$, $\Omega^{\text{BP},\ell^1,\text{s}}_{m,N,K}$, $\Omega^{\text{UL},\ell^1,\text{s}}_{m,N,K}$, $\Omega^{\text{CL},\ell^1,\text{s}}_{m,N,K}$, $\Omega^{\text{BP,TV,s}}_{m,N,K}$, or $\Omega^{\text{UL,TV,s}}_{m,N,K}$. For linear programming, basis pursuit with ℓ^1 regularisation, unconstrained lasso with ℓ^1 regularisation and constrained lasso we have $y_1 = t_0$ and so $y_1 \le t_0 + 2^{-4K}$. For basis pursuit TV, we have

$$y_1 \le \frac{\delta m}{\theta(1/2, 1/2, m)} + \left[7 - \frac{3}{m}\right]\frac{10^{-K}}{4} \le \delta w + \left|\delta\left(w - \frac{m}{\theta(1/2, 1/2, m)}\right)\right| + \left[7 - \frac{3}{m}\right]\frac{10^{-K}}{4}$$

$$\le \delta w + \delta \cdot 2^{-4K} + \left[7 - \frac{3}{m}\right]\frac{10^{-K}}{4} \le t_0 + 2^{-4K}$$

where the first inequality holds because $\theta(1/2, 1/2, m) \le m$ and the final inequality because $\delta \le 1$. For unconstrained lasso TV

$$y_1 = \frac{7 \cdot 10^{-K}}{4} + \lambda\left[1 + \frac{1}{m-1}\right] - \frac{3 \cdot 10^{-K}}{4(m-1)} \le \frac{7 \cdot 10^{-K}}{4} + \lambda\left[1 + \frac{1}{m-1}\right] = t_0 \le t_0 + 2^{-4K}.$$

Thus in each of the cases $y_1' \leq y_1 + 2^{-4K} \leq t_0 + 2 \cdot 2^{-4K}$ and so the subroutine outputs 'InputStrong'.

We now consider the case where $\iota = (y, \cdot)$ is in one of $\Omega_{m,N,K}^{\mathrm{LP,w}}$, $\Omega_{m,N,K}^{\mathrm{BP},\ell^1,\mathrm{w}}$, $\Omega_{m,N,K}^{\mathrm{UL},\ell^1,\mathrm{w}}$, $\Omega_{m,N,K}^{\mathrm{BP,TV,w}}$, or $\Omega_{m,N,K}^{\mathrm{UL,TV,w}}$. We first show that $y_1 \geq t_1 - 2^{-4K}$ and hence $y_1' \geq t_1 - 2 \cdot 2^{-4K}$. For linear programming, basis pursuit with ℓ^1 regularisation, unconstrained lasso (with both ℓ^1 and TV regularisation) and constrained lasso we have $y_1 = t_1$ and thus the claim is true. For basis pursuit with TV, we have

$$y_1 \geq \frac{\delta m}{\theta(1/2, 1/2, m)} + \left[7 - \frac{3}{m-1}\right]\frac{10^{-K+1}}{4} \geq \delta w - \delta \cdot 2^{-4K} + \left[7 - \frac{3}{m-1}\right]\frac{10^{-K+1}}{4} \geq t_1 - 2^{-4K}$$

since $\delta \leq 1$, as claimed. Finally, for unconstrained lasso we have

$$y_1 = \frac{7 \cdot 10^{-K+1}}{4} + \lambda\left[1 + \frac{1}{m-1}\right] - \frac{3 \cdot 10^{-K+1}}{4(m-1)} \geq \frac{6 \cdot 10^{-K+1}}{4} + \lambda\left[1 + \frac{1}{m-1}\right] = t_1$$

since $m \geq 4$ and so the claim follows. Therefore the subroutine outputs 'InputWeak' provided we can also show that $y_1' > t_0 + 2 \cdot 2^{-4K}$ (i.e. the subroutine does not branch and output 'InputStrong').

We thus will aim to show that $y_1' > t_0 + 2 \cdot 2^{-4K}$ for each of the computational problems. For linear programming we have $y_1' \geq 2 \cdot 10^{-K+1} - 2^{-4K} > 2 \cdot 10^{-K} + 2 \cdot 2^{-4K} = t_0 + 2 \cdot 2^{-4K}$. The same argument for basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation gives $y_1' > t_0 + 2 \cdot 2^{-4K}$. Similarly, for constrained lasso we get $y_1' \geq 10^{-K+2} - 2^{-4K} > 10^{-K+1} + 9 \cdot 10^{-K+1} - 5 \cdot 10^{-K}/4 > t_0 + 2 \cdot 2^{-4K}$. For basis pursuit with TV, starting from $y_1 \geq t_1 - 2^{-4K}$ we get

$$y_1 \geq t_0 - 2^{-4K} + \left[7 - \frac{3}{m-1}\right]\frac{10^{-K+1}}{4} - \left[7 - \frac{3}{m}\right]\frac{10^{-K}}{4} = t_0 - 2^{-4K} + \left[63 - \frac{30}{m-1} + \frac{3}{m}\right]\frac{10^{-K}}{4}$$

and since $-30/(m-1) + 3/m$ is increasing in m, for $m \in [4, \infty)$ (as can be seen by analysing the derivative), we obtain $y_1 \geq t_0 - 2^{-4K} + (63 - 30/3 + 3/4)\frac{10^{-K}}{4} > t_0 - 2^{-4K} + 5 \cdot 10^{-K} > t_0 + 3 \cdot 2^{-4K}$ and thus $y_1' \geq y_1 - 2^{-4K} > t_0 + 2 \cdot 2^{-4K}$. Finally for unconstrained lasso with TV we start from $y_1 \geq t_1$ and argue

$$y_1 \geq t_1 = \frac{30 \cdot 10^{-K}}{2} + \lambda\left[1 + \frac{1}{m-1}\right] = \frac{7 \cdot 10^{-K}}{4} + \lambda\left[1 + \frac{1}{m-1}\right] + \frac{23 \cdot 10^{-K}}{2} \geq t_0 + 4 \cdot 2^{-4K},$$

so that $y_1' \geq y_1 - 2^{-4K} > t_0 + 2 \cdot 2^{-4K}$.

Next, we note that the only call to the oracle $\mathscr{O}_{\mathrm{vec}}$ requires $4K$ digits, which is constant, as K is assumed to be fixed. It remains to estimate the complexity. Note that it suffices to show that the Turing runtime is polynomial in $\log(m)$, as this will then imply the desired polynomial bound on the BSS runtime.

We do so by separately considering the different possible problems.

a. For linear programming, basis pursuit, unconstrained lasso with ℓ^1 and constrained lasso, the runtime of computing the numerator and the denominator of t_0 and t_1 is $\mathcal{O}(1)$ since K, δ, λ, and τ are all assumed to be fixed.

b. For basis pursuit with TV, the computation of w can be done as in Lemma 13.1 - we use Newton-Raphson iteration to compute w as in [58, Page 92-93]. The complexity of this operation, done to $4K$ bits of precision, is polynomial in $\mathrm{Len}[m/(m-1)]$ and $4K$. Since $\mathrm{Len}[m/(m-1)] = \mathcal{O}(\log(m))$ and K is fixed, the overall complexity of computing w is polynomial in $\log(m)$. Note that such a w also has $\mathrm{Len}(w)$ bounded above by a polynomial in $\log(m)$. The computation of t_0 and t_1 is then done by finitely many arithmetic operations on fractions each of lengths bounded above by a polynomial in $\mathrm{Len}(\delta)$, $\mathrm{Len}(w)$, $\mathrm{Len}(m)$ and $\mathrm{Len}(10^{-K+1})$. Since δ and K are assumed to be fixed across all inputs and $\mathrm{Len}(w)$ is bounded above by a polynomial in $\log(m)$, the overall complexity of this step is bounded above by a polynomial in $\log(m)$.

c. For unconstrained lasso with TV the complexity of computing t_0 and t_1 is bounded above by a polynomial in $\text{Len}(\lambda)$, $\text{Len}(m)$, and K. Since λ and K are fixed, the complexity of this step is bounded above by a polynomial in $\log(m)$.

In all the cases above both t_0 and t_1 are computed in Turing runtime bounded above by a polynomial in $\log(m)$, so their length must also be bounded above by a polynomial in $\log(m)$. Furthermore, by Lemma 12.1, we must have $\text{Len}(y_1') \leq \text{Len}(y_1) + 4K = \mathcal{O}(1)$. Hence the comparisons in step 2 can be done in complexity polynomial in $\log(m)$. The final output of 'InputWeak' or 'InputStrong' can be done as an $\mathcal{O}(1)$ boolean assignment.

We conclude that in the bit complexity model the subroutine '*IdentifyStrongOrWeak*' takes at most some polynomial in $\log(m)$ bit operations. Estimating the arithmetic complexity is simpler - there are finitely many arithmetic operations done, except in step 1 where the number of Newton-Raphson iterates required can be bound by a polynomial in $\log(m)$. Each Newton-Raphson iteration takes finitely many arithmetic operations and hence the overall arithmetic complexity is bounded by a polynomial in $\log(m)$.

$\square$

14. Proof of Theorem 3.4: parts (i) and (ii)

Parts (i) and (ii) of Theorem 3.4 are formally stated in Proposition 9.32, which we now prove with the set Ω being one of (12.7) depending on the problem. Before proceeding to the breakdown epsilon bounds and algorithm constructions, we note that Lemmas 12.2, 12.6, and 12.7 guarantee the desired bounds on the condition numbers and Lemma 12.1 establishes the upper and lower bounds on the size of the inputs.

14.1. **Proof of Proposition 9.32.** We consider the fixed-dimensional computational problems $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}$, where $\Omega_{m,N}$ is one of (12.6) with $k = K$. Writing $\Omega_{m,N} = \Omega_{m,N}^{\text{s}} \cup \Omega_{m,N}^{\text{w}}$, for the corresponding "strong" and "weak" components as defined in (12.3) and (12.5), Lemma 12.10 establishes the existence of a $\hat{\Lambda}^{\text{s}} = \{f_j^{\text{s}} \mid j \leq n_{\text{var}}, n \in \mathbb{N}\} \in \mathcal{L}^1(\Lambda_{m,N})$ such that, for the computational problem $\{\Xi, \Omega_{m,N}^{\text{s}}, \mathcal{M}_N, \hat{\Lambda}^{\text{s}}\}$, we have $\epsilon_{\mathbb{P}h\text{B}}^{\text{s}}(\text{p}) > 10^{-K}$, for $\text{p} \in [0, 1/2)$, as well as $\epsilon_{\mathbb{P}\text{B}}^{\text{s}}(\text{p}) > 10^{-K}$, for $\text{p} \in [0, 1/3)$.

Similarly, Lemma 12.11 establishes the existence of a $\hat{\Lambda}^{\text{w}} = \{f_{j,n}^{\text{w}} \mid j \leq n_{\text{var}}, n \in \mathbb{N}\} \in \mathcal{L}^1(\Lambda_{m,N})$ such that, for the computational problem $\{\Xi, \Omega_{m,N}^{\text{w}}, \mathcal{M}_N, \hat{\Lambda}^{\text{w}}\}$, we have $\epsilon_{\mathbb{P}\text{B}}^{\text{w}}(\text{p}) > 10^{-(K-1)}$ for $\text{p} \in [0, 1/2)$ and $\epsilon_{\text{B}}^{\text{w}} > 10^{-(K-1)}$.

Now, defining $\hat{\Lambda}_{m,N} := \{f_{j,n} \mid j \leq n_{\text{var}}, n \in \mathbb{N}\}$, where we let $f_{j,n}(\iota) = f_{j,n}^{\text{s}}(\iota)$ if $\iota \in \Omega_{m,N}^{\text{s}}$ and $f_{j,n}(\iota) = f_{j,n}^{\text{w}}(\iota)$ if $\iota \in \Omega_{m,N}^{\text{w}}$, for $j \leq n_{\text{var}}$ and $n \in \mathbb{N}$, we have that $\hat{\Lambda}_{m,N}$ provides Δ_1-information for $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}$, and in view of Remark 10.4, we have that all the breakdown epsilon bounds mentioned above also hold for $\{\Xi, \Omega_{m,N}, \mathcal{M}_N, \hat{\Lambda}_{m,N}\}$. This already establishes part (i) of Proposition 9.32 as well as the breakdown epsilon bound in part (ii) (and, indeed, the breakdown epsilon bounds in Proposition 9.33, which will be useful later).

In order to complete the proof of part (ii) of Proposition 9.32, it remains to show the existence of a recursive (i.e., implementable on a Turing machine) algorithm (which we will call *Randomised K digit algorithm*) that returns K correct digits with probability greater than or equal to $2/3$ on all inputs of the problems $\{\tilde{\Xi}, \tilde{\Omega}_{m,N}, \mathcal{M}_N, \tilde{\Lambda}_{m,N}\} = \{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}^{\Delta_1}$ for varying m and N, where $\Omega_{m,N}$ is one of (12.6) and Ξ is the corresponding solution map. As in the previous section, we fix the notation for an element $\tilde{\iota}$ of $\tilde{\Omega}_{m,N}$. For the linear programming case, we write $\tilde{\iota} = \left(\{y_j^{(n)}\}_{n=0}^{\infty}, \{A_{j,k}^{(n)}\}_{n=0}^{\infty}\right)_{j,k}$, corresponding to an $\iota = (y, A) \in \Omega_{m,N}$. For the ℓ^1 and TV problems the notation is analogous, except that we respectively write L and T instead of A.

To construct this algorithm, we need a randomised subroutine, which we call *BiasedCoinFlip*, that takes a natural number n and returns 'true' with probability $1/n$ and 'false' with probability $1-1/n$. Note

that this subroutine halts with probability 1, and, for each execution of the subroutine, the probabilities of returning 'true', respectively 'false', are assumed independent of previous executions of the subroutine. Such a subroutine can easily be constructed using a randomised Turing machine with access to coin flips that return true with probability $1/2$ and false with probability $1/2$.

To construct the desired *Randomised K digit algorithm*, we first need to design a subroutine *Guess* that randomly chooses between K digit approximations to two plausible solutions, each with probability $1/2$. Concretely, we define:

Subroutine Guess:

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\mathrm{vec}}$ and $\mathscr{O}_{\mathrm{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$) of an input $\tilde{\iota}$.

Output: A potential solution vector $x \in \mathbb{D}^N$ (in the Turing case) or $x \in \mathbb{R}^N$ (in the BSS case).

1. First, we make a random coin flip that returns 'true' with probability $1/2$ and 'false' with probability $1/2$.
   a. If the coin flip outputted 'true' then the output of *Guess* depends on the problem at hand: for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_1\|_p \leq 10^{-K}$, for constrained lasso we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \leq 10^{-K}$, for basis pursuit with TV regularisation we output the result of *OutputEta*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ and for unconstrained lasso with TV regularisation we output the result of *OutputPsi*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$.
   b. If the coin flip outputted 'false' then our output depends on the problem at hand: for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2\|_p \leq 10^{-K}$, for constrained lasso we output $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \leq 10^{-K}$, for basis pursuit with TV regularisation we output the result of P_{flip}*OutputEta*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ and for unconstrained lasso with TV regularisation we output the result of P_{flip}*OutputPsi*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$.

With the subroutine *Guess* and the subroutines constructed in §13, we are ready to specify the desired *Randomised K digit algorithm*. In accordance to the claim of part (ii) of Proposition 9.32, this algorithm does not necessarily halt.

Randomised K digit algorithm

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\mathrm{vec}}$ and $\mathscr{O}_{\mathrm{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$) of an input $\tilde{\iota}$.

Output: With probability at least $2/3$, some vector $x \in \mathbb{R}^N$ with $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$.

1. We execute *IdentifyStrongOrWeak*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N)$. If this evaluates to 'InputWeak', we execute the subroutine *Weak*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_\epsilon = K)$ and terminate. Otherwise, we continue to step 2.
2. Initialise $n = 1$ and execute the following loop: First, execute *BiasedCoinFlip*$(2^{n-1} + 2)$. If this subroutine returns 'true' then we execute *Guess*$^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N)$ and terminate. If instead *BiasedCoinFlip* returns 'false' then we increment n. Next, in the linear programming case we use the oracle $\mathscr{O}_{\mathrm{mat}}$ to read $A_{1,1}^{(n)}$ and $A_{1,2}^{(n)}$ and set $d = A_{1,1}^{(n)} - A_{1,2}^{(n)}$. For the ℓ^1 problems we read $L_{1,1}^{(n)}$ and $L_{1,2}^{(n)}$ and set $d = L_{1,1}^{(n)} - L_{1,2}^{(n)}$, and for the TV problems we read $T_{1,1}^{(n)}$ and $T_{1,N}^{(n)}$ and set $d = T_{1,1}^{(n)} - T_{1,N}^{(n)}$. We then branch depending on the value of d:

a. If $d > 2 \cdot 2^{-n}$ then we choose $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_1\|_p \le 10^{-K}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we choose $x \in \mathbb{D}^{-K}$ $\|x - 4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \le 10^{-K}$. For basis pursuit TV, we set x to be the result of $P_{\text{flip}} OutputEta^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$. Finally, for unconstrained lasso with TV regularisation we set x to be the result of $P_{\text{flip}} OutputPsi^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$. In all cases, we output x and then terminate the procedure.

b. Alternatively, if $d < -2 \cdot 2^{-n}$ then we choose $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2\|_p \le 10^{-K}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we choose $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \le 10^{-K}$. For basis pursuit TV, we set x to be the result of the subroutine $OutputEta^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$. Finally, for unconstrained lasso with TV regularisation we set x to be the result of $OutputPsi^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$. In all cases, we output x and then terminate the procedure.

If neither of these conditions are met then the loop continues by incrementing n and executing the next iteration.

We need to prove that this algorithm does indeed achieve what is stated in its preamble, i.e., we need to show that, for each given input, the algorithm terminates with probability greater than or equal to $2/3$ with a correct output, i.e., a vector x at most 10^{-K} away from a true solution. To this end, we let ι be the element of Ω that $\tilde{\iota}$ corresponds to, and consider the following four cases separately:

Case 1 : For this case, ι is of the following form: $\left(y^{\text{LP,s}}, A(\alpha, 1/2, m, N)\right)$ for linear programming, $\left(y^{\text{BP},\ell^1,\text{s}}, L(\alpha, 1/2, m, N)\right)$ for basis pursuit denoising with ℓ^1 regularisation, $\left(y^{\text{UL},\ell^1,\text{s}}, L(\alpha, 1/2, m, N)\right)$ for unconstrained lasso, $\left(y^{\text{CL,s}}, L(\alpha, 1/2, m, N)\right)$ for constrained lasso, $\left(y^{\text{BP,TV,s}}, T(\alpha, 1/2, m, N)\right)$ for basis pursuit denoising with TV regularisation and $\left(y^{\text{UL,TV,s}}, T(\alpha, 1/2, m, N)\right)$ for unconstrained lasso with TV regularisation, where $1/4 < \alpha < 1/2$ for the ℓ^1 problems, $r_n < \alpha < 1/2$ for basis pursuit with TV, and $s_n < \alpha < 1/2$ for unconstrained lasso with TV.

In this case, Lemma 11.1 (respectively Lemma 11.3 or Lemma 11.2) show that the solution is $4 \cdot 10^{-K} e_2$ for linear programming (respectively, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation). For constrained lasso, applying Lemma 11.4 shows that the solution is $4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K} e_2)$. Similarly, Lemma 11.5 shows that the solution is $\eta(y_1, \alpha, 1/2)$ for basis pursuit with TV regularisation and Lemma 11.6 shows that the solution is $\psi(y_1, \alpha, 1/2)$ for unconstrained lasso with TV regularisation. Therefore, for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation the algorithm is correct whenever it outputs $x \in \mathbb{D}^N$ within 10^{-K} of $4 \cdot 10^{-K} e_2$, and is likewise correct for constrained lasso whenever it outputs $x \in \mathbb{D}^N$ within 10^{-K} of $4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K} e_2)$. The correctness of $OutputEta$ (Lemma 13.1) thus implies that the algorithm is correct for basis pursuit TV whenever the output is $OutputEta^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ and similarly the correctness of $OutputPsi$ (Lemma 13.2) implies that the algorithm is correct for unconstrained lasso TV whenever the output is $OutputPsi^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$.

By Lemma 13.4, the subroutine *IdentifyStrongOrWeak* in step 1 of the algorithm will evaluate to 'InputStrong', and hence the algorithm will proceed with the loop in step 2. We therefore proceed with an analysis of this loop.

Next, let F_n be the event that the subroutine *Guess* is executed in the n-th iteration of the loop and note that, on the event $\left(\bigcup_{r=1}^n F_r\right)^c$, the value of d after n iterations satisfies $d \le \alpha + 2^{-n} - (1/2 - 2^{-n}) < 2 \cdot 2^{-n}$ (since $\alpha < 1/2$) and hence step 2a never results in the termination of the algorithm. By contrast, $d \le \alpha + 2^{-n} - 1/2 + 2^{-n}$ and since α is independent of n and strictly smaller than $1/2$ this expression

will be smaller than $-2 \cdot 2^{-n}$ for sufficiently large n. Now, define

$$n_0 = \inf\{n \in \mathbb{N} \mid F_n \text{ does not occur and } d < -2 \cdot 2^{-n}\}.$$

Note that the value of n_0 depends on $\tilde{\iota}$ through d.

Then $\mathbb{P}(F_n) = 0$, for $n > n_0$, whereas for $n \leq n_0$, $\mathbb{P}(F_n)$ is equal to the probability that the algorithm has executed $n-1$ iterations of the loop without terminating and the execution of $BiasedCoinFlip(2^{n-1} + 2)$ returns 'true'. Note that these two events are independent since the result of $BiasedCoinFlip(2^{n-1} + 2)$ is independent of all prior calls to the subroutine $BiasedCoinFlip$. Thus $\mathbb{P}(F_n) = \left[1 - \mathbb{P}\left(\bigcup_{r=1}^{n-1} F_r\right)\right](2^{n-1} + 2)^{-1}$ for $n \leq n_0$ and, since the events F_r, for $r = 1, \ldots, n-1$, are disjoint, we obtain the recurrence $\mathbb{P}(F_n) = \left[1 - \sum_{r=1}^{n-1} \mathbb{P}(F_r)\right](2^{n-1} + 2)^{-1}$ for $n \leq n_0$ and $\mathbb{P}(F_n) = 0$ for $n > n_0$. Using strong induction one can show that this implies that $\mathbb{P}(F_n) = 3^{-1} \cdot 2^{-(n-1)}$ for $n \leq n_0$ and $\mathbb{P}(F_n) = 0$ for $n > n_0$.

We have argued that n_0 is finite – this implies that the algorithm halts with probability 1. Because the algorithm is correct if it outputs an x within 10^{-K} of $4 \cdot 10^{-K} e_2$ for LP and ℓ^1 problems or $4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3$ for the constrained lasso case, as well as $OutputEta^{\mathcal{O}_{\mathrm{vec}}, \mathcal{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ in the basis pursuit with TV case or $OutputPsi^{\mathcal{O}_{\mathrm{vec}}, \mathcal{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ for the unconstrained lasso with TV case (and we have already argued that $Weak$ is never executed) we conclude that the only possible incorrect outputs are an x within 10^{-K} of $4 \cdot 10^{-K} e_1$ (in the linear programming or ℓ^1 cases) or $4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3$ (in the constrained lasso case), or $x = P_{\mathrm{flip}} OutputEta^{\mathcal{O}_{\mathrm{vec}}, \mathcal{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ (in the basis pursuit TV case), respectively $x = P_{\mathrm{flip}} OutputPsi^{\mathcal{O}_{\mathrm{vec}}, \mathcal{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ (in the unconstrained lasso TV case). Each of these can occur only if the subroutine $Guess$ returns 'true'. Since $Guess$ returns 'true' with probability $1/2$, an incorrect output occurs with probability $\sum_{n=1}^{\infty} \mathbb{P}(F_n)/2 \leq \sum_{n=1}^{\infty} 3^{-1} \cdot 2^{-n} = 1/3$. Thus, with probability at least $1 - 1/3 = 2/3$, the algorithm produces a correct output.

Case 2: For this case, ι is of the following form: $\left(y^{\mathrm{LP},\mathrm{s}}, A(1/2, 1/2, m, N)\right)$ for linear programming, $\left(y^{\mathrm{BP},\ell^1,\mathrm{s}}, L(1/2, 1/2, m, N)\right)$ for basis pursuit denoising and $\left(y^{\mathrm{UL},\ell^1,\mathrm{s}}, L(1/2, 1/2, m, N)\right)$ for unconstrained lasso, $\left(y^{\mathrm{CL},\mathrm{s}}, L(1/2, 1/2, m, N)\right)$ for constrained lasso, $\left(y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2, m, N)\right)$ for basis pursuit denoising with TV regularisation and $\left(y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2, m, N)\right)$ for unconstrained lasso with TV regularisation.

As in the proof of Case 1, and again by Lemma 13.4, the algorithm proceeds to execute the loop in step 2. This time, however, the algorithm never terminates at either step 2a or step 2b – indeed, in the n-th iteration we have $d \leq 1/2 + 2^{-n} - (1/2 - 2^{-n}) = 2 \cdot 2^{-n}$ and similarly $d \geq 1/2 - 2^{-n} - (1/2 + 2^{-n}) = -2 \cdot 2^{-n}$. Thus the algorithm only terminates in the n-th iteration if the subroutine $Guess$ is executed. As in the proof for Case 1, we let F_n be the event that the subroutine $Guess$ is executed in the n-th iteration. The same argument as before shows that $\mathbb{P}(F_n) = 3^{-1} \cdot 2^{-(n-1)}$ (and this occurs for all n since step 2a and step 2b never result in the termination of the algorithm). Hence the probability the algorithm terminates with the execution of the subroutine $Guess$ is given by $\mathbb{P}\left(\bigcup_{r=1}^{\infty} F_r\right) = \sum_{r=1}^{\infty} \mathbb{P}(F_r) = \sum_{r=1}^{\infty} 3^{-1} \cdot 2^{-n+1} = 2/3$.

Note that, whenever $Guess$ is executed by the algorithm, then, for each of linear programming, basis pursuit with ℓ^1 regularisation and unconstrained lasso with ℓ^1 regularisation, the output of the algorithm is an $x \in \mathbb{D}^N$ within 10^{-K} of either $4 \cdot 10^{-K} e_1$ or $4 \cdot 10^{-K} e_2$, which, by Lemma 11.1, Lemma 11.3, or Lemma 11.2, satisfies $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K}$. Similarly, for constrained lasso the output of the algorithm is an $x \in \mathbb{D}^{-K}$ within 10^{-K} of either $4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3$ or $4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3$ and this time Lemma 11.4 implies that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K}$. The situation is only marginally more complicated for basis pursuit with TV regularisation – indeed, Lemma 11.5 shows that $\eta(y_1, 1/2, 1/2), P_{\mathrm{flip}} \eta(y_1, 1/2, 1/2) \in \Xi(\iota)$ and thus, writing x for the output of

Guess, we have by Lemma 13.1 that $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$. Finally, for unconstrained lasso with TV regularisation Lemma 11.6 shows that $\psi(y_1, 1/2, 1/2), P_{\text{flip}}\psi(y_1, 1/2, 1/2) \in \Xi(\iota)$ and thus, writing x for the output of *Guess*, Lemma 13.2 implies that $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$.

We conclude that, with probability $2/3$, the algorithm produces a correct output.

Case 3: For this case, ι is of the following form: $\big(y^{\text{LP,s}}, A(1/2, \beta, m, N)\big)$ for linear programming, $\big(y^{\text{BP},\ell^1,\text{s}}, L(1/2, \beta, m, N)\big)$ for basis pursuit denoising with ℓ^1 regularisation, $\big(y^{\text{UL},\ell^1,\text{s}}, L(1/2, \beta, m, N)\big)$ for unconstrained lasso, $\big(y^{\text{CL,s}}, L(1/2, \beta, m, N)\big)$ for constrained lasso, $\big(y^{\text{BP,TV,s}}, T(1/2, \beta, m, N)\big)$ for basis pursuit denoising with TV regularisation and $\big(y^{\text{UL,TV,s}}, T(1/2, \beta, m, N)\big)$ for unconstrained lasso with TV regularisation, where $1/4 < \beta < 1/2$ in the ℓ^1 case, $r_n < \beta < 1/2$ in the basis pursuit with TV case and $s_n < \beta < 1/2$ in the unconstrained lasso with TV case.

In this case, the argument for correctness proceeds as in Case 1, with the only exceptions being that now step 2b never results in the termination of the algorithm (as opposed to Case 1 where step 2a never results in the termination of the algorithm) and that now the correct solution is $4 \cdot 10^{-K} e_1$ for linear programming and basis pursuit with ℓ^1 regularisation, $4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3$ for constrained lasso, $P_{\text{flip}}\eta(y_1, 1/2, \beta)$ for basis pursuit with TV, or $P_{\text{flip}}\psi(y_1, 1/2, \beta)$ for unconstrained lasso with TV. Similarly to the argument for Case 1, the algorithm will only provide the wrong answer or fail to halt if *Guess* is called and outputs an $x \in \mathbb{D}^N$ within 10^{-K} of $4 \cdot 10^{-K} e_2$ for linear programming and basis pursuit with ℓ^1 regularisation, $4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3$ for constrained lasso, $\eta(y_1, 1/2, \beta)$ for basis pursuit with TV, or $\psi(y_1, 1/2, \beta)$ for unconstrained lasso with TV. This occurs with probability bounded above by $\sum_{n=1}^{\infty} \mathbb{P}(F_n)/2 = 1/3$ and thus the algorithm produces a correct output with probability at least $2/3$.

Case 4: For this case, ι is of the following form: $\big(y^{\text{LP,w}}, A(\alpha, \beta, m, N)\big)$ for linear programming, $\big(y^{\text{BP},\ell^1,\text{w}}, L(\alpha, \beta, m, N)\big)$ for basis pursuit denoising with ℓ^1 regularisation, $\big(y^{\text{UL},\ell^1,\text{w}}, L(\alpha, \beta, m, N)\big)$ for unconstrained lasso, $\big(y^{\text{CL,w}}, L(\alpha, \beta, m, N)\big)$ for unconstrained lasso, $\big(y^{\text{BP,TV,w}}, T(\alpha, \beta, m, N)\big)$ for basis pursuit denoising with TV regularisation and $\big(y^{\text{UL,TV,w}}, T(\alpha, \beta, m, N)\big)$ for unconstrained lasso with TV regularisation, where $(\alpha, \beta) \in \mathcal{L}$ in the ℓ^1 case, $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},K-1}$ in the basis pursuit with TV case and $(\alpha, \beta) \in \mathcal{L}^{\text{UL,TV},K-1}$ in the unconstrained lasso with TV case.

By the correctness of *IdenifyStrongOrWeak* (Lemma 13.4), the algorithm proceeds immediately by executing $Weak^{\mathcal{O}_{\text{vec}}, \mathcal{O}_{\text{mat}}}(m, N, k_\epsilon = K)$. The correctness of *Weak* (Lemma 13.3) then shows that the algorithm always outputs some x with $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$. Therefore, in this case the algorithm is correct with probability 1.

These are the only possible cases for ι, and thus, with probability at least $2/3$, the algorithm halts and outputs an x such that $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$, as desired.

15. PROOF OF THEOREM 3.4: PARTS (III) AND (IV)

Parts (iii) and (iv) of Theorem 3.4 are formally stated in Proposition 9.33, which we now prove. Recall that the breakdown epsilon bounds in Proposition 9.33 were already proved in §14.1. Thus, all that remains to prove in part (iii) is the existence of a deterministic algorithm that achieves at least $10^{-(K-1)}$ accuracy on the problems $\{\tilde{\Xi}, \tilde{\Omega}_{m,N}, \mathcal{M}_N, \tilde{\Lambda}_{m,N}\} = \{\Xi, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}^{\Delta_1}$ for varying dimensions m and N, where $\Omega_{m,N}$ is one of (12.6) and Ξ and $\Lambda_{m,N}$ are the corresponding solution map and set of evaluations.

15.1. Proof of part (iii) of Proposition 9.33. As in the previous two sections, we fix the notation for an element $\tilde{\iota}$ of $\tilde{\Omega}_{m,N}$, writing $\tilde{\iota} = \big(\{y_j^{(n)}\}_{n=0}^{\infty}, \{A_{j,k}^{(n)}\}_{n=0}^{\infty}\big)_{j,k}$, corresponding to an $\iota = (y, A) \in \Omega_{m,N}$, in the linear programming case, and analogously for the ℓ^1 and TV problems, writing respectively L and T instead of A. The desired algorithm makes use of the subroutines established in 13 and is specified for each of the computational problems as follows:

Deterministic $K - 1$ digit algorithm

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\text{vec}}$ and $\mathscr{O}_{\text{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$) of an input $\tilde{\iota}$.

Output: A vector $x \in \mathbb{D}^N$ (in the Turing case) or $x \in \mathbb{R}^N$ (in the BSS case) with $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+1}$.

1. We execute *IdentifyStrongOrWeak*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N)$. If this evaluates to 'InputWeak', we execute the algorithm *Weak*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_\epsilon = K - 1)$ and terminate. Otherwise, we continue to step 2.

2. For each of linear programming, basis pursuit with ℓ^1 regularisation and unconstrained lasso with ℓ^1 regularisation we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K} e_1 + 2 \cdot 10^{-K} e_2\|_p \leq 10^{-K}$ and terminate. For constrained lasso we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K} e_1 + 2 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \leq 10^{-K}$. For basis pursuit with TV regularisation we set $\eta^* = $ *OutputEta*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ and output $x = (P_{\text{flip}} \eta^* + \eta^*)/2$ and terminate and finally for unconstrained lasso with TV regularisation we set $\psi^* = $ *OutputPsi*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K, k_\epsilon = K)$ and output $x = (P_{\text{flip}} \psi^* + \psi^*)/2$.

Unlike the algorithm described in the proof of part (ii) of Proposition 9.32, it is clear that this algorithm always terminates. Hence it suffices to show that if x is the output of the algorithm then $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+1}$. We do this by separately considering the following two cases for $\iota \in \Omega$ that $\tilde{\iota}$ corresponds to.

Case 1: For this case, ι is of the following form: $\left(y^{\text{LP,s}}, A(\alpha, \beta, m, N)\right)$ for linear programming, $\left(y^{\text{BP},\ell^1,\text{s}}, L(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with ℓ^1 regularisation, $\left(y^{\text{UL},\ell^1,\text{s}}, L(\alpha, \beta, m, N)\right)$ for unconstrained lasso, $\left(y^{\text{CL,s}}, L(\alpha, 1/2, m, N)\right)$ for constrained lasso, $\left(y^{\text{BP,TV,s}}, T(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with TV regularisation and $\left(y^{\text{UL,TV,s}}, T(\alpha, \beta, m, N)\right)$ for unconstrained lasso with TV regularisation, where $(\alpha, \beta) \in \mathcal{L}$ for the ℓ^1 problems, $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},K}$ for basis pursuit with TV, and $(\alpha, \beta) \in \mathcal{L}^{\text{UL,TV},K}$ for unconstrained lasso with TV. By the correctness of *IdentifyStrongOrWeak* (Lemma 13.4), in this case the algorithm will execute step 2.

By Lemma 11.1 (respectively Lemma 11.3 or Lemma 11.2) we have $\Xi(\iota) \cap \{4 \cdot 10^{-K} e_1, 4 \cdot 10^{-K} e_2\} \neq \varnothing$ for linear programming (respectively, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation). Thus for any of linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation we conclude that $\text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \max\{d_{\mathcal{M}}(x, 4 \cdot 10^{-K} e_1), d_{\mathcal{M}}(x, 4 \cdot 10^{-K} e_2)\} \leq \|2 \cdot 10^{-K} e_1 + 2 \cdot 10^{-K} e_2\|_p + 2 \cdot 10^{-K} = (2 + 2^{1+1/p}) 10^{-K} \leq 10^{-K+1}$. The same argument but this time with Lemma 11.4 and the addition of $(\tau - 4 \cdot 10^{-K}) e_3$ where appropriate shows that $\text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K+1}$ for constrained lasso.

Similarly, Lemma 11.5 shows that $\Xi(\iota) \cap \{\eta(y_1, \alpha, \beta), P_{\text{flip}} \eta(y_1, \alpha, \beta)\} \neq \varnothing$ for basis pursuit with TV regularisation. Thus if x is the output of the algorithm and $\eta = \eta(y_1, \alpha, \beta)$ we have

$$\text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \max\{d_{\mathcal{M}}(x, \eta), d_{\mathcal{M}}(x, P_{\text{flip}} \eta)\}$$

$$\leq \left\| \frac{\eta^* + P_{\text{flip}} \eta^*}{2} - \frac{\eta - P_{\text{flip}} \eta}{2} \right\|_p + \max\left\{ \left\| \frac{\eta + P_{\text{flip}} \eta}{2} - \eta \right\|_p, \left\| \frac{\eta + P_{\text{flip}} \eta}{2} - P_{\text{flip}} \eta \right\|_p \right\}$$

$$\leq \|\eta^* - \eta\|_p + \frac{1}{2} \|\eta - P_{\text{flip}} \eta\|_p = \|\eta^* - \eta\|_p + \frac{1}{2} \|(\eta_1 - \eta_N)(e_1 - e_N)\|_p.$$

By Lemma 13.1 we obtain $\|\eta^* - \eta\|_p \leq 10^{-K} < 10^{-K+1}/6$. Moreover, $\|(\eta_1 - \eta_N)(e_1 - e_N)\|_p = 2^{1/p} |\eta_1 - \eta_N|$ and by applying Lemma 12.8 (equation 12.19) with $n = K$ we obtain $|\eta_1 - \eta_N| \leq \max_{(\alpha,\beta) \in \mathcal{L}^{\text{BP,TV},K}} |\eta(y_1, \alpha, \beta) - \eta(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-K+1} 2^{-1/p}/6$. We conclude that $\text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K+1}/6 + 5 \cdot 10^{-K+1}/6 = 10^{-K+1}$.

Finally, for unconstrained lasso with TV regularisation, applying Lemma 11.6 shows that $\Xi(\iota) \cap \{\psi(y_1, \alpha, \beta), P_{\text{flip}}\psi(y_1, \alpha, \beta)\} \neq \varnothing$. Therefore if x is the output of the algorithm and $\psi = \psi(y_1, \alpha, \beta)$ then an argument similar to the above establishes

$$\text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \max\{d_{\mathcal{M}}(x, \psi), d_{\mathcal{M}}(x, P_{\text{flip}}\psi)\} \leq \|\psi^* - \psi\|_p + \frac{1}{2}\|(\psi_1 - \psi_N)(e_1 - e_N)\|_p.$$

This time, by Lemma 13.2 we obtain $\|\psi^* - \psi\|_p \leq 10^{-K} < 10^{-K+1}/6$. Moreover, $\|(\psi_1 - \psi_N)(e_1 - e_N)\|_p = 2^{1/p}|\psi_1 - \psi_N|$ and by Lemma 12.9 (equation 12.23) with $n = K$ we obtain $|\psi_1 - \psi_N| \leq \max_{(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},K}} |\psi(y_1, \alpha, \beta) - \psi(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-K+1}2^{-1/p}/6$. We therefore conclude that $\text{dist}_{\mathcal{M}}(x, \Xi(\iota))) \leq 5 \cdot 10^{-K+1}/6 + 10^{-K+1}/6 = 10^{-K+1}$.

We have thus established that for any of linear programming, basis pursuit or unconstrained lasso (with either ℓ^1 or with TV regularisation), the output x satisfies $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K+1}$.

Case 2: For this case, ι is of the following form: $(y^{\text{LP,w}}, A(\alpha, \beta, m, N))$ for linear programming, $(y^{\text{BP},\ell^1,\text{w}}, L(\alpha, \beta, m, N))$ for basis pursuit denoising with ℓ^1 regularisation, $(y^{\text{UL},\ell^1,\text{w}}, L(\alpha, \beta, m, N))$ for unconstrained lasso, $(y^{\text{CL,w}}, L(\alpha, \beta, m, N))$ for constrained lasso, $(y^{\text{BP,TV,w}}, T(\alpha, \beta, m, N))$ for basis pursuit denoising with TV regularisation and $(y^{\text{UL,TV,w}}, T(\alpha, \beta, m, N))$ for unconstrained lasso with TV regularisation, where $(\alpha, \beta) \in \mathcal{L}$ for linear programming and the ℓ^1 regularised problems, $(\alpha, \beta) \in \mathcal{L}^{\text{BP,TV},K-1}$ for basis pursuit with TV, and $(\alpha, \beta) \in \mathcal{L}^{\text{UL,TV},K-1}$ for unconstrained lasso with TV.

By the correctness of *IdenifyStrongOrWeak* (Lemma 13.4), the algorithm proceeds immediately by executing *Weak*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_\epsilon = K - 1)$. The correctness of *Weak* (Lemma 13.3) then shows that the algorithm always outputs some x with $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \text{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K+1}$.

These are the only possible cases for each of the inputs, and we can thus conclude that the algorithm halts and outputs an $x \in \mathbb{R}^N$ such that $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+1}$, completing the proof of Proposition 9.33 part (iii).

15.2. **Proof of part (iv) of Proposition 9.33.**

The desired algorithm that achieves $10^{-(K-2)}$ accuracy and has polynomial Turing arithmetic and BSS runtime, as well as Turing space complexity, and requires at most a polynomial in $\log(n_{\text{var}})$ digits form the oracles, where $n_{\text{var}} = mN + m$ is the number of variables, is specified as follows:

Polynomial time $K - 2$ digit algorithm

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\text{vec}}$ and $\mathscr{O}_{\text{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$) of an input $\tilde{\iota}$.

Output: A vector $x \in \mathbb{D}^N$ (in the Turing case) or $x \in \mathbb{R}^N$ (in the BSS case) with $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+2}$.

1. We execute *IdentifyStrongOrWeak*$^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N)$ and branch depending on the result:
   a. If the output of *IdentifyStrongOrWeak* was 'InputWeak' then we output depending on the problem at hand. For each of linear programming, basis pursuit with ℓ^1 regularisation and unconstrained lasso with ℓ^1 regularisation we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K+1}e_1 + 2 \cdot 10^{-K+1}e_2\|_p \leq 10^{-(K-1)}$ and terminate. For constrained lasso we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K+1}e_1 + 2 \cdot 10^{-K+1}e_2 + (\tau - 4 \cdot 10^{-K+1})e_3\|_p \leq 10^{-(K-1)}$ and terminate. For basis pursuit with TV regularisation we set $\eta^* = \textit{OutputEta}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K - 1, k_\epsilon = K - 1)$ and output $x = (P_{\text{flip}}\eta^* + \eta^*)/2$ and terminate and finally for unconstrained lasso with TV regularisation we set $\psi^* = \textit{OutputPsi}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_0 = K - 1, k_\epsilon = K - 1)$ and output $x = (P_{\text{flip}}\psi^* + \psi^*)/2$ and terminate.

b. If the output of *IdentifyStrongOrWeak* was 'InputStrong' then we output depending on the problem at hand. For each of linear programming, basis pursuit with ℓ^1 regularisation and unconstrained lasso with ℓ^1 regularisation we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K}e_1 + 2 \cdot 10^{-K}e_2\|_p \leq 10^{-(K-1)}$ and terminate. For constrained lasso we output an $x \in \mathbb{D}^N$ with $\|x - 2 \cdot 10^{-K}e_1 + 2 \cdot 10^{-K}e_2 + (\tau - 4 \cdot 10^{-K})e_3\|_p \leq 10^{-(K-1)}$ and terminate. For basis pursuit with TV regularisation we set $\eta^* = OutputEta^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K - 1)$ and output $x = (P_{\mathrm{flip}}\eta^* + \eta^*)/2$ and terminate and finally for unconstrained lasso with TV regularisation we set $\psi^* = OutputPsi^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}(m, N, k_0 = K, k_\epsilon = K - 1)$ and output $x = (P_{\mathrm{flip}}\psi^* + \psi^*)/2$ and terminate.

We show that the algorithm outputs a vector x with $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+2}$ by analysing the following two cases for $\iota \in \Omega$ that $\tilde{\iota}$ corresponds to:

Case 1: For this case, ι is of the following form: $\left(y^{\mathrm{LP,s}}, A(\alpha, \beta, m, N)\right)$ for linear programming, $\left(y^{\mathrm{BP}, \ell^1, \mathrm{s}}, L(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with ℓ^1 regularisation, $\left(y^{\mathrm{UL}, \ell^1, \mathrm{s}}, L(\alpha, \beta, m, N)\right)$ for unconstrained lasso, $\left(y^{\mathrm{CL,s}}, L(\alpha, \beta, m, N)\right)$ for constrained lasso, $\left(y^{\mathrm{BP,TV,s}}, T(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with TV regularisation and $\left(y^{\mathrm{UL,TV,s}}, T(\alpha, \beta, m, N)\right)$ for unconstrained lasso with TV regularisation, where $(\alpha, \beta) \in \mathcal{L}$ in the ℓ^1 case, $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV},K}$ in the basis pursuit with TV case and $(\alpha, \beta) \in \mathcal{L}^{\mathrm{UL,TV},K}$ in the unconstrained lasso with TV case. By the correctness of *IdentifyStrongOrWeak* (Lemma 13.4), in this case the algorithm will execute step 1b).

We have already shown in the proof of part (iii) that this will result in an output x with $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K+1}$. It thus remains to analyse the complexity of the algorithm. To this end, first note that Lemmas 13.1, 13.2, and 13.4 imply that *OutputEta*, *OutputPsi*, and *IdentifyStrongOrWeak* request at most a polynomial in $\log(n_{\mathrm{var}})$ digits from the oracles, and therefore so does the algorithm overall. Next, it suffices to show that the Turing and BSS runtime of the algorithm is polynomial in n_{var}, as this will then imply the desired polynomial bound on the Turing arithmetic runtime and the Turing space complexity as well. Note that Lemma 13.4 implies that the Turing and BSS runtime of executing *IdentifyStrongOrWeak* is bounded by a polynomial of $\log(m)$, whereas Lemmas 13.1 and 13.2 imply that the Turing and BSS runtimes of executing *OutputEta* and *OutputPsi* are bounded by a polynomial of K and $\log(N)$. This, in particular, means that $\mathrm{Len}(\eta^*)$ and $\mathrm{Len}(\psi^*)$, and hence the Turing runtime of computing x, are bounded by a polynomial of $\log(N)$. As K is fixed, we deduce that the the overall Turing and BSS runtime of the algorithm is bounded by a polynomial of $\log(m)$ and $\log(N)$, which is itself bounded by a polynomial of n_{var}, as desired.

Case 2: For this case, ι is of the following form: $\left(y^{\mathrm{LP,w}}, A(\alpha, \beta, m, N)\right)$ for linear programming, $\left(y^{\mathrm{BP}, \ell^1, \mathrm{w}}, L(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with ℓ^1 regularisation, $\left(y^{\mathrm{UL}, \ell^1, \mathrm{w}}, L(\alpha, \beta, m, N)\right)$ for unconstrained lasso, $\left(y^{\mathrm{CL,w}}, L(\alpha, \beta, m, N)\right)$ for unconstrained lasso, $\left(y^{\mathrm{BP,TV,w}}, T(\alpha, \beta, m, N)\right)$ for basis pursuit denoising with TV regularisation and $\left(y^{\mathrm{UL,TV,w}}, T(\alpha, \beta, m, N)\right)$ for unconstrained lasso with TV regularisation, where $(\alpha, \beta) \in \mathcal{L}$ in the ℓ^1 case, $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV},K-1}$ in the basis pursuit with TV case and $(\alpha, \beta) \in \mathcal{L}^{\mathrm{UL,TV},K-1}$ in the unconstrained lasso with TV case. In this case, the correctness of *IdentifyStrongOrWeak* (Lemma 13.4) implies that the algorithm will execute step 1a). Our analysis is very similar to that of part (iii) Case 1.

By Lemma 11.1 (respectively Lemma 11.3 or Lemma 11.2) we have $\Xi(\iota) \cap \{4 \cdot 10^{-K+1}e_1, 4 \cdot 10^{-K+1}e_2\} \neq \varnothing$ for linear programming (respectively, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation). Thus for any of linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation we conclude that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \max\{d_{\mathcal{M}}(x, 4 \cdot 10^{-K+1}e_1), d_{\mathcal{M}}(x, 4 \cdot 10^{-K+1}e_2)\} \leq \|2 \cdot 10^{-K+1}e_1 + 2 \cdot 10^{-K+1}e_2\|_p + 2 \cdot 10^{-(K-1)} = (2 + 2^{1+1/p})10^{-K+1} \leq 10^{-K+2}$.

The argument for constrained lasso is identical but now we add $(\tau - 4 \cdot 10^{-K+1})e_3$ to both the true solution and the output value of the algorithm. Applying Lemma 11.4 allows us to conclude that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-K+2}$.

Similarly, Lemma 11.5 shows that $\Xi(\iota) \cap \{\eta(y_1, \alpha, \beta), P_{\mathrm{flip}}\eta(y_1, \alpha, \beta)\} \neq \varnothing$ for basis pursuit with TV regularisation. Thus, writing $\eta = \eta(y_1, \alpha, \beta)$, we obtain as in the proof of part (iii), case 1 that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \|\eta^* - \eta\|_p + |\eta_1 - \eta_N|2^{1/p}/2$.

Next, by Lemma 13.1 we obtain $\|\eta^* - \eta\|_p \leq 10^{-(K-1)} < 10^{-K+2}/6$. Moreover, Lemma 12.8 (equation 12.19) with $n = K - 1$ gives $|\eta_1 - \eta_N| \leq \max_{(\alpha,\beta) \in \mathcal{L}^{\mathrm{BP,TV},K-1}} |\eta(y_1, \alpha, \beta) - \eta(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-K+2}2^{-1/p}/6$. We conclude that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 5 \cdot 10^{-K+2}/6 + 10^{-K+2}/6 = 10^{-K+2}$.

Finally, for unconstrained lasso with TV regularisation Lemma 11.6 shows that $\Xi(\iota) \cap \{\psi(y_1, \alpha, \beta), P_{\mathrm{flip}}\psi(y_1, \alpha, \beta)\} \neq \varnothing$. Therefore, writing $\psi = \psi(y_1, \alpha, \beta)$, we obtain as before that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq \|\psi^* - \psi\|_p + |\psi_1 - \psi_N|2^{1/p}/2$ Thus applying Lemma 12.9 (equation 12.23) with $n = K - 1$ gives $|\psi_1 - \psi_N| \leq \max_{(\alpha,\beta) \in \mathcal{L}^{\mathrm{BP,TV},K-1}} |\psi(y_1, \alpha, \beta) - \psi(y_1, \alpha, \beta)_N| \leq 5 \cdot 10^{-K+2}2^{-1/p}/6$. We conclude that $\mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 5 \cdot 10^{-K+2}/6 + 10^{-K+2}/6 = 10^{-K+2}$.

Therefore, in all cases we have $\mathrm{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) = \mathrm{dist}_{\mathcal{M}}(x, \Xi(\iota)) \leq 10^{-(K-2)}$. Finally, the complexity analysis is entirely analogous to the one presented in Case 1. As these were the only possible cases for ι, the proof of part (iv) of Proposition 9.33 is complete.

16. Proof of Theorem 5.1

Theorem 5.1 is formally stated in Proposition 9.41, which we now prove.

16.1. Constructing the sets of inputs.
The constructions here will be similar to those in the proof of Theorem 3.4. As before, each computational problem will require a separate input set. We will denote the input sets for LP, ℓ^1 BP, ℓ^1 UL, CL, TV BP and TV UL by

$$\Omega^{\mathrm{LP,E}}, \Omega^{\mathrm{BP},\ell^1,\mathrm{E}}, \Omega^{\mathrm{UL},\ell^1,\mathrm{E}}, \Omega^{\mathrm{CL},\ell^1,\mathrm{E}}, \Omega^{\mathrm{BP,TV,E}}, \Omega^{\mathrm{UL,TV,E}} \tag{16.1}$$

respectively. These input sets may depend on K, ω, ω, the dimensions m, N, and any relevant regularisation parameters. As in the proof of Theorem 3.4, we will omit this dependency in order to avoid cluttered notation.

For the ℓ^1 problems, there is no dependence on ω or ω and we simply set

$$\Omega^{\mathrm{LP,E}} = \Omega^{\mathrm{LP,s}}_{m,N,K}, \ \Omega^{\mathrm{BP},\ell^1,\mathrm{E}} = \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,K}, \ \Omega^{\mathrm{UL},\ell^1,\mathrm{E}} = \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,K}, \ \Omega^{\mathrm{CL},\ell^1,\mathrm{E}} = \Omega^{\mathrm{CL},\ell^1,\mathrm{s}}_{m,N,K}$$

where the sets $\Omega^{\mathrm{LP,s}}_{m,N,K}, \Omega^{\mathrm{BP},\ell^1,\mathrm{s}}_{m,N,K}, \Omega^{\mathrm{UL},\ell^1,\mathrm{s}}_{m,N,K}$ and $\Omega^{\mathrm{CL},\ell^1,\mathrm{s}}_{m,N,K}$ are defined as in §12.2.

The situation is slightly more complicated for the TV problems: starting with basis pursuit TV, we recall η from §11.3 and r_K as defined in §12.2, and choose $r'_K \in (r_K, 1/2)$ so that if $\alpha \in [r'_K, r_k]$ and $y_1 \in [0, 3/2]$, then $\|\eta(y_1, \alpha, 1/2) - \eta(y_1, 1/2, 1/2)\|_p \leq 10^{-K} - \omega$. The existence of r'_K is guaranteed by the continuity of η and the assumption that $10^{-K} - \omega > 0$.

For unconstrained lasso TV we recall ψ from §11.3 and s_K as defined in §12.2, and choose $s'_K \in (s_K, 1/2)$ so that if $\alpha \in [r'_K, r_k]$ and $y_1 \in [0, 107/180]$ then $\|\psi(y_1, \alpha, 1/2) - \psi(y_1, 1/2, 1/2)\|_p \leq 10^{-K} - \omega$. Again, the existence of s'_K is guaranteed by the continuity of ψ and the assumption that $10^{-K} - \omega > 0$.

We then set

$$\mathcal{L}^{\mathrm{BP,TV,E},K} = ([r'_K, 1/2] \times \{1/2\}) \cup (\{1/2\} \times [r'_K, 1/2])$$

$$\mathcal{L}^{\mathrm{UL,TV,E},K} = ([s'_K, 1/2] \times \{1/2\}) \cup (\{1/2\} \times [s'_K, 1/2]).$$

and

$$\Omega^{\mathrm{BP,TV,E}} = \left\{ \left(y^{\mathrm{BP,TV,s}}, T(\alpha, \beta, m, N)\right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{BP,TV,E},K} \right\},$$

$$\Omega^{\mathrm{UL,TV,E}} = \left\{ \left(y^{\mathrm{UL,TV,s}}, T(\alpha, \beta, m, N)\right) \mid (\alpha, \beta) \in \mathcal{L}^{\mathrm{UL,TV,E},K} \right\}$$

Note then that $\Omega^{\mathrm{BP,TV,E}} \subseteq \Omega^{\mathrm{BP,TV,s}}_{m,N,K}$ and $\Omega^{\mathrm{UL,TV,E}} \subseteq \Omega^{\mathrm{UL,TV,s}}_{m,N,K}$ so once again the statements about the condition and size of the inputs follows from Lemmas 12.1, 12.2, 12.6, and 12.7.

16.2. **Proof of Proposition 9.41 part (i).** Note that part (i) is an immediate consequence of part (ii) as discussed in Remark 5.2. Therefore we will focus on proving part (ii). To do this we will employ Proposition 10.6 with $\kappa = 10^{-K}$. For each of the input sets Ω discussed above, we will construct an input $\iota_0 \in \Omega$, sequences of inputs $\{\iota_n^1\}_{n=1}^\infty$ and $\{\iota_n^2\}_{n=1}^\infty$ in Ω, subsets S^1 and S^2 of $\mathbb{R}^N$ and vectors $x^1, x^2 \in \mathbb{R}^N$ satisfying requirements (a) to (f) in Proposition 10.6 with $\kappa = 10^{-K}$. To do so, we present an argument similar to the one used in Lemma 12.10, covering each computational problem separately.

Case $\{\Xi_{\mathrm{LP}}, \Omega^{\mathrm{LP,E}}\}$: We recall A from (11.1) and $y^{\mathrm{LP,s}}$ from (12.4) and set

$$\iota_n^0 = (y^{\mathrm{LP,s}}, A(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{LP,s}}, A(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{LP,s}}, A(1/2 - 4^{-n}, 1/2, m, N)),$$

$$x^1 = 4 \cdot 10^{-K} e_1, \qquad x^2 = 4 \cdot 10^{-K} e_2, \qquad S^1 = \{x^1\}, \quad \text{and} \quad S^2 = \{x^2\},$$

from which Proposition 10.6 (d) immediately holds. Arguing as in Lemma 12.10 gives us Proposition 10.6 (a). Lemma 11.1 gives Proposition 10.6 (b). It is then obvious that Proposition 10.6 (c) holds. Now, by Lemma 11.1, we have that, for $\alpha \in [0, 1/2)$, $\Xi_{\mathrm{LP}}(y^{\mathrm{LP,s}}, A(\alpha, 1/2, m, N)) = \{x^2\}$ and similarly, for $\beta \in [0, 1/2)$, $\Xi_{\mathrm{LP}}(y^{\mathrm{LP,s}}, A(1/2, \beta, m, N)) = \{x^1\}$, and thus

$$\Xi_{\mathrm{LP}}(\Omega^{\mathrm{LP,E}}) \subseteq \{x^1\} \cup \{x^2\} \cup \Xi_{\mathrm{LP}}(\iota^0). \tag{16.2}$$

which implies 10.6 (e). Finally, to show 10.6 (f) we note that Lemma 11.1 implies $x^1, x^2 \in \Xi(\iota^0)$.

Case $\{\Xi_{\mathrm{BPDN}}, \Omega^{\mathrm{BP},\ell^1,\mathrm{E}}\}$: We recall $y^{\mathrm{BP},\ell^1,\mathrm{s}}$ from (12.4) and set

$$\iota^0 = (y^{\mathrm{BP},\ell^1,\mathrm{s}}, L(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{BP},\ell^1,\mathrm{s}}, L(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{BP},\ell^1,\mathrm{s}}, L(1/2 - 4^{-n}, 1/2, m, N)),$$

$$x^1 = 4 \cdot 10^{-K} e_1, \qquad x^2 = 4 \cdot 10^{-K} e_2, \qquad S^1 = \{x^1\}, \quad \text{and} \quad S^2 = \{x^2\}.$$

The remainder of the argument is identical to the LP case except we use Lemma 11.3 instead of Lemma 11.1 and obtain

$$\Xi_{\mathrm{BPDN}}(\Omega^{\mathrm{BP},\ell^1,\mathrm{E}}) \subseteq \{x^1\} \cup \{x^2\} \cup \Xi_{\mathrm{BPDN}}(\iota^0) \tag{16.3}$$

instead of (16.2).

Case $\{\Xi_{\mathrm{UL}}, \Omega^{\mathrm{UL},\ell^1,\mathrm{E}}\}$: We recall $y^{\mathrm{UL},\ell^1,\mathrm{s}}$ from (12.4) and set

$$\iota^0 = (y^{\mathrm{UL},\ell^1,\mathrm{s}}, L(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{UL},\ell^1,\mathrm{s}}, L(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{UL},\ell^1,\mathrm{s}}, L(1/2 - 4^{-n}, 1/2, m, N)),$$

$$x^1 = 4 \cdot 10^{-K} e_1, \qquad x^2 = 4 \cdot 10^{-K} e_2, \qquad S^1 = \{x^1\}, \quad \text{and} \quad S^2 = \{x^2\}.$$

The remainder of the argument is identical to the LP case except we use Lemma 11.2 instead of Lemma 11.1 and obtain

$$\Xi_{\mathrm{UL}}(\Omega^{\mathrm{UL},\ell^1,\mathrm{E}}) \subseteq \{x^1\} \cup \{x^2\} \cup \Xi_{\mathrm{UL}}(\iota^0) \tag{16.4}$$

instead of (16.2).

Case $\{\Xi_{\mathrm{CL}}, \Omega^{\mathrm{CL},\ell^1,\mathrm{E}}\}$: We recall $y^{\mathrm{CL},\mathrm{s}}$ from (12.4) and set

$$\iota_n^0 = (y^{\mathrm{CL},\mathrm{s}}, L(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{CL},\mathrm{s}}, L(1/2, 1/2 - 4^{-n}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{CL},\mathrm{s}}, L(1/2 - 4^{-n}, 1/2, m, N)),$$

$$x^1 = 4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3, \qquad x^2 = 4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3,$$

$$S^1 = \{x^1\}, \quad \text{and} \quad S^2 = \{x^2\},$$

from which Proposition 10.6 (d) immediately holds. Arguing as in Lemma 12.10 gives us Proposition 10.6 (a). Lemma 11.4 gives Proposition 10.6 (b). It is then obvious that Proposition 10.6 (c) holds. Next, by Lemma 11.4, we have that, for $\alpha \in [0, 1/2)$, $\Xi_{\mathrm{CL}}(y^{\mathrm{CL},\mathrm{s}}, L(\alpha, 1/2, m, N)) = \{x^2\}$ and similarly, for $\beta \in [0, 1/2)$, $\Xi_{\mathrm{CL}}(y^{\mathrm{CL},\mathrm{s}}, L(1/2, \beta, m, N)) = \{x^1\}$. We thus obtain

$$\Xi_{\mathrm{CL}}(\Omega^{\mathrm{LP},\mathrm{E}}) \subseteq \{x^1\} \cup \{x^2\} \cup \Xi_{\mathrm{CL}}(\iota^0) \tag{16.5}$$

which implies 10.6 (e). Finally, to show 10.6 (f) we note that Lemma 11.4 implies $x^1, x^2 \in \Xi(\iota^0)$.

Case $\{\Xi_{\mathrm{BPTV}}, \Omega^{\mathrm{BP},\mathrm{TV},\mathrm{E}}\}$: We recall T from (11.7) and $y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}$ from (12.4), and choose $n_0 \in \mathbb{N}$ so that $1/2 - 4^{-n_0} \in [r_K', 1/2]$. Now, writing $y_1 = y_1^{\mathrm{BP},\mathrm{TV},\mathrm{s}}$, we set

$$\iota^0 = (y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2 - 4^{-n-n_0}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(1/2 - 4^{-n-n_0}, 1/2, m, N)),$$

$$x^1 = P_{\mathrm{flip}} \eta(y_1, 1/2, 1/2), \qquad x^2 = \eta(y_1, 1/2, 1/2),$$

$$S^1 = \{P_{\mathrm{flip}} \eta(y_1, \alpha, 1/2) \mid \alpha \in [r_K', 1/2]\}, \quad \text{and} \quad S^2 = \{\eta(y_1, 1/2, \beta) \mid \beta \in [r_K', 1/2]\}.$$

By the choice of n_0 and the definition of r_K' we have that ι^0, ι_n^1 and ι_n^2 are all elements of $\Omega^{\mathrm{BP},\mathrm{TV},\mathrm{E}}$, and we also see immediately from the definition that Proposition 10.6 (d) holds. Next, by Lemma 11.5 we get Proposition 10.6 (b), whereas the continuity of η together with Lemma 11.5 yield Proposition 10.6 (c). The same argument as in the proof of Lemma 12.10 gives us Proposition 10.6 (a).

Now, by Lemma 12.1 we have that $y_1 \in [0, 3/2]$, and so, by Lemma 11.5 and the definition of r_K' we obtain that, for $\alpha \in [r_K', 1/2]$ and $\iota = (y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(\alpha, 1/2, m, N))$ we have $\|\Xi_{\mathrm{BPTV}}(\iota) - x^1\|_\infty = \|\eta(y_1, \alpha, 1/2) - \eta(y_1, 1/2, 1/2)\|_\infty \leq 10^{-K} - \omega$. Similarly, for $\beta \in [r_K', 1/2]$ and $\iota = (y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(1/2, \beta, m, N))$ we obtain $\|\Xi_{\mathrm{BPTV}}(\iota) - x^2\|_\infty \leq 10^{-K} - \omega$. Hence

$$\Xi_{\mathrm{BPTV}}(\Omega^{\mathrm{BP},\mathrm{TV},\mathrm{E}}) \subseteq \overline{\mathcal{B}}_{10^{-K} - \omega}(x^1) \cup \overline{\mathcal{B}}_{10^{-K} - \omega}(x^2) \cup \overline{\mathcal{B}}_{10^{-K} - \omega}(\Xi_{\mathrm{BPTV}}(\iota^0)) \tag{16.6}$$

We conclude that Proposition 10.6 (e) holds. Finally, to show 10.6 (f) we note that Lemma 11.5 implies $x^1, x^2 \in \Xi(\iota^0)$.

Case $\{\Xi_{\mathrm{ULTV}}, \Omega^{\mathrm{UL},\mathrm{TV},\mathrm{E}}\}$: We recall $y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}$ from (12.4) and choose $n_0 \in \mathbb{N}$ such that $1/2 - 4^{-n_0} \in [s_K', 1/2]$. Writing $y_1 = y_1^{\mathrm{UL},\mathrm{TV},\mathrm{s}}$, we set

$$\iota^0 = (y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2, m, N)),$$

$$\iota_n^1 = (y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}, T(1/2, 1/2 - 4^{-n-n_0}, m, N)), \qquad \iota_n^2 = (y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}, T(1/2 - 4^{-n-n_0}, 1/2, m, N)),$$

$$x^1 = P_{\mathrm{flip}} \psi(y_1, 1/2, 1/2), \qquad x^2 = \psi(y_1, 1/2, 1/2),$$

$$S^1 = \{P_{\mathrm{flip}} \psi(y_1, \alpha, 1/2) \mid \alpha \in [s_K', 1/2]\}, \quad \text{and} \quad S^2 = \{\psi(y_1, 1/2, \beta) \mid \beta \in [s_K', 1/2]\}.$$

The remainder of the argument is very similar to that in the case $\{\Xi_{\mathrm{BPTV}}, \Omega^{\mathrm{BP},\mathrm{TV},\mathrm{E}}\}$ with the major difference being that we replace η by ψ, r_K' by s_K', references to Lemma 11.5 by references to Lemma 11.6, $y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}$ by $y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}$, and the bounds $y_1 \in [0, 3/2]$ by $y_1 \in [0, 107/180]$. This then yields

$$\Xi_{\mathrm{ULTV}}(\Omega^{\mathrm{UL},\mathrm{TV},\mathrm{E}}) \subseteq \overline{\mathcal{B}}_{10^{-K} - \omega}(x^1) \cup \overline{\mathcal{B}}_{10^{-K} - \omega}(x^2) \cup \overline{\mathcal{B}}_{10^{-K} - \omega}(\Xi_{\mathrm{ULTV}}(\iota^0)) \tag{16.7}$$

instead of (16.6). As before, we are able to conclude that assumptions (a) to (f) in Proposition 10.6 hold.

Therefore, for each computational problem we have shown that assumptions (a) to (f) in Proposition 10.6 hold, and thus there exits a $\tilde{\Lambda}^+ \in \mathcal{L}^{\mathcal{O},\omega,\tilde{\Xi}}(\tilde{\Lambda})$ such that, for the computational problem $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}^+\}$, we have $\epsilon_{\mathbb{PB}}^s(p) \geq 1/2$, as desired, completing the proof of part (i) of Proposition 9.41.

16.3. **Proof of Proposition 9.41 part (ii).** The fact that for the oracle problem $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}^{\mathcal{O},\omega}$ with respect to $\{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}$ we have $\epsilon_B^s \geq 1/2$ follows directly from part (i) of the proposition. It thus suffices to show that the oracle problem $\{\tilde{\Xi}, \tilde{\Omega}, \mathcal{M}, \tilde{\Lambda}\}^{\mathcal{O},\omega}$ with respect to $\{\Xi^E, \tilde{\Omega}, \{0,1\}, \tilde{\Lambda}\}$ can be computed in the arithmetic model to within 10^{-K} accuracy. The first step of our recursive algorithm that achieves this will make use of the exit flag to determine if the input is a representation of ι^0 – if not, the algorithm proceeds by executing a loop similar to the algorithm *Weak* defined in §13.

To this end, for the linear programming case, we recall (9.25) and fix the notation for an element of $\tilde{\Omega}^{\mathcal{O}}$ by writing $\tilde{\iota} = \left(\{y_j^{(n)}\}_{n=0}^\infty, \{A_{j,k}^{(n)}\}_{n=0}^\infty\right)_{j,k} \oplus \Xi^E(\tilde{\iota})$, corresponding to an $\iota = (y, A) \in \Omega$ and the solution $\Xi^E(\tilde{\iota})$ to the exit flag problem, i.e., $\Xi^E(\tilde{\iota}) = 1$ if $\text{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$ and $\Xi^E(\tilde{\iota}) = 0$ else. For the ℓ^1 and TV problems the notation is entirely analogous, except that we respectively write L and T instead of A. The exact specification of the algorithm is now as follows:

Algorithm ComputeTrueSolution:

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\text{vec}}$, $\mathscr{O}_{\text{mat}}$, and $\mathscr{O}_{\text{sol}}$ providing access to the components $y_j^{(n)}$, $A_{j,k}^{(n)}$ (respectively $L_{j,k}^{(n)}$ or $T_{j,k}^{(n)}$), and $\Xi^E(\tilde{\iota})$ of an input $\tilde{\iota}$.

Output: A vector $x \in \mathbb{D}^N$ (for the Turing machine) or $x \in \mathbb{R}^N$ (for the BSS machine) with $\text{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$.

1. If $\Xi^E(\tilde{\iota}) = 1$, we run Γ on $\tilde{\iota}$ and output $\Gamma(\tilde{\iota})$. Otherwise, we proceed to the next step.

2. We execute a loop that proceeds as follows – at each iteration, we increase n, starting with $n = 1$. What we do now depends on the problem at hand. In the linear programming case we use the oracle $\mathscr{O}_{\text{mat}}$ to read the values $A_{1,1}^{(n)}$ and $A_{1,2}^{(n)}$ and set $d = A_{1,1}^{(n)} - A_{1,2}^{(n)}$. In the ℓ^1 case we use the oracle $\mathscr{O}_{\text{mat}}$ to read the values $L_{1,1}^{(n)}$ and $L_{1,1}^{(n)}$ and set $d = L_{1,1}^{(n)} - L_{1,2}^{(n)}$. For the TV problems we similarly read the values $T_{1,1}^{(n)}$ and $T_{1,N}^{(n)}$ and set $d = T_{1,1}^{(n)} - T_{1,N}^{(n)}$.

   Next, we branch depending on the value of d:

   a. If $d > 2 \cdot 2^{-n}$ then we output an $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_1\|_p \leq 10^{-K}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we output an $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_1 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \leq 10^{-K}$. For basis pursuit TV we apply the subroutine *OutputEta* to obtain $\eta^* = \textit{OutputEta}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_K = K, k_\epsilon = K)$, to which we apply P_{flip} and output as x (so that $x = P_{\text{flip}} \eta^*$). Finally, for unconstrained lasso TV we apply the subroutine *OutputPsi* to obtain $\psi^* = \textit{OutputPsi}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_K = K, k_\epsilon = K)$, to which we apply P_{flip} and output as x (so that $x = P_{\text{flip}} \psi^*$). In all of the above cases we terminate the algorithm after outputting x.

   b. Alternatively, if $d < -2 \cdot 2^{-n}$ then we output an $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2\|_p \leq 10^{-K}$ for linear programming, basis pursuit with ℓ^1 regularisation or unconstrained lasso with ℓ^1 regularisation. For constrained lasso we output an $x \in \mathbb{D}^N$ with $\|x - 4 \cdot 10^{-K} e_2 + (\tau - 4 \cdot 10^{-K}) e_3\|_p \leq 10^{-K}$. For basis pursuit TV we apply the subroutine *OutputEta* to obtain $\eta^* = \textit{OutputEta}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_K = K, k_\epsilon = K)$, which we output as x. Finally, for unconstrained lasso TV we apply the subroutine *OutputPsi* with parameter ϵ to obtain $\psi^* = \textit{OutputPsi}^{\mathscr{O}_{\text{vec}}, \mathscr{O}_{\text{mat}}}(m, N, k_K = K, k_\epsilon = K)$, which we output as x. In all of the above cases we terminate the algorithm after outputting x.

If neither of these conditions are met then the loop continues by executing the next iteration.

Lemma 16.1. *For every input $\tilde{\iota}$ in $\tilde{\Omega}$, ComputeTrueSolution outputs a vector x with $\operatorname{dist}_{\mathcal{M}}(x, \tilde{\Xi}(\tilde{\iota})) \leq 10^{-K}$.*

Proof. First, observe that if 'ComputeTrueSolution' exits at step 1 then 'ComputeTrueSolution' outputs a value within 10^{-K} of the true solution – this follows from the definition of the exit flag oracle.

Next, for each of the computational problems, let ι^0, x^1, and x^2 be as in the proof of part (i). Now, for any $\tilde{\iota}$ corresponding to ι^0, we claim that the algorithm always terminates at step 1. Indeed, we proved there that $\Xi(\Omega) \subseteq \overline{\mathcal{B}}_{10^{-K}-\omega}(x^1) \cup \overline{\mathcal{B}}_{10^{-K}-\omega}(x^2) \cup \Xi(\iota^0)$ and that $x^1, x^2 \in \Xi(\iota^0)$. Consequently, we have $\Xi(\Omega) \subseteq \overline{\mathcal{B}}_{10^{-K}-\omega}(\Xi(\iota^0))$. Furthermore, by assumption 9.22 we have $\operatorname{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \Xi(\Omega)) \leq \omega$. Hence

$$\operatorname{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) = \operatorname{dist}_{\mathcal{M}}(\Gamma(\tilde{\iota}), \Xi(\iota^0)) \leq \operatorname{dist}_{\mathcal{M}}(\Gamma(\iota), \Xi(\Omega)) + \sup_{w \in \Xi(\Omega)} \operatorname{dist}_{\mathcal{M}}(w, \tilde{\Xi}(\tilde{\iota}^0))$$

$$\leq \omega + 10^{-K} - \omega = 10^{-K},$$

completing the argument that the algorithm always exits at step 1.

All that remains is to prove that 'ComputeTrueSolution' is correct whenever $\tilde{\iota}$ corresponds to an element $\iota \in \Omega$ other than ι^0 as well as $\Xi^E(\tilde{\iota}) = 0$. The proof of this step is very similar to the proof of Lemma 13.3. Indeed, since $\iota \neq \iota^0$, ι has to be of one of the following forms for the ℓ^1 problems: $(y^{\mathrm{LP},\mathrm{s}}, L(\alpha, \beta, m, N))$ for linear programming, $(y^{\mathrm{BP},\ell^1,\mathrm{s}}, L(\alpha, \beta, m, N))$ for basis pursuit denoising with ℓ^1 regularisation and $(y^{\mathrm{UL},\ell^1,\mathrm{s}}, L(\alpha, \beta, m, N))$ for unconstrained lasso with ℓ^1 regularisation where $\alpha, \beta \in \mathcal{L}$, $\alpha \neq \beta$. Similarly, for the TV problems ι must be of one of the following forms: $(y^{\mathrm{BP},\mathrm{TV},\mathrm{s}}, T(\alpha, \beta, m, N))$ for basis pursuit TV where $\alpha \neq \beta$ and $(\alpha, \beta) \in \mathcal{L}^{\mathrm{BP},\mathrm{TV},\mathrm{E},K}$ and $(y^{\mathrm{UL},\mathrm{TV},\mathrm{s}}, T(\alpha, \beta, m, N))$ for unconstrained lasso TV where $\alpha \neq \beta$ and $(\alpha, \beta) \in \mathcal{L}^{\mathrm{UL},\mathrm{TV},\mathrm{E},K}$. The remainder of the proof from here is identical to that of Lemma 13.3, except that all references to the value 10^{-K+1} are now replaced by 10^{-K}. $\square$

The above construction establishes that, given an oracle for the exit flag, one can compute $\tilde{\Xi}$ to precision 10^{-K}, concluding the proof of part (ii).

16.4. **Proof of Proposition 9.41 part (iii).** We recall the definition of A^{E} from (11.13), define the input set $\Omega^\sharp$ according to

$$\Omega^\sharp := \left\{ (y^{\mathrm{L}}(y_1, m), A^{\mathrm{E}}(\alpha, m, N)) \mid y_1 = 3 \cdot 10^{-K} \cdot \alpha \quad \text{and} \quad \alpha \in [0, 1/2) \right\},$$

and let Ξ be either Ξ_{LP} or Ξ_{BPDN} with $\delta = 0$. We first construct a $\hat{\Lambda} \in \mathcal{L}^1(\Lambda)$ such that, for the computational problem $\{\Xi^E, \Omega^\sharp, \{0, 1\}, \hat{\Lambda}\}$ and $\mathrm{p} > 1/2$, we have $\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) \geq 1/2$ and second, we design an algorithm that outputs a solution to the oracle problem $\{\Xi^E, \tilde{\Omega}^\sharp, \{0, 1\}, \tilde{\Lambda}\}^{\mathcal{O},\omega}$ with respect to $\{\tilde{\Xi}, \tilde{\Omega}^\sharp, \mathcal{M}, \tilde{\Lambda}\}$.

To accomplish the first task, we will use Proposition 10.6 with $\kappa = 10^{-K}$. Concretely, we set

$$\iota^0 = (y^{\mathrm{L}}(0, m), A^{\mathrm{E}}(0, m, N)), \quad \iota_n^1 = (y^{\mathrm{L}}(3 \cdot 10^{-K} \cdot 2^{-n}, m), A^{\mathrm{E}}(2^{-n}, m, N)), \quad \iota_n^2 = \iota^0$$

$$x^1 = 3 \cdot 10^{-K} e_1 \in \mathbb{R}^N, \quad x^2 = 0 \in \mathbb{R}^N, \quad S^1 = \{x^1\}, \quad \text{and} \quad S^2 = \{x^2\}.$$

We now prove that conditions (a) to (e) of Proposition 10.6 hold. Starting with (a), we note that $\operatorname{dist}_{\mathcal{M}}(S^1, S^2) = d_{\mathcal{M}}(x^1, 0) = 3 \cdot 10^{-K}$. By Lemma 11.7, $\Xi(\iota_n^j) = \{x^j\}$ for all $n \in \mathbb{N}$ and $j = 1, 2$. This gives us both (b) and (c). We also have $\iota^0 = \iota_n^2$ and $|f(\iota_n^1) - f(\iota^0)| \leq \max(3 \cdot 10^{-K} 2^{-n}, 2^{-n}) = 2^{-n}$, for all $f \in \Lambda$, and thus (d) holds. Finally, for $\iota \in \Omega^\sharp$ we can use Lemma 11.7 to see that $\Xi(\iota) = \{x^1\}$ or $\Xi(\iota) = \{x^2\}$ and hence (e) holds. An application of Proposition 10.6 therefore establishes that $\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) \geq 1/2$, for $\mathrm{p} > 1/2$, as desired.

Next, we construct the desired algorithm, which will be identical for both basis pursuit and linear programming. Similarly to the proof of part (ii), we write $\tilde{\iota} = \left(\{y_j^{(n)}\}_{n=0}^\infty, \{A_{j,k}^{(n)}\}_{n=0}^\infty\right)_{j,k} \oplus \{g_k(\tilde{\iota})\}_k$, for an input $\tilde{\iota} \in \tilde{\Omega}$ corresponding to an $\iota = (y, A) \in \Omega$ and a solution oracle $\{g_k(\tilde{\iota})\}_k \in \mathcal{B}_\omega^\infty(\tilde{\Xi}(\tilde{\iota}))$. We now define:

Algorithm ComputeExitFlag:

Inputs: Dimensions m, N.

Oracles: $\mathcal{O}_{\mathrm{sol}}$ providing access to the components g_k of an input $\tilde{\iota}$

Output: Either 1 if $\mathrm{dist}_\mathcal{M}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) \le 10^{-K}$ or 0 if $\mathrm{dist}_\mathcal{M}(\Gamma(\tilde{\iota}), \tilde{\Xi}(\tilde{\iota})) > 10^{-K}$.

1. We use $\mathcal{O}_{\mathrm{sol}}$ to read $g_1(\tilde{\iota})$: if $|g_1(\tilde{\iota})| \le 10^{-K}$ then we set $x = 0 \in \mathbb{R}^N$, otherwise we set $x = 3 \cdot 10^{-K} e_1$.

2. We output 1 if $\|\Gamma(\tilde{\iota}) - x\|_\infty \le 10^{-K}$. Otherwise, we output 0.

To prove the correctness of 'ComputeExitFlag', we will prove that $\tilde{\Xi}(\tilde{\iota}) = \{x\}$ for every input $\tilde{\iota} \in \tilde{\Omega}^\sharp$. Indeed, let ι be the element of $\Omega^\sharp$ that $\tilde{\iota}$ corresponds to. There are two cases to consider: either $\iota = (y^{\mathrm{L}}(3 \cdot 10^{-K}\alpha, m), A^{\mathrm{E}}(\alpha, m, N))$ for some $\alpha > 0$ or $\iota = (y^{\mathrm{L}}(0, m), A^{\mathrm{E}}(0, m, N))$. In the first case Lemma 11.7 tells us that $\Xi(\iota) = \{3 \cdot 10^{-K}\}$. In particular, since $\{g_k(\tilde{\iota})\}_k \in \mathcal{B}_\omega^\infty(\tilde{\Xi}(\tilde{\iota}))$, we must have $|g_1(\tilde{\iota})| \ge 3 \cdot 10^{-K} - \omega > 10^{-K}$ and hence 'ComputeExitFlag' sets $x = 3 \cdot 10^{-K}$, which is the only element of $\Xi(\iota)$. If instead $\iota = (y^{\mathrm{L}}(0, m), A^{\mathrm{E}}(0, m, N))$ then Lemma 11.7 tells us that $\Xi(\iota) = 0$. Hence $|g_1(\tilde{\iota})| \le \omega \le 10^{-K}$. Thus 'ComputeExitFlag' sets $x = 0$, which is the only element of $\Xi(\iota) = \tilde{\Xi}(\tilde{\iota})$. We have thus shown that $\tilde{\Xi}(\tilde{\iota}) = \{x\}$. Therefore $\Gamma^E(\tilde{\iota}) = 1$ if $\|\Gamma(\tilde{\iota}) - x\|_\infty \le 10^{-K}$ and $\Gamma^E(\tilde{\iota}) = 0$ otherwise. But this is exactly the output of 'ComputeExitFlag' in step 2 and thus the algorithm 'ComputeExitFlag' computes the exit flag as claimed.

17. PROOF OF THEOREM 6.1

Theorem 6.1 is formally stated in Proposition 9.43, which we now prove. Our strategy is similar to that for the proof of Theorem 3.4. We begin by defining the input set Ω and then verify the conditions of (10.5). This, together with an additional algorithm we write to prove part (iii) will complete the proof of Proposition 9.43.

17.1. **Constructing the input and evaluation sets.** Let M and K be as in the statement in the proposition, recall the definition of $\mathcal{S}_k^9$ from (9.28) and define $M_k = 10^{-k}(\lfloor 10^k M \rfloor + 1)$, for $k \in \mathbb{N} \cup \{0\}$. Note that $M_k \le M_{k-1}$, for all $k \in \mathbb{N}$, with equality if and only if $M \in \mathcal{S}_k^9$.

We recall the definitions of y^{L} and $A^{\mathrm{LP,D}}$ from (11.14) and for natural m and N with $m < N$ define the sets $\Omega_{m,N}^{\mathrm{LP,s}}$ and $\Omega_{m,N}^{\mathrm{LP,w}}$ as follows:

$$\Omega_{m,N}^{\mathrm{LP,s}} = \{(y^{\mathrm{L}}(M_K\alpha, m), A^{\mathrm{LP,D}}(\alpha, 1, m, N)) \mid \alpha \in [0,1]\}$$

$$\Omega_{m,N}^{\mathrm{LP,w}} = \{(y^{\mathrm{L}}(M_{K-1}\alpha, m), A^{\mathrm{LP,D}}(\alpha, 0, m, N)) \mid \alpha \in (0,1]\}$$
$$\cup \{(y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(-\alpha, 0, m, N)) \mid \alpha \in (0,1]\}$$

and set $\Omega = \bigcup_{1 \le m < N} \Omega_{m,N}^{\mathrm{LP,s}} \cup \Omega_{m,N}^{\mathrm{LP,w}}$.

17.2. **Proof of Proposition 9.43 parts (i) and (ii).** Fix $m, N \in \mathbb{N}$ with $m < N$ and define the constant sequence $\{\iota_n^1\}_{n=1}^\infty \subseteq \Omega_{m,N}^{\mathrm{LP,s}}$ and input $\iota^0 \in \Omega_{m,N}^{\mathrm{LP,s}}$ by $\iota_n^1 = \iota^0 = (y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(4^{-n}, 1, m, N))$, as well as the sequence $\{\iota_n^2\}_{n=1}^\infty \subseteq \Omega_{m,N}^{\mathrm{LP,s}}$ by $\iota_n^2 = (y^{\mathrm{L}}(4^{-n}M_K), A^{\mathrm{LP,D}}(4^{-n}, 1, m, N))$. If we define $S^1 = \{1\}$ and $S^2 = \{0\}$, then $(4^{-n}M_K)/4^{-n} = 10^{-K}(\lfloor 10^K M \rfloor + 1)$, and so by Lemma 11.8 we have $\Xi_K(\iota_n^1) = \Xi_K(\iota^0) \in S^1$ and $\Xi_K(\iota_n^2) \in S^2$, for all $n \in \mathbb{N}$. The conditions (a) – (e) of Proposition 10.5 are readily seen to hold with for these sequences and the sets S^1 and S^2, allowing us to conclude that

there exists a $\hat{\Lambda}^{\mathrm{s}} \in \mathcal{L}^1(\Lambda_{m,N})$ such that $\{\Xi_K, \Omega_{m,N}^{\mathrm{LP,s}}, \{0,1\}, \hat{\Lambda}^{\mathrm{s}}\} \notin \Sigma_1^G$ as well as $\epsilon_{\mathbb{PB}}^{\mathrm{s}}(\mathrm{p}) \geq 1/2$, for $\mathrm{p} \in [0, 1/2)$.

Next, consider the sequences $\{\iota_n^1\}_{n=1}^\infty$ and $\{\iota_n^2\}_{n=1}^\infty$ given by $\iota_n^1 = (y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(-4^{-n}, 0, m, N))$ $\in \Omega_{m,N}^{\mathrm{LP,w}}$ and $\iota_n^2 = (y^{\mathrm{L}}(M_{K-1} \cdot 4^{-n}, m), A^{\mathrm{LP,D}}(4^{-n}, 0, m, N)) \in \Omega_{m,N}^{\mathrm{LP,w}}$. Defining $\iota^0 \notin \Omega$ according to $\iota^0 = (y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(0, 0, m, N))$, we have $|f(\iota_n^j) - f(\iota^0)| \leq 4^{-n}$, for $f \in \Lambda_{m,N}$, $j \in \{1, 2\}$, and $n \in \mathbb{N}$, whereas by Lemma 11.8 we obtain $\Xi(\iota_n^1) \in S^1 = \{1\}$ and $\Xi(\iota_n^2) \in S^2 = \{0\}$, and thus Proposition 10.5 implies the existence of a $\hat{\Lambda}^{\mathrm{w}} \in \mathcal{L}^1(\Lambda_{m,N})$ such that, for the computational problem $\{\Xi_{K-1}, \Omega_{m,N}^{\mathrm{LP,w}}, \{0,1\}, \hat{\Lambda}^{\mathrm{w}}\}$, we have $\epsilon_{\mathbb{PB}}^{\mathrm{w}}(\mathrm{p}) \geq 1/2$, for $\mathrm{p} \in [0, 1/2)$.

Now, defining $\hat{\Lambda}_{m,N} := \{f_{j,n} \mid j \leq n_{\mathrm{var}}, n \in \mathbb{N}\}$, where we let $f_{j,n}(\iota) = f_{j,n}^{\mathrm{s}}(\iota)$ if $\iota \in \Omega_{m,N}^{\mathrm{s}}$ and $f_{j,n}(\iota) = f_{j,n}^{\mathrm{w}}(\iota)$ if $\iota \in \Omega_{m,N}^{\mathrm{w}}$, for $j \leq n_{\mathrm{var}}$ and $n \in \mathbb{N}$, we have that $\hat{\Lambda}_{m,N}$ provides Δ_1-information for $\{\Xi_K, \Omega_{m,N}, \mathcal{M}_N, \Lambda_{m,N}\}$, and in view of Remark 10.4, we have that all the breakdown epsilon bounds mentioned above also hold for $\{\Xi_K, \Omega_{m,N}, \{0,1\}, \hat{\Lambda}_{m,N}\}$. This establishes parts (i) and (ii) of Proposition 9.43 as well as the weak breakdown epsilon bound in part (iii).

17.3. **Proof of Proposition 9.43, $M \notin \mathcal{S}_K^9 \implies$ (iii).** To complete the proof of part (iii) under the additional assumption $M \notin \mathcal{S}_K^9$, it remains to construct an algorithm that, for each m and N with $m < N$, outputs an approximate solution to $\{\Xi_{K-1}, \Omega_{m,N}, \{0,1\}, \Lambda_{m,N}\}^{\Delta_1} = \{\tilde{\Xi}_{K-1}, \tilde{\Omega}_{m,N}, \{0,1\}, \tilde{\Lambda}_{m,N}\}$ of accuracy 10^{-K-1}. Our construction is similar to the one used in §15. Concretely, the algorithm is constructed as follows. First note that $M_K < M_{K-1}$, since we are assuming $M \notin \mathcal{S}_K^9$. Now, writing $\tilde{\iota} = \left(\{y_j^{(n)}\}_{n=0}^\infty, \{A_{j,k}^{(n)}\}_{n=0}^\infty\right)_{j,k}$ for an input $\tilde{\iota} \in \tilde{\Omega}_{m,N}$ corresponding to an $\iota = (y, A) \in \Omega_{m,N}$, we define:

Algorithm $K - 1$ digit Smale's 9th problem

Inputs: Dimensions m, N.

Oracles: $\mathscr{O}_{\mathrm{mat}}$ providing access to the components $A_{1,k}^{(n)}$ of an input $\tilde{\iota} \in \tilde{\Omega}_{m,N}$.

Output: Either 1 or 0.

(1) We use $\mathscr{O}_{\mathrm{mat}}$ to read $A_{1,2}^{(2)}$. If $A_{1,2}^{(2)} \geq 3/4$ then we output 1 and terminate the algorithm, otherwise we proceed to the next step.

(2) We execute a loop that proceeds as follows – at each iteration, we increase n, starting with $n = 1$. Use oracle $\mathscr{O}_{\mathrm{mat}}$ to read $A_{1,1}^{(n)}$. If $A_{1,1}^{(n)} > 2^{-n}$ we output 0 and terminate the algorithm. If instead $A_{1,1}^{(n)} < -2^{-n}$ we output 1 and terminate the algorithm.

To show the correctness of this algorithm, we consider three cases depending on which $\iota \in \Omega_{m,N}$ the input $\tilde{\iota}$ corresponds to:

Case 1: The input $\iota = (y^{\mathrm{L}}(M_K\alpha, m), A^{\mathrm{LP,D}}(\alpha, 1, m, N)) \in \Omega_{m,N}^{\mathrm{LP,s}}$, for some $\alpha[0, 1]$. Observing that $(M_K\alpha)/\alpha = M_K < M_{K-1}$ whenever $\alpha \in (0, 1]$, Lemma 11.8 yields $\Xi_{K-1}(\iota) = 1$, for all $\alpha[0, 1]$. On the other hand, since $A^{\mathrm{LP,D}}(\alpha, 1, m, N)_{1,2} = 1$ by the definition of $A^{\mathrm{LP,D}}$, we obtain $A_{1,2}^{(2)} \geq 1 - 1/4 = 3/4$ and hence the algorithm also outputs 1.

Case 2: The input $\iota = (y^{\mathrm{L}}(M_{K-1}\alpha, m), A^{\mathrm{LP,D}}(\alpha, 0, m, N))$ for some $\alpha \in (0, 1]$. In this case we have $(M_{K-1}\alpha)/\alpha = M_{K-1}$, and thus by Lemma 11.8 we obtain that $\Xi_{K-1}(\iota) = 0$. To analyse the output of the algorithm, note first that since $A^{\mathrm{LP,D}}(\alpha, 1, m, N)_{1,2} = 0$ by the definition of $A^{\mathrm{LP,D}}$ we obtain $A_{1,2}^{(2)} \leq 0 + 1/4 < 3/4$ and hence the algorithm proceeds to step 2. We have $A_{1,1}^{(n)} \geq \alpha - 2^{-n} > -2^{-n}$ since α is positive. Hence the loop never terminates by outputting 1. However, again using that α is positive, for sufficiently large n we have $A_{1,1}^{(n)} \geq \alpha - 2^{-n} > 2^{-n}$ and hence the algorithm eventually outputs 0.

Case 3: The input $\iota = (y^{\mathrm{L}}(0, m), A^{\mathrm{LP,D}}(-\alpha, 0, m, N))$ for some $\alpha \in (0, 1]$. In this case, by Lemma 11.8 we obtain that $\Xi_{K-1}(\iota) = 1$. Our analysis of the output of the algorithm is similar to Case 2 and we once again proceed to step 2. This time the loop never terminates by outputting 0 since for all n

we have $A_{1,1}^{(n)} \leq -\alpha + 2^{-n} \leq 2^{-n}$ because α is positive. Instead for sufficiently large n we have $A_{1,1}^{(n)} \leq -\alpha - 2^{-n} < -2^{-n}$ and hence the algorithm eventually outputs 1.

These are the only possible cases for ι and so the algorithm always returns $\Xi_{K-1}(\iota)$, as desired.

17.4. Proof of Proposition 9.43, $M \notin \mathcal{S}_{K-1}^9 \implies$ (iv).

The assumption $M \notin \mathcal{S}_{K-1}^9$ implies that $M_{K-1} < M_{K-2}$ and therefore, as $(M_K\alpha)/\alpha = M_K \leq M_{K-1} < M_{K-2}$ and $(M_{K-1}\alpha)/\alpha = M_{K-1} < M_{K-2}$, for all $\alpha \in (0,1]$, Lemma 11.8 implies that $\Xi_{K-2}(\iota) = 1$ for all $\iota \in \Omega_{m,N}$. Our construction of an algorithm to compute Ξ_{K-2} is thus trivial – we simply create an algorithm that immediately outputs the value 1.

18. Geometry of solutions to problems (1.2) - (1.3) – Part II

18.1. A $(5s-1) \times 5s$ matrix with the RNP of order $s = 2^n$ for non-unique minimisers of ℓ^1 BP and ℓ^1 UL.

We construct a family of robust nullspace matrices which have a line segment of minimisers when performing basis pursuit denoising or unconstrained lasso with ℓ^1 regularisation.

Proposition 18.1. *Fix a natural number $s = 2^{n-1}$ and real numbers $\alpha, \gamma > 0$ with $\alpha > \gamma(\alpha^2+1)$. Then there is a matrix $A \in \mathbb{R}^{m \times N}$ with $m = 5s - 1$ and $N = 5s$ such that the following properties hold:*

(1) $\|AA^*\|_2 \leq \sqrt{\left(1 + \frac{\alpha^2\gamma^2}{4}\right)^2 + \frac{2\alpha^2\gamma^4}{16} + \frac{\gamma^4}{16}}$, $\quad \|(AA^*)^{-1}\|_2 \leq \sqrt{1 + 2\alpha^2 + \frac{1}{\gamma^4}\left(4 + \alpha^2\gamma^2\right)^2}$.

(2) A obeys the robust nullspace property of order s with parameters $\rho = 1/3$ and
$$\tau = \sqrt{\tfrac{4}{3}}\|A\|_2\|(AA^*)^{-1}\|_2.$$

(3) For any $\delta \geq 0$ there is a $y = Ax$ for some s-sparse x with $\|y\|_2 \leq \delta\sqrt{1+\alpha^2}$, such that

$$\Xi_{\mathrm{BPDN}}(y, A) = \{t\left(c_1\mathbf{1}_{2s} \oplus \mathbf{0}_{2s} \oplus c_2\mathbf{1}_s\right) + (1-t)\left(\mathbf{0}_{2s} \oplus -c_1\mathbf{1}_{2s} \oplus c_2\mathbf{1}_s\right) \mid t \in [0,1]\},$$

where

$$c_1 = \frac{\delta}{\sqrt{s}}\frac{\left(-\gamma - \gamma\alpha^2 + \alpha\right)}{C}, \quad c_2 = \frac{2\delta}{\gamma\sqrt{s}}\left(\frac{\alpha\gamma - 1}{C} + 1\right), \quad C = \sqrt{\gamma^2 + (1-\alpha\gamma)^2}. \qquad (18.1)$$

(4) For any $\lambda \geq 0$, there is a $y = Ax$ for some s-sparse x with $\|y\|_2 \leq \frac{\lambda C\sqrt{s}}{\gamma}\sqrt{1+\alpha^2}$, such that

$$\Xi_{\mathrm{UL}}(y, A) = \{t\left(\hat{c}_1\mathbf{1}_{2s} \oplus \mathbf{0}_{2s} \oplus \hat{c}_2\mathbf{1}_s\right) + (1-t)\left(\mathbf{0}_{2s} \oplus -\hat{c}_1\mathbf{1}_{2s} \oplus \hat{c}_2\mathbf{1}_s\right)\},$$

where now

$$\hat{c}_1 = \frac{\lambda}{\gamma}\left(-\gamma - \gamma\alpha^2 + \alpha\right), \quad \hat{c}_2 = \frac{2\lambda}{\gamma^2}\left(\alpha\gamma - 1 + C\right), \quad C = \sqrt{\gamma^2 + (1-\alpha\gamma)^2}. \qquad (18.2)$$

Proof. Let H_n be the $2^n \times 2^n$ dimensioned Hadamard matrix in 'natural ordering' (see B.2 for a definition). Set $m = 5s - 1$ and $N = 5s$, and let A be the $\mathbb{R}^{m \times N}$ matrix defined by

$$A = \frac{1}{2\sqrt{s}}P\begin{pmatrix} H_1 \otimes H_n & \eta \otimes H_{n-1} \\ \mathbf{0}_{s \times 4s} & \gamma H_{n-1} \end{pmatrix}, \quad \eta = (0, 0, \alpha\gamma, 0)^T$$

$$Pe_j = e_{j-1}, \; Pe_1 = 0,$$

where $\{e_j\}$ represents the canonical basis, so that P is the projection from $1, 2, \ldots, N$ to $2, 3, \ldots, N$. To show (1) we first compute

$$
\begin{pmatrix} H_1 \otimes H_n & \eta \otimes H_{n-1} \\ \mathbf{0}_{s \times 4s} & \gamma H_{n-1} \end{pmatrix} \begin{pmatrix} H_1 \otimes H_n & \eta \otimes H_{n-1} \\ \mathbf{0}_{s \times 4s} & \gamma H_{n-1} \end{pmatrix}^*
$$

$$
= \begin{pmatrix} 4s\, I_{4s} + \eta \otimes \eta^* \otimes sI_s & \gamma(\eta \otimes H_{n-1})H_{n-1}^* \\ \gamma((\eta \otimes H_{n-1})H_{n-1}^*)^* & \gamma^2 sI_s \end{pmatrix}
$$

$$
= \begin{pmatrix} 4s\, I_{4s} + \alpha^2\gamma^2 \xi \otimes \xi^* \otimes sI_s & \alpha\gamma^2 \xi \otimes sI_s \\ \alpha\gamma^2(\xi \otimes sI_s)^* & \gamma^2 sI_s \end{pmatrix} = 4s\, (I_{2s} \oplus (S \otimes I_s)),
$$

where $\xi = (0, 0, 1, 0)^T$ and

$$
S = \begin{pmatrix} (1 + \frac{\alpha^2\gamma^2}{4}) & 0 & \frac{\alpha\gamma^2}{4} \\ 0 & 1 & 0 \\ \frac{\alpha\gamma^2}{4} & 0 & \frac{\gamma^2}{4} \end{pmatrix}, \quad S^{-1} = \begin{pmatrix} 1 & 0 & -\alpha \\ 0 & 1 & 0 \\ -\alpha & 0 & \frac{1}{\gamma^2}(4 + \alpha^2\gamma^2) \end{pmatrix}. \tag{18.3}
$$

Thus, by the definition of P it follows that

$$
AA^* = I_{2s-1} \oplus (S \otimes I_s), \quad (AA^*)^{-1} = I_{2s-1} \oplus (S^{-1} \otimes I_s), \tag{18.4}
$$

and so to prove (1) we only need to estimate $\|S\|$ and $\|S^{-1}\|$ from (18.3). To do that, note that for a symmetric matrix $M \in \mathbb{R}^{2 \times 2}$ we have $\|M\|_2 \leq \|M\|_F = (M_{11}^2 + 2M_{12}^2 + M_{22}^2)^{\frac{1}{2}}$, and we therefore find $\|AA^*\|_2 \leq 1 \vee \sqrt{(1 + \alpha^2\gamma^2/4)^2 + \alpha^2\gamma^4/8 + \gamma^4/16}$, $\|(AA^*)^{-1}\|_2 \leq 1 \vee \sqrt{1 + 2\alpha^2 + \frac{1}{\gamma^4}(4 + \alpha^2\gamma^2)^2}$, establishing (1).

To prove (2), take an arbitrary vector let $v \in \mathbb{R}^N$ and write $v = \xi + A^*w$ where $\xi \in \ker(A)$. Note that $\ker(A)$ consists exactly of vectors of the form $\beta(\mathbf{1}_{4s} \oplus \mathbf{0}_s)$, for $\beta \in \mathbb{R}$. Now let $K = \{1, 2, \ldots, 4s\}$ and consider a set $S \subset \{1, \ldots, N\}$ of cardinality s. We then have $\|\xi_S\|_2 = \beta\sqrt{s}$ and $\|\xi_{S^c \cap K}\|_1 = \beta|S^c \cap K|$. Consequently

$$
\|\xi_S\|_2 = \frac{\sqrt{s}}{|S^c \cap K|}\|\xi_{S^c \cap K}\|_1 \leq \frac{\sqrt{s}}{|S^c \cap K|}(\|v_{S^c \cap K}\|_1 + \|(A^*w)_{S^c \cap K}\|_1)
$$

$$
\leq \frac{s\|v_{S^c}\|_1}{\sqrt{s}|S^c \cap K|} + \sqrt{\frac{s}{|S^c \cap K|}}\frac{\|(A^*w)_{S^c \cap K}\|_1}{\sqrt{|S^c \cap K|}}
$$

$$
\leq \frac{\rho\|v_{S^c}\|_1}{\sqrt{s}} + \sqrt{\frac{s}{|S^c \cap K|}}\|(A^*w)_{S^c \cap K}\|_2
$$

with $\rho = \frac{1}{3}$, where the last line follows because $|S^c \cap K| \geq 4s - s = 3s$. Hence,

$$
\|v_S\|_2 \leq \frac{\rho\|v_{S^c}\|_1}{\sqrt{s}} + \sqrt{\rho}\|(A^*w)_{S^c \cap K}\|_2 + \|(A^*w)_S\|_2 \leq \frac{\rho\|v_{S^c}\|_1}{\sqrt{s}} + \sqrt{1 + \rho}\|A^*w\|_2
$$

where the last inequality follows by applying the Cauchy-Schwartz inequality with the vectors $(\sqrt{\rho}, 1)$ and $(\|(A^*w)_{S^c \cap K}\|_2, \|(A^*w)_S\|_2)$. To bound $\|A^*w\|_2$, we see that

$$
\|A^*w\|_2 \leq \|A\|_2\|(AA^*)^{-1}(AA^*)w\|_2 \leq \|A\|_2\|(AA^*)^{-1}\|_2\|AA^*w\|_2 = \|A\|_2\|(AA^*)^{-1}\|_2\|Av\|_2
$$

where the last inequality is due to $Av = AA^*w + A\xi = AA^*w$ (recall that $\xi \in \ker(A)$). We conclude that

$$
\|v_S\|_2 \leq \frac{\rho\|v_{S^c}\|_1}{\sqrt{s}} + \sqrt{1 + \rho}\|A^*w\|_2 \leq \frac{\rho\|v_{S^c}\|_1}{\sqrt{s}} + \tau\|Av\|_2.
$$

where $\tau = \sqrt{\frac{4}{3}}\|A\|_2\|(AA^*)^{-1}\|_2$. This establishes (2).

We next prove (3) with the vector $y = \mathbf{0}_{2s-1} \oplus \alpha\delta \oplus \mathbf{0}_{2s-1} \oplus \delta \oplus \mathbf{0}_{s-1} = Ax$, where $x = \frac{2\delta}{\gamma\sqrt{s}}\left(\mathbf{0}_{4s} \oplus \mathbf{1}_s\right)$ is s-sparse. A simple calculation shows that $\|y\|_2 \leq \delta\sqrt{1+\alpha^2}$. With the aim of proving (3), define $x^{opt} = \frac{c_1}{2}\mathbf{1}_{2s} \oplus \left(-\frac{c_1}{2}\mathbf{1}_{2s}\right) \oplus c_2\mathbf{1}_s$ and note that then

$$Ax^{opt} - y = \frac{\delta}{C}\left(\mathbf{0}_{2s-1} \oplus (-\gamma) \oplus \mathbf{0}_{2s-1} \oplus (\alpha\gamma - 1) \oplus \mathbf{0}_{s-1}\right).$$

Thus $\|Ax^{opt} - y\|_2 = \delta$, and so x^{opt} is a feasible point for the BP denoising problem.

Next, we define a dual vector $p = \frac{2C\sqrt{s}}{\delta\gamma}(Ax^{opt} - y) = 2\sqrt{s}\left(\mathbf{0}_{2s-1} \oplus (-1) \oplus \mathbf{0}_{2s-1} \oplus (\alpha - \frac{1}{\gamma}) \oplus \mathbf{0}_{s-1}\right) \in \mathbb{R}^m$, and note that then by some simple calculations $\langle Ax^{opt} - y, p\rangle = \delta\|p\|_2$ and $-A^*p = \mathbf{1}_{2s} \oplus (-\mathbf{1}_{2s}) \oplus \mathbf{1}_s$. It is easily seen that $\alpha\gamma - 1 + C > 0$ and hence $c_2 > 0$. Moreover $\alpha > \gamma(\alpha^2 + 1)$ by an assumption in the statement of the proposition so that $c_1 > 0$. Hence $-A^*p \in \partial\|\cdot\|_1(x^{opt})$. Therefore, for arbitrary $z \in \mathbb{R}^N$ satisfying $\|Az - y\|_2 \leq \delta$, we have

$$\|z\|_1 \overset{C-S}{\geq} \|z\|_1 + \langle Az - y, p\rangle - \delta\|p\|_2 = \|z\|_1 + \langle Az - y, p\rangle - \langle Ax^{opt} - y, p\rangle$$
$$= \|z\|_1 - \langle z - x^{opt}, -A^*p\rangle \geq \|x^{opt}\|_1. \tag{18.5}$$

As x^{opt} is feasible for the BP denoising problem, we deduce that z is a minimiser if and only if the inequalities in (18.5) hold as equalities. This is the case if and only if $Az - y = \delta p/\|p\|_2 = Ax^{opt} - y$, $z_j \geq 0$ for $j \in \{1, \ldots, 2s\} \cup \{4s+1, \ldots, N\}$, as well as $z_j \leq 0$ for $j \in \{2s+1, \ldots, 4s\}$. In particular $A(x^{opt} - z) = 0$, and so, recalling that $\ker(A) = \{\beta(\mathbf{1}_{4s} \oplus \mathbf{0}_s)\,|\,\beta \in \mathbb{R}\}$, we must have $x^{opt} - z = \beta\left(\mathbf{1}_{4s} \oplus \mathbf{0}_s\right)$, for some $\beta \in \mathbb{R}$. It now follows that the minimisers are exactly as claimed in the statement of the proposition.

We finally establish (4) with $y = \mathbf{0}_{2s-1} \oplus \frac{\lambda C\alpha\sqrt{s}}{\gamma} \oplus \mathbf{0}_{2s-1} \oplus \frac{\lambda C\sqrt{s}}{\gamma} \oplus \mathbf{0}_{s-1} = Ax$, where $x = \frac{2\lambda C}{\gamma^2}\left(\mathbf{0}_{4s} \oplus \mathbf{1}_s\right)$ is s-sparse. Note that such a y satisfies $\|y\|_2 = \frac{\lambda C\sqrt{s}}{\gamma}\sqrt{1+\alpha^2}$. To this end, we define $x^{opt} = \frac{\hat{c}_1}{2}\mathbf{1}_{2s} \oplus \left(-\frac{\hat{c}_1}{2}\mathbf{1}_{2s}\right) \oplus \hat{c}_2\mathbf{1}_s$ and the dual vector

$$p = Ax^{opt} - y = \frac{\lambda\sqrt{s}}{\gamma}\left(\mathbf{0}_{2s-1} \oplus (-\gamma) \oplus \mathbf{0}_{2s-1} \oplus (\alpha\gamma - 1) \oplus \mathbf{0}_{s-1}\right).$$

Note that then by some simple calculations $-\frac{2}{\lambda}A^*p = \mathbf{1}_{2s} \oplus (-\mathbf{1}_{2s}) \oplus \mathbf{1}_s$. Since $\alpha > \gamma(\alpha^2 + 1)$ by the assumption in the theorem we must have $\hat{c}_1 > 0$ and furthermore $C > |1 - \alpha\gamma|$ so that $\hat{c}_2 \geq 0$. Hence $p \in \partial\|\cdot\|_1(x^{opt})$, and so for arbitrary $z \in \mathbb{R}^N$ we have

$$\frac{1}{2}\|Az - y\|_2^2 + \frac{\lambda}{2}\|z\|_1 \geq \langle Az - y, p\rangle - \frac{1}{2}\|p\|_2^2 + \frac{\lambda}{2}\|z\|_1$$
$$= \langle Az - Ax^{opt}, p\rangle + \langle Ax^{opt} - y, p\rangle - \frac{1}{2}\|p\|_2^2 + \frac{\lambda}{2}\|z\|_1$$
$$= \|p\|_2^2 - \frac{1}{2}\|p\|_2^2 + \frac{\lambda}{2}\left(\|z\|_1 - \langle z - x^{opt}, -\frac{2}{\lambda}A^*p\rangle\right) \tag{18.6}$$
$$\geq \frac{1}{2}\|Ax^{opt} - y\|_2^2 + \frac{\lambda}{2}\|x^{opt}\|_1.$$

where the first line follows by expanding $\|Az - y - p\|_2^2 \geq 0$.

We deduce that z is a minimiser if and only if the inequalities in (18.6) hold as equalities. This is the case if and only if $\|Ax - y - p\|_2^2 = 0$ (i.e. $Az - y = Ax^{opt} - y$), $z_j \geq 0$ for $j \in \{1, \ldots, 2s\} \cup \{4s+1, \ldots, N\}$, as well as $z_j \leq 0$ for $j \in \{2s+1, \ldots, 4s\}$. In particular $A(x^{opt} - z) = 0$, and so, recalling that $\ker(A) = \{\beta(\mathbf{1}_{4s} \oplus \mathbf{0}_s)\,|\,\beta \in \mathbb{R}\}$, we must have $x^{opt} - z = \beta\left(\mathbf{1}_{4s} \oplus \mathbf{0}_s\right)$, for some $\beta \in \mathbb{R}$. It now follows that the minimisers are exactly as claimed in the statement of the proposition. $\square$

18.2. **Perturbing ℓ^1 BP and ℓ^1 UL with several minimisers.** Before stating the main result of this section, we need to introduce the concept of the set of minimisers with minimal support.

Definition 18.2 (Minimisers with minimal support). Let $\{\Xi, \Omega\}$ denote the computational problem of either basis pursuit or unconstrained lasso with ℓ^1 regularisation. For feasible inputs $(y, U) \in \Omega$ we define the set of minimisers with minimal support by

$$\Xi^{\mathrm{ms}}(y, U) := \{x \in \Xi(y, U) \,|\, \forall x' \in \Xi(y, U), \, \mathrm{supp}(x') \subset \mathrm{supp}(x) \Rightarrow x = x'\}. \qquad (18.7)$$

Lemma 18.3. *Let $\{\Xi, \Omega\}$ denote the computational problem of either basis pursuit or unconstrained lasso with ℓ^1 regularisation with dimensions m and N. Suppose that $(y, U) \in \Omega$ is input data such that $\Xi^{ms}(y, U)$ contains two distinct points x^1 and x^2. Then, for every $\epsilon \in (0, 1)$, there exist positive semidefinite matrices $E^1 = E^1(\epsilon)$ and $E^2 = E^2(\epsilon)$ with $\|E^1\|_\infty \le \epsilon$ and $\|E^2\|_\infty \le \epsilon$ such that $\Xi(y, U(I - E^1)) = \{x^1\}$ and $\Xi(y, U(I - E^2)) = \{x^2\}$.*

Proof of Lemma 18.3. We define the $N \times N$ diagonal positive semidefinite matrices

$$E^j(\epsilon) := \mathrm{diag}(\epsilon \, \chi_{\{1 \notin \mathrm{supp}(x^j)\}}, \epsilon \, \chi_{\{2 \notin \mathrm{supp}(x^j)\}}, \ldots, \epsilon \, \chi_{\{N \notin \mathrm{supp}(x^j)\}}), \quad j = 1, 2,$$

$\chi_{\{i \notin \mathrm{supp}(x^j)\}}$ is 1 if $i \notin \mathrm{supp}(x^j)$ and 0 otherwise. We proceed to show that x^j is the unique vector in $\xi(y, U - UE^j)$, for $j = 1, 2$. It suffices to argue for $j = 1$, as the proof for $j = 2$ is analogous. Additionally, we will discuss basis pursuit and lasso separately.

Basis pursuit: Firstly, as $(U - UE^1)v = Uv$ for $v \in \mathbb{R}^N$ with $\mathrm{supp}(v) = \mathrm{supp}(x^1)$, we have $\|(U - UE^1)x^1 - y\|_2 \le \delta$. Let us suppose that $\tilde{x}^1 \in \mathbb{R}^N$ is a vector such that $\|(U - UE^1)\tilde{x}^1 - y\|_2 \le \delta$ and $\|\tilde{x}^1\|_1 \le \|x^1\|_1$. Set $\hat{x}^1 = \tilde{x}^1 - E^1\tilde{x}^1$. Then $\|U\hat{x}^1 - y\|_2 \le \delta$, and, for every $k \in \mathrm{supp}(\hat{x}^1)$ with $k \notin \mathrm{supp}(x^1)$ we have $|\hat{x}_k^1| = (1 - \epsilon)|\tilde{x}_k^1| < |\tilde{x}_k^1|$, whereas for $k \in \mathrm{supp}(x^1)$ we have $\hat{x}_k^1 = \tilde{x}_k^1$. Therefore, unless $\mathrm{supp}(\hat{x}^1) \subset \mathrm{supp}(x^1)$, we have $\|\hat{x}^1\|_1 < \|\tilde{x}^1\|_1 \le \|x^1\|_1$, contradicting the fact that $x^1 \in \Xi_{\mathrm{BPDN}}(y, U)$. Hence $\mathrm{supp}(\hat{x}^1) \subset \mathrm{supp}(x^1)$ and so $\hat{x}^1 = x^1$, by definition of $\Xi^{\mathrm{ms}}(y, U)$. We deduce that $\tilde{x}^1 = x^1$, and so x^1 is the unique vector in $\Xi_{\mathrm{BPDN}}(y, U - UE^1)$.

Lasso: Let us suppose that $\tilde{x}^1 \in \mathbb{R}^N$ satisfies $\|(U - UE^1)\tilde{x}^1 - y\|_2^2 + \lambda\|\tilde{x}^1\|_1 \le \|(U - UE^1)x^1 - y\|_2^2 + \lambda\|x^1\|_1$. As in the basis pursuit case, we set $\hat{x}^1 = \tilde{x}^1 - E^1\tilde{x}^1$. As before, unless $\mathrm{supp}(\hat{x}^1) \subset \mathrm{supp}(x^1)$ we have $\|\hat{x}^1\|_1 < \|\tilde{x}^1\|_1$ in which case

$$\begin{aligned}
\|U\hat{x}^1 - y\|_2^2 + \lambda\|\hat{x}^1\|_1 &< \|(U - UE^1)\tilde{x}^1 - y\|_2^2 + \lambda\|\tilde{x}^1\|_1 \\
&\le \|(U - UE^1)x^1 - y\|_2^2 + \lambda\|x^1\|_1 = \|Ux^1 - y\|_2^2 + \lambda\|x^1\|_1
\end{aligned}$$

contradicting the fact that $x^1 \in \Xi_{\mathrm{UL}}(y, U)$. We thus must have $\mathrm{supp}(\hat{x}^1) \subset \mathrm{supp}(x^1)$, and so $\hat{x}^1 = x^1$ by the definition of $\Xi^{\mathrm{ms}}(y, U)$. It follows that $\tilde{x}^1 = x^1$, and so x^1 is the unique vector in $\Xi_{\mathrm{UL}}(y, U - UE^1)$. $\qquad \square$

19. PROOF OF THEOREM 7.1: PARTS (I) AND (II)

Parts (i) and (ii) of Theorem 7.1 are formally stated in Proposition 9.36, which we now prove. We choose the constant C mentioned in Proposition 9.36 to be $C = (C_5 + 3) \vee 5$ where C_5 is the universal constant in Theorem B.6.

19.1. **Proof of part (i) of Proposition 9.36.**

Proof. We will establish this part using Proposition 10.5. Concretely, we will construct input sequences $\{\iota_n^1 = (y^0, A^{1,n})\}_{n=1}^\infty \subset \Omega_{s,m,N}^0$, $\{\iota^2 = (y^0, A^{2,n})\}_{n=1}^\infty \subset \Omega_{s,m,N}^0$ and an input $\iota^0 = (y^0, A^0) \in \Omega_{s,m,N}^0$ such that the following hold:

  (i) there exists $x^1, x^2 \in \mathbb{R}^N$ such that $\Xi(\iota_n^1) = \{x^1\}$ and $\Xi(\iota_n^2) = \{x^2\}$, for all $n \in \mathbb{N}$, and $\|x^1 - x^2\|_2 > \delta$.

(ii) the inputs satisfy

$$\|A^{j,n} - A^0\|_{\max} \leq 4^{-n}, \tag{19.1}$$

for all $n \in \mathbb{N}$ and $j \in \{1, 2\}$.

Once we have done this, the result will follow by applying item (iii) of Proposition 10.5 to obtain $\epsilon^{\mathrm{s}}_{\mathbb{P}hB}(\mathrm{p}) > \delta/2 \geq 10^{-K}$. Proposition 18.1 comes close to constructing a suitable A^0. However, the dimensions used in Proposition 18.1 are fixed as soon as s is chosen. To construct the desired matrix A^0 of dimensions m and N satisfying $N > m$ and $m \geq Cs \log^2(2s) \log(N)$, but otherwise arbitrary, we will concatenate a matrix provided by Proposition 18.1 and a matrix satisfying the nullspace property provided by Theorem B.6.

We begin by applying Proposition 18.1 with the constants $\alpha = 1.4$ and $\gamma = 0.37$ to obtain $A \in \mathbb{R}^{(5s-1)\times 5s}$, $y \in \mathbb{R}^{5s-1}$ and $x \in \mathbb{R}^{5s}$ such that the following hold.

a) By item (1) of Proposition 18.1 as well as the choices of α and γ, both $\|AA^*\|_2 < 31.3$ and $\|(AA^*)^{-1}\|_2 < 1.2$. In particular, $\|A\|_2 \leq 6$.

b) By item (2) of Proposition 18.1 A satisfies the ℓ^2-RNP of order s with parameters ρ_A and τ_A, where

$$\rho_A := 1/3 < \rho \quad \text{and} \quad \tau_A := \sqrt{4/3}\|A\|_2\|(AA^*)^{-1}\|_2 \leq 1.2 \cdot 6 \cdot 1.2 < 9.$$

c) By item (3) of Proposition 18.1, we have $y = Ax$ with x an s-sparse vector. Furthermore, $\|y\|_2 \leq \delta\sqrt{1+\alpha^2} \leq 2$ where we have used the choice of α and the bound $\delta \leq 1$ assumed in the statement of Theorem 7.1.

d) Again, using item (3) of Proposition 18.1, $\Xi^{\mathrm{ms}}(y, A) = \{\hat{x}^1, \hat{x}^2\}$ where

$$\hat{x}^1 = c_1 \mathbf{1}_{2s} \oplus \mathbf{0}_{2s} \oplus c_2 \mathbf{1}_s, \qquad \hat{x}^2 = \mathbf{0}_{2s} \oplus -c_1 \mathbf{1}_{2s} \oplus c_2 \mathbf{1}_s,$$

and the constants c_1 and c_2 are as given in (18.1).

Next, since $s \geq 2$ and $N > m \geq 5s \geq 10$ we must have

$$5s - 1 \leq 5s \leq \frac{5s \log(N) \log[s \log(N)] \log^2(s)}{\log(10) \log[2 \log(10)] \log^2(2)} \leq 3s \log(N) \log[s \log(N)] \log^2(s)$$

so that $m - (5s - 1) \geq (C - 3)s \log(N) \log[s \log(N)] \log^2(s)$. In particular, since $(C - 3) \geq C_5$ and $(C - 3) \geq 2$ we have

$$m - (5s - 1) \geq C_5 s \log(N) \log[s \log(N)] \log^2(s)$$

$$(N - 5s) \geq m - (5s - 1) \geq 4 \log(10) \log[2 \log(10)] \log^2(2) > 3.$$

Therefore we can apply Theorem B.6 to conclude there exists a matrix $F \in \mathbb{R}^{(m-(5s-1))\times(N-5s)}$ such that $\|F\|_2 \leq \sqrt{N/m}$ and F obeys the robust nullspace property with parameters (ρ_F, τ_F) satisfying $\rho_F < 1/3$ and $\tau_F < 2$.

At this stage we can finally define ι^0, ι_n^1 and ι_n^2: using Lemma 18.3 and d), we can find sequences of diagonal positive semidefinite matrices $\{E^{1,n}\}_{n=1}^{\infty}$ and $\{E^{2,n}\}_{n=1}^{\infty}$, with $\|E^{1,n}\|_{\max}, \|AE^{1,n}\|_2 \leq \gamma_n$ and $\|E^{2,n}\|_{\max}, \|AE^{2,n}\|_2 \leq \gamma_n$ such that $\Xi(y, A(I - E^{1,n})) = \{\tilde{x}^1\}$ and $\Xi(y, A(I - E^{2,n})) = \{\tilde{x}^2\}$ where the positive real numbers γ_n are chosen so that

$$0 < \gamma_n \leq \left(4^{-n} \wedge \frac{2}{3[10(\sqrt{s}+1)]} \wedge \frac{3\rho - 1}{30(\sqrt{s}+\rho)} \wedge \frac{\tau - 10}{10\tau}\right).$$

We define the matrices $A^0 = A \oplus F$, $A^{1,n} = A(I - E^{1,n}) \oplus F$, $A^{2,n} = A(I - E^{2,n}) \oplus F \in \mathbb{R}^{m\times N}$ as well as the vector $y^0 = y \oplus \mathbf{0}_{m-(5s-1)} \in \mathbb{R}^m$, and set $\iota^0 = (y^0, A^0)$, $\iota_n^j = (y^0, A^{j,n})$ for $j \in \{1, 2\}$, $n \in \mathbb{N}$.

Next, we establish that ι^0 is an element of $\Omega^0_{s,m,N}$. Firstly, note that $\|A^0\|_2 \leq \|A\| \vee \|F\|_2 \leq 6\sqrt{N/m} \leq b_2\sqrt{N/m}$ where we have used the bound $\|F\|_2 \leq \sqrt{N/m}$ from Theorem B.6 and the bound $\|A\|_2 \leq 6$ from a). Set $x^0 = x \oplus 0_{N-5s}$ where x is taken from c). Note that $A^0 x^0 = y \oplus 0_{m-(5s-1)} = y^0$ so that $y^0 = A^0 x^0$ with x^0 an s-sparse vector.

The last condition that we will check to ensure that $\iota^0 \in \Omega^0_{s,m,N}$ is the robust nullspace property. Let v be an arbitrary vector in $\mathbb{R}^N$ and write $v = v^A \oplus v^F \in \mathbb{R}^N$ with $v^A \in \mathbb{R}^{5s}$ and $v^F \in \mathbb{R}^{N-5s}$. Next, let S be a subset of $\{1, \ldots, N\}$ of cardinality s, and write $S_A = S \cap \{1, \ldots, 5s\}$, $S_F = S \cap \{5s+1, \ldots, N\}$, $S'_A = \{1, \ldots, 5s\} \setminus S_A$, and $S'_F = \{5s+1, \ldots, N\} \setminus S_F$. Then, as A (respectively F) satisfies the ℓ^2-RNP with constants ρ_A and τ_A (respectively ρ_F and τ_F), we have

$$\|v_S\|_2 \leq \|(v^A)_{S_A}\|_2 + \|(v^F)_{S_F}\|_2 \leq \frac{\rho_A}{\sqrt{s}}\|(v^A)_{S'_A}\|_1 + \tau_A\|Av\|_2 + \frac{\rho_F}{\sqrt{s}}\|(v^F)_{S'_F}\|_1 + \tau_F\|Fv\|_2$$

$$\leq \frac{\rho_A \vee \rho_F}{\sqrt{s}}\left(\|(v^A)_{S'_A}\|_1 + \|(v^F)_{S'_F}\|_1\right) + \sqrt{\tau_A^2 + \tau_F^2}\sqrt{\|Av^A\|_2^2 + \|Fv^F\|_2^2}$$

$$\leq \frac{\rho_{A^0}}{\sqrt{s}}\|v_{S^c}\|_1 + \tau_{A^0}\|A^0 v\|_2,$$

where $\rho_{A^0} := \rho_A \vee \rho_F < 1/3$ and $\tau_{A^0} := \sqrt{\tau_A^2 + \tau_F^2} < \sqrt{9^2 + 2^2} < 10$, and so A^0 satisfies the ℓ^2-RNP of order s with parameters $1/3$ and 10 (and in particular, A^0 also satisfies the ℓ^2-RNP with parameters ρ and τ).

Next, we argue that $\iota_n^j \in \Omega^0_{s,m,N}$, where $n \in \mathbb{N}$ and $j = 1$ or $j = 2$. We have $\|A^{j,n}\|_2 \leq \|A^0\|_2\|(I - E^{j,n})\|_2 < \|A^0\|_2 < b_2\sqrt{N/m}$ since $E^{j,n}$ is a positive semidefinite diagonal matrix with entries bounded above by γ_n and $\gamma_n < 1$. Setting $x^{j,n} = ((I - E^{j,n})^{-1}x) \oplus 0_{N-5s}$ where x is taken from c) gives $y^0 = A^{j,n}x^{j,n}$ for each n. Because $I - E^{j,n}$ is a diagonal matrix we conclude that $y^0 = A^{j,n}x^{j,n}$ and that $x^{j,n}$ is an s-sparse vector.

All that remains to prove that $\iota_n^j \in \Omega^0_{s,m,N}$ is to check that the robust nullspace property is satisfied with parameters ρ and τ. We have already shown that A^0 satisfies the RNP with parameters $(1/3, 10)$. We then use Lemma B.9 as well as the definition of γ_n to see that $A^{j,n}$ obeys the robust nullspace property with parameters (ρ_n^j, τ_n^j) where

$$\rho_n^j = \frac{\frac{1}{3} + 10\gamma_n\sqrt{s}}{1 - 10\gamma_n} \leq \frac{\frac{1}{3} + 10\sqrt{s}\left(\frac{3\rho-1}{30(\sqrt{s}+\rho)}\right)}{1 - 10\left(\frac{3\rho-1}{30(\sqrt{s}+\rho)}\right)} = \rho, \quad \tau_n^j = \frac{10}{1 - 10\gamma_n} \leq \frac{10}{1 - \frac{10(\tau-10)}{10\tau}} = \tau$$

We have therefore shown that $\iota^0 \in \Omega^0_{s,m,N}$ as well as $\iota_n^j \in \Omega^0_{s,m,N}$, for $j \in \{1, 2\}$ and $n \in \mathbb{N}$. Furthermore, by the definition of γ_n, $A^{j,n}$ and $E^{j,n}$ we have $\|A^{j,n} - A^0\|_{\max} \leq 4^{-n}$. By d), the block diagonality of $A^{j,n}$ and the fact that the last $m - (5s - 1)$ entries of y^0 are zero, it follows that the solution to the ℓ^1 basis pursuit problem with input $\iota_n^j \in \Omega^0_{s,m,N}$ is the unique point $x^j := \hat{x}^j \oplus \mathbf{0}_{N-5s}$, for $j \in \{1, 2\}$ and $n \in \mathbb{N}$. Now, recalling (18.1) and our choice $\alpha = 1.4$ and $\gamma = 0.37$, we find

$$\|x^1 - x^2\|_2 = \|\hat{x}^1 - \hat{x}^2\|_2 = \|c_1\mathbf{1}_{2s} \oplus c_1\mathbf{1}_{2s} \oplus \mathbf{0}_s\|_2 = c_1\sqrt{4s} = 2\delta \cdot \frac{-\gamma - \gamma\alpha^2 + \alpha}{\sqrt{\gamma^2 + (1-\alpha\gamma)^2}} > \frac{2\delta}{2} = \delta.$$

We have thus verified the conditions of Proposition 10.5 and so we deduce that $\epsilon_{\mathrm{B}}^{\mathrm{s}} \geq \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \geq \delta/2$ for $\mathrm{p} \in [0, 1/2)$, completing the proof of part (i) of Proposition 9.36.

$\square$

19.2. Proof of part (ii) of Proposition 9.36.

Proof. It suffices to show that $\epsilon_{\mathrm{B}}^{\mathrm{s}} \geq \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \geq \lambda\sqrt{s}/2$ for $\mathrm{p} \in [0, 1/2)$, for the ℓ^1 unconstrained lasso problem with input set $\Omega^0_{s,m,N}$. We proceed similarly to the proof of part (i), i.e., we will construct input sequences $\{\iota_n^1 = (y^0, A^{1,n})\}_{n=1}^{\infty} \subset \Omega^0_{s,m,N}$, $\{\iota_n^2 = (y^0, A^{2,n})\}_{n=1}^{\infty} \subset \Omega^0_{s,m,N}$ and an input $\iota^0 = (y^0, A^0) \in \Omega^0_{s,m,N}$ such that the following hold:

(i) there exists $x^1, x^2 \in \mathbb{R}^N$ such that $\Xi(\iota_n^1) = \{x^1\}$ and $\Xi(\iota_n^2) = \{x^2\}$, for all $n \in \mathbb{N}$, and $\|x^1 - x^2\|_2 > \lambda\sqrt{s}$.

(ii) the inputs satisfy

$$\|A^{j,n} - A^0\|_{\max} \le 4^{-n}, \tag{19.2}$$

for all $n \in \mathbb{N}$ and $j \in \{1, 2\}$.

Our choice of A^0 and $A^{j,n}$ for $j \in \{1, 2\}$ and $n \in \mathbb{N}$ is identical to part (i). This time however, we choose y using part (iv) of Proposition 18.1 with $\alpha = 1.4$ and $\gamma = 0.37$ and set $y^0 = y \oplus \mathbf{0}_{m-(5s-1)} \in \mathbb{R}^m$. In part (i) we showed that $\|A^0\|_2, \|A^{j,n}\|_2 \le b_2\sqrt{N/m}$. By the bound on y given in part (iv) of Proposition 18.1 as well as the choices of α and γ we have $\|y^0\|_2 = \|y\|_2 \le \lambda\sqrt{s}\gamma^{-1}\sqrt{\gamma^2 + (1-\alpha\gamma)^2}\sqrt{1+\alpha^2} < \lambda\sqrt{s} \cdot 3 < 2$.

We also showed in part (i) that the matrices A^0 and A_n^j satisfy the ℓ^2 RNP of order s with parameters (ρ, τ) and almost identical argument shows that $A^{j,n}x^{j,n} = y^0$ and $A^0x^0 = y^0$ for some s-sparse vectors $x^{j,n} \in \mathbb{R}^N$ and $x^0 \in \mathbb{R}^N$. Thus ι^0, ι_n^1, and ι_n^2 are each elements of $\Omega_{s,m,N}^0$. Moreover, we also showed in part (i) that $\|A^{j,n} - A^0\|_{\max} \le 4^{-n}$.

It again follows that the solution to the ℓ^1 unconstrained lasso problem with input $\iota_n^j \in \Omega_{s,m,N}^0$ is the unique point $x^j := \hat{x}^j \oplus \mathbf{0}_{N-5s}$, for $j \in \{1, 2\}$ and $n \in \mathbb{N}$. Recalling (18.2) and the choices $\alpha = 1.4$ and $\gamma = 0.37$, we find that

$$\|x^1 - x^2\|_2 = \|\tilde{x}^1 - \tilde{x}^2\|_2 = \|\hat{c}_1\mathbf{1}_{2s} \oplus \hat{c}_1\mathbf{1}_{2s} \oplus \mathbf{0}_s\|_2 = \hat{c}_1\sqrt{4s} = 2\lambda\sqrt{s} \cdot \frac{-\gamma - \gamma\alpha^2 + \alpha}{\gamma} > \frac{2\lambda\sqrt{s}}{2} = \lambda\sqrt{s}.$$

We have thus verified the conditions of Proposition 10.5, and so we deduce that $\epsilon_{\mathrm{B}}^{\mathrm{s}} \ge \epsilon_{\mathbb{P}h\mathrm{B}}^{\mathrm{s}}(\mathrm{p}) \ge \lambda\sqrt{s}/2$ for $\mathrm{p} \in [0, 1/2)$, for the ℓ^1 unconstrained lasso problem with input set $\Omega_{s,m,N}^0$, completing the proof of part (ii) of Theorem 9.36. $\qquad\square$

20. Proof of Theorem 7.1: part (iii)

20.1. **Preliminaries - results from the literature.** The ellipsoid algorithm will be the main driver in the proof of part (iii) of Theorem 7.1. Even though it is well known that this algorithm neither has the optimal theoretical complexity nor performs well in practice, it is a powerful and flexible tool to establish the "in P" statements rigorously.

We start by presenting the framework necessary to implement the ellipsoid algorithm in the context of ℓ^1 BP as necessitated in the proof of the theorem. We refer the reader to Appendix C for detailed definitions, and here we only list the notation assuming familiarity with the underlying concepts. For a compact convex set $\mathcal{K} \subset \mathbb{R}^n$ and real $\zeta > 0$, write $S(\mathcal{K}, -\zeta) = \{x \in \mathbb{R}^n \mid \overline{\mathcal{B}}_\zeta(x) \subset \mathcal{K}\}$ and $S(\mathcal{K}, \zeta) = \{x \in \mathbb{R}^n \mid \overline{\mathcal{B}}_\zeta(x) \cap \mathcal{K} \ne \varnothing\}$. For the same $\mathcal{K}$, we write $\mathrm{SEP}_\mathcal{K}$ for the weak separation oracle for $\mathcal{K}$.

The theorems we will cite can only be stated for compact convex sets that are contained within a ball of known radius R, so we introduce the following definition for conciseness.

Definition 20.1 (Circumscribed class). We call a set $\mathscr{K}$ a *circumscribed class* if it is a subset of

$$\{(\mathcal{K}, n, R) \mid n \in \mathbb{N}, R \in \mathbb{Q} \cap (0, \infty), \mathcal{K} \text{ compact and } \mathcal{K} \subset \overline{\mathcal{B}}_R(0) \subset \mathbb{R}^n\}.$$

We state the following result which combines [37, Cor. 4.2.7] on the existence of an 'oracle-polynomial' algorithm for the weak optimisation problem with the observation on pages 101 and 102 of [37] that if a separation oracle for a compact convex set $\mathcal{K}$ can be implemented on a Turing machine whose runtime is polynomial in $\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R)$, then 'oracle-polynomial' implies that the weak optimisation problem can be solved in time polynomial in $\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R)$ as well.

Theorem 20.2 ([37, Cor. 4.2.7] Ellipsoid algorithm , Turing case). *Suppose $\mathscr{K}$ is a circumscribed class equipped with a Turing encoding function $\mathrm{Data}^{\mathrm{Tur}} : \mathscr{K} \to \mathcal{A}^*$ (as in Definition C.4), and assume that*

$\mathscr{K}$ *is Turing-polynomially separable with respect to* $\mathrm{Data}^{\mathrm{Tur}}$ *(according to Definition C.5). Then there exists a Turing machine that takes in* $\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R) \in \mathcal{A}^*$, *for* $(\mathcal{K}, n, R) \in \mathscr{K}$, *a* $c \in \mathbb{Q}^n$, *and a rational* $\zeta > 0$ *as input and solves the weak optimisation problem* $(\mathcal{K}, R, c, \zeta)$ *(see Definition C.1 for the definition of a weak optimisation problem), such that the runtime of the Turing machine is bounded by a polynomial in*

$$\mathrm{Len}\big(\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R)\big), \ \mathrm{Len}(R), \ \mathrm{Len}(c), \ \mathrm{Len}(\zeta), \ and \ n.$$

The other result we will need is an analogue of Theorem 20.2 in the BSS model for convex bodies that are specified as the intersection of the level sets of several convex functions. This will be sufficient for our needs.

Theorem 20.3 ([57, Thm. 1.1] Ellipsoid algorithm, BSS case)**.** *Suppose* $\mathscr{K}$ *is a circumscribed class so that, for every* $(\mathcal{K}, n, R) \in \mathscr{K}$, *there exist convex functions* $f_1^{\mathcal{K}}, \ldots, f_{M(\mathcal{K})}^{\mathcal{K}} : \mathbb{R}^n \to \mathbb{R}$ *such that* $\mathcal{K} = \{z \in \mathbb{R}^n \mid f_j^{\mathcal{K}}(z) \leq 0, j = 1, \ldots, M(\mathcal{K})\}$. *Furthermore, assume that* $\mathscr{K}$ *is equipped with a BSS encoding function* $\mathrm{Data}^{\mathrm{BSS}} : \mathscr{K} \to \mathcal{V}$ *(as in Definition C.6) and assume that* $\mathscr{K}$ *is BSS-polynomially separable with respect to* $\mathrm{Data}^{\mathrm{BSS}}$ *(according to Definition C.7). Then there exists a BSS machine that takes in* $\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}, n, R) \in \mathcal{V}$, *for* $(\mathcal{K}, n, R) \in \mathscr{K}$, *a* $c \in \mathbb{R}^n$, *and a real* $\zeta > 0$ *as input and solves the weak optimisation problem* $(\mathcal{K}, R, c, \zeta)$, *such that the runtime of the BSS machine is bounded by a polynomial in*

$$\dim\big(\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}, n, R)\big) \quad and \quad \log\big(2 + V(\mathcal{K}, n, R)/\zeta\big),$$

where the scale factor $V(\mathcal{K}, n, R)$ *is defined according to*

$$V(\mathcal{K}, n, R) = \max_{1 \leq j \leq M(\mathcal{K})} \left[\max_{\|z\|_2 \leq R} f_j^{\mathcal{K}}(z) - \min_{\|z\|_2 \leq R} f_j^{\mathcal{K}}(z) \right] \vee R\|c\|_2.$$

20.2. **Proof of part (iii) of Proposition 9.36.** Part (iii) of Theorem 7.1 is formally stated as part (iii) of Proposition 9.36, which we now prove. To do so, we will make use of are Theorems 20.2 and 20.3 on the complexity of the ellipsoid algorithm. In addition, we will make use of the standard compressed sensing result Theorem B.7 and the easy Lemma B.8 stated in Appendix B.

As in §13, we fix the notation for an element $\tilde{\iota}$ of $\tilde{\Omega}$, writing $\tilde{\iota} = \big(\{y_j^{(n)}\}_{n=0}^{\infty}, \{A_{j,k}^{(n)}\}_{n=0}^{\infty}\big)_{j,k}$, corresponding to an $\iota = (y, A) \in \Omega_{s,m,N}^{\epsilon}$.

Proof of Proposition 9.36 part (iii). To start with, we mention the following quantities that the algorithm stores in a lookup table

$$L_1 = \left\lceil \log_2\left(\frac{16\tau}{1 - \rho}\right) \right\rceil, \quad L_2 = \lceil \log_2(b_2) \rceil, \quad L_3 = 4\tau b_1 \tag{20.1}$$

The algorithm computes the following quantities, which (for the convenience of the reader) are listed alongside a brief description of their purpose:

$n_1 = \left\lceil \dfrac{\mathrm{Len}(m)}{2} \right\rceil + 4K + 5 + L_1$ — The number of digits of precision for the input vector requested from the oracle

$n_2 = \left\lceil \dfrac{\mathrm{Len}(N) + \mathrm{Len}(m)}{2} \right\rceil + 1 + L_1$ — The number of digits of precision for the input matrix requested from the oracles.

$n_3 = \lceil \mathrm{Len}(N) \rceil + 4K + 6 + L_1 + L_2$ — The solution precision for the reformulated optimisation problem (20.3).

$n_4 = \left\lceil \dfrac{\mathrm{Len}(N)}{2} \right\rceil + K + 3$ — The precision used when converting algorithm output to a dyadic vector.

$$\delta' = \frac{10^{-K}(1-\rho)}{32\tau} + 2^{-4K-3-L_1} + 4b_2N\,2^{-n_3} \qquad \text{The denoising parameter used in (20.3)}$$

$$R = NL_3 \qquad\qquad \text{Used as a bound on solutions of the refor-} \atop \text{mulated optimisation problem.} \qquad (20.2)$$

Note that these quantities are functions of m, N, K, however we suppress making this dependence explicit to lighten the notation.

For $\iota = (y, A) \in \Omega^{\varepsilon}_{s,m,N}$, instead of directly solving the basis pursuit problem with ℓ^1 regularisation, our algorithm will be based on the solution of the following reformulated optimisation problem:

$$\min_{z \in \mathbb{R}^{2N}} \langle \mathbf{1}_{2N}, z \rangle \quad \text{s.t.} \quad z \geq \mathbf{0}_{2N}, \|\hat{A}z - y'\|_2 \leq \delta', \|z\|_2 \leq R, \qquad (20.3)$$

where $\hat{A} = (A' \ -A') \in \mathbb{R}^{m \times 2N}$, and y' and A' are approximations to y, A. More formally, for $\iota = (y, A) \in \Omega^{\varepsilon}_{m,N}$, we write

$$\mathcal{I}(\iota) := \left\{ (y', A') \in \mathbb{R}^m \times \mathbb{R}^{m \times N} \,\middle|\, \|y' - y\|_2 \leq 2^{-n_1}, \|A' - A\|_{\max} \leq 2^{-n_2} \right\}. \qquad (20.4)$$

Next, for $\iota \in \Omega^{\varepsilon}_{m,N}$ and $(y', A') \in \mathcal{I}(\iota)$, define the compact convex set $\mathcal{K}_{y',A'} = \{z \in \mathbb{R}^{2N} \,|\, z \geq \mathbf{0}_{2N}, \|\hat{A}z - y'\|_2 \leq \delta', \|z\|_2 \leq R\}$, where $\hat{A} = (A' \ -A')$, and set

$$\mathcal{K} = \{(\mathcal{K}_{y',A'}, 2N, R) \,|\, m, N \in \mathbb{N}, m \leq N, \iota \in \Omega^{\varepsilon}_{m,N}, (y', A') \in \mathcal{I}(\iota)\}.$$

As each of the $\mathcal{K}_{y',A'}$ are compact convex sets with $\mathcal{K}_{y',A'} \subset \overline{\mathcal{B}}_R(0) \subset \mathbb{R}^{2N}$, $\mathcal{K}$ is a circumscribed class.

An important observation that will be useful throughout is the following: the bound on K in (9.18) and the assumption that $16\tau\delta \leq (1 - \rho)$ yields

$$10^{-K} \geq \frac{16\tau(\delta + \epsilon)}{1 - \rho} \geq \frac{32\tau\epsilon}{1 - \rho}. \qquad (20.5)$$

We are now in a position to precisely define our algorithm, where the instructions whose labels end with a 'T' are only executed by the Turing model and those whose labels end with a 'B' are only executed by the BSS model:

Algorithm for basis pursuit with ℓ^1 regularisation:

Inputs: Dimensions m, N and an accuracy parameter K with $(m, N, K) \in \mathbb{N}^3$.

Oracles: $\mathcal{O}_{\mathrm{vec}}$ and $\mathcal{O}_{\mathrm{mat}}$ providing access to the components $y_j^{(n)}$ and $A_{j,k}^{(n)}$ of an input $\tilde{\iota} \in \tilde{\Omega}^{\epsilon}_{s,m,N}$.

Output: $x_{\mathrm{out}} \in \mathbb{D}^N$ (in the Turing case) or $x_{\mathrm{out}} \in \mathbb{R}^N$ (in the BSS case) with $\mathrm{dist}_{\mathcal{M}}(x_{\mathrm{out}}, \tilde{\Xi}_{\mathrm{BPDN}}(\tilde{\iota})) \leq 10^{-K}$.

1T. Using the inputs m, N and K as well as the stored values L_1, L_2 and L_3, compute the quantities $n_1, n_2, n_3, n_4, \delta'$ and R in (20.2). We encode the integers $n_1, \ldots n_4$ using their dyadic expansions. The rational numbers δ' and R are stored as pairs of integers each encoded using their dyadic expansions.

1B. Using the inputs m, N and K as well as the stored values L_1, L_2 and L_3, compute the quantities $n_1, n_2, n_3, n_4, \delta'$ and R where we encode $n_1, \ldots n_4$ as integers and δ', R as real numbers.

2. Call the oracles to obtain $(y', A') \in \mathcal{I}(\iota)$ with $y' = y^{(n_1)} \in \mathbb{R}^m$ and $A' = A^{(n_2)} \in \mathbb{R}^{m \times N}$.

3. Use the ellipsoid algorithm provided by Theorem 20.2 (in the Turing case) or Theorem 20.3 (in the BSS case) to solve the weak optimisation problem $(\mathcal{K}_{y',A'}, R, \mathbf{1}_{2N}, 2^{-n_3})$, yielding the point z^*.

4. Compute x^* according to $x_j^* = z_j^* - z_{j+N}^*$, for $j \in \{1, \ldots, N\}$.

5T. For the Turing machine, approximate x^* by a vector of dyadic rationals so that every component has precision n_4 by means of Newton-Raphson iteration and output the resulting vector as x_{out}.

5B. For the BSS machine, output $x_{\mathrm{out}} := x^*$.

To prove the correctness and polynomial bound on the complexity of the algorithm, we will do each of the following

I) We will show that the set $S(\mathcal{K}_{y',A'}, -2^{-n_3})$ is non-empty. We will do this by defining

$$\hat{x} := ((x)^+, (-x)^+) + 2^{-n_3}\mathbf{1}_{2N} \tag{20.6}$$

and showing that $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$, where $(x)^+ = \max\{0, x\}$ coordinate-wise.

II) We will to show that the conditions for Theorem 20.2 (in the Turing case) or Theorem 20.3 (in the BSS case) are met. With the result that $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$, this will imply that z^* is well defined

III) We need to show that the algorithm is correct i.e. that $\text{dist}_{\mathcal{M}}(x_{\text{out}}, \Xi(\iota)) \le 10^{-K}$.

IV) We need to show that the algorithm has bit complexity (in the Turing model) bounded above by a polynomial in N and K.

V) We need to show that the algorithm has arithmetic complexity (in the BSS model) bounded above by a polynomial in N and K.

For the purpose of simplifying the proof, we also introduce the following quantities

$$\varepsilon_1 = 2^{-4K-5-\lceil \log_2(\beta) \rceil}, \quad \varepsilon_2 = 2^{-1-\lceil \log_2(\beta) \rceil}.$$

Using these definitions, we note that the y' and A' defined in Instruction 2 of the algorithm will satisfy

$$\|y - y'\|_2 \le \sqrt{m} \cdot 2^{-n_1} \le 2^{\frac{\log_2(m)}{2}} \cdot 2^{-n_1} \le \varepsilon_1 \tag{20.7}$$

$$\|A - A'\|_2 \le \|A - A'\|_{\text{F}} \le \sqrt{mN} \cdot 2^{-n_2} \le 2^{\frac{\log_2(m)+\log_2(N)}{2}} \cdot 2^{-n_2} \le \varepsilon_2. \tag{20.8}$$

where we have used the fact that $\log_2(n) \le \text{Len}(n)$ for any integer n. We see from these definitions that δ' can be written as $\delta' = 10^{-K}(1-\rho)/(32\tau) + 2\epsilon_1 + 4b_2N2^{-n_3}$.

Step I - Showing that $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$

Using (20.7) and (20.8) we obtain

$$\|y'\|_2 \le \varepsilon_1 + b_1\sqrt{N/m} \le 2b_1\sqrt{N/m} \quad \text{and} \quad \|A'\|_2 \le \varepsilon_2 + b_2\sqrt{N/m} \le 2b_2\sqrt{N/m}. \tag{20.9}$$

Furthermore, by Lemma B.8, the definition of $\Omega^\varepsilon_{m,N}$ and (20.5) we get

$$\|x\|_2 \le \tau\left(\varepsilon + b_1\sqrt{N/m}\right) \le 2\tau b_1 N \quad \text{and} \quad \|A'x - y'\|_2 \le \varepsilon + 2\varepsilon_2 \le \frac{10^{-K}(1-\rho)}{32\tau} + 2\varepsilon_2 \tag{20.10}$$

Using this, we obtain the following for $u \in \overline{\mathcal{B}}_{2^{-n_3}}(0)$,

$$\left[\hat{x} + u\right]_j \ge 2^{-n_\varsigma} - u_j \ge 0, \quad \text{for all } j \in \{1, \dots, 2N\},$$

$$\|\hat{A}(\hat{x} + u) - y'\|_2 = \|A'x - y' + \hat{A}u\|_2$$

$$\le \|A'x - y'\|_2 + 2\|A'\|_2 \cdot \|u\|_2 \le \frac{10^{-K}(1-\rho)}{32\tau} + 2\varepsilon_1 + 2b_2\sqrt{N/m} \cdot 2^{-n_3} \le \delta'$$

$$\|\hat{x} + u\|_2 \le \|((x)^+, (-x)^+)\|_2 + 2^{-n_3}\|\mathbf{1}_{2N}\|_2 + \|u\|_2$$

$$\le 2\tau b_1 N + 2^{-n_3}\sqrt{2N} + 2^{-n_3} \le 2\tau b_1 N + \tau b_1\sqrt{2N}/2 + \tau b_1/2 \le R$$

where the penultimate inequality follows because $2^{-n_3} \le 1$ and $\tau, b_1 \ge 1$.

Therefore, we conclude that $\overline{\mathcal{B}}_{2^{-n_3}}(\hat{x}) \subset \mathcal{K}_{y',A'}$ and thus $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$.

Step II - Showing that the conditions for Theorem 20.2 (in the Turing case) or Theorem 20.3 (in the BSS case) are met

We will show that $\mathcal{K}$ is Turing-polynomially separable and BSS-polynomially separable (as specified in Definitions C.5 and C.7) with appropriate encoding functions $\text{Data}^{\text{Tur}} : \mathcal{K} \to \mathcal{A}^*$ and $\text{Data}^{\text{BSS}} :$

$\mathcal{K} \to \mathcal{V}$. Concretely, in the Turing case we define

$$\text{Data}^{\text{Tur}}\big((\mathcal{K}_{y',A'}, 2N, R)\big) := m \ ; \ 2N \ ; \ \delta' \ ; \ R \ ; \ y_1' \ ; \ y_2' \ \cdots \ ; \ y_m' \ ; \ A_{1,1}' \ ; \ A_{1,2}' \ \cdots \ ; \ A_{m,N}' \in \mathcal{A}^*$$

(20.11)

where $\mathcal{A} = \{0, \ 1, -, \ ., \ ;\}$ and all the dyadic rationals are written out in their binary representation, whereas in the BSS case we define

$$\text{Data}^{\text{BSS}}\big((\mathcal{K}_{y',A'}, 2N, R)\big) = \big(m, 2N, \delta', R, y_1', \cdots, y_m', A_{1,1}', A_{1,2}', \cdots, A_{m,N}'\big) \in \mathcal{V}.$$

where $\mathcal{V}$ is defined in Definition C.6.

We now present a polynomial-runtime subroutine that acts as a weak separation oracle for $\mathcal{K}_{y',A'}$ in both the Turing and BSS cases. We term this subroutine 'WSS'.

Subroutine WSS

Input: $\text{Data}^{\text{Tur}}\big((\mathcal{K}_{y',A'}, 2N, R)\big)$ (in the Turing case) or $\text{Data}^{\text{BSS}}\big((\mathcal{K}_{y',A'}, 2N, R)\big)$ (in the BSS case) specifying the set $\mathcal{K}_{y',A'}$, a vector w with $2N$ entries, and $\xi > 0$ (rational in the Turing case and real in the BSS case).

Output: either a $d \in \mathbb{R}^{2N}$ (rational in the Turing case) such that $\|d\|_\infty = 1$ and $\langle d, z - w \rangle < \xi$, for all $z \in S(\mathcal{K}_{y',A'}, -\xi)$, or the assertion that $w \in S(\mathcal{K}_{y',A'}, \xi)$.

1. Compute $\|\hat{A}w - y'\|_2^2$ and $\|w\|_2^2$ and verify whether or not $w \in \mathcal{K}_{y',A'}$ by testing the inequalities $w \geq \mathbf{0}_{2N}$ coordinate-wise, the inequality $\|\hat{A}w - y'\|_2^2 \leq \delta'^2$, and the inequality $\|w\|_2^2 \leq R^2$ in the definition of $\mathcal{K}_{y',A'}$. If the inequalities all hold then assert $w \in S(\mathcal{K}_{y',A'}, \xi)$ and exit this subroutine.

2. If the subroutine has not terminated, one of the following must be true.
   a. $w_j < 0$, for some $j \in \{1, \ldots, 2N\}$. In this case, we set $d = -e_j$.
   b. $\|\hat{A}w - y'\|_2^2 > \delta'^2$. In this case and assuming that a. does not hold, we set $g = \nabla\big(\|\hat{A} \cdot -y'\|_2^2\big)|_w = 2\hat{A}^T(\hat{A}w - y')$ and $d = g/\|g\|_\infty$.
   c. $\|w\|_2^2 > R^2$. In this case and assuming that neither a. nor b. hold, we set $d = w/\|w\|_\infty$.

Remark 20.4. Strictly speaking, 'WSS' is in fact a subroutine that solves the strong separation problem for $\mathcal{K}_{y',A'}$ (see [37, Def 2.1.4]). However, for the purpose of applying either Theorem 20.2 or Theorem 20.3 it is sufficient to show that 'WSS' is a weak separation subroutine.

This subroutine is well defined except for one potential issue with the execution of the division $g/\|g\|_\infty$ in instruction 2b if $g = 0$. We will first argue that this cannot be the case by showing that if w is such that $\|\hat{A}w - y'\|_2^2 > \delta'^2$ then $g \neq 0$.

Assume otherwise (i.e. that $g = 0$) for the sake of a contradiction. Then (decomposing y as $y' = y^0 + \hat{A}v$ where $y^0 \in \ker(\hat{A}^*)$) we obtain from $g = 0$ that $\hat{A}^*\hat{A}(w - v) = 0$. Hence $\|\hat{A}(w - v)\|^2 = \langle \hat{A}^*\hat{A}(w - v), w - v \rangle = 0$ so that $\hat{A}(w - v) = 0$. Therefore $\delta' > \|\hat{A}w - y'\|_2 = \|y^0\|_2$. But this contradicts the already established fact that $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$ - in particular, $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$ implies that $\|\hat{A}\hat{x} - y'\|_2 \leq \delta'$ and we derive a contradiction from the following argument:

$$\|y^0\|_2^2 \leq \|\hat{A}(\hat{x} - v)\|_2^2 + \|y_0\|_2^2 = \|\hat{A}(\hat{x} - v)\|_2^2 - 2\langle \hat{A}(\hat{x} - v), y_0 \rangle + \|y_0\|_2^2 = \|\hat{A}\hat{x} - y'\|_2^2 \leq \delta'^2$$

where the equality follows from the fact that $y^0 \in \ker(\hat{A}^*)$.

We have therefore shown that 'WSS' is well defined. To show that this is indeed a weak separation subroutine for $\mathcal{K}_{y',A'}$, we first show that 'WSS' exits either with the correct assertion that $w \in S(\mathcal{K}_{y',A'}, \xi)$ or by constructing d with $\langle d, z - w \rangle < 0 < \xi$, for all $z \in \mathcal{K}_{y',A'}$. To see this, note first that if 'WSS' exits in line 1 then $w \in \mathcal{K}_{y',A'}$ and that $\mathcal{K}_{y',A'} \subseteq S(\mathcal{K}_{y',A'}, \xi)$. Hence if 'WSS' exits in line 1 then it correctly asserts that $w \in S(\mathcal{K}_{y',A'}, \xi)$.

If 'WSS' does not exit in line 1, we examine the three possible cases for line 2, assuming that z is a vector in $S(\mathcal{K}_{y',A'}, -\xi) \subset \mathcal{K}_{y',A'}$.

    a. If $w_j < 0$ for some $j \in \{1, \ldots, 2N\}$ then the output d of 'WSS' satisfies $\langle d, z - w \rangle = -z_j + w_j < 0$ where we have used that $z \in \mathcal{K}_{y',A'}$ implies that $z_j \geq 0$.

    b. If $\|\hat{A}w - y'\|_2^2 > \delta'^2$ then

$$\langle \hat{A}^T(\hat{A}w - y'), z - w \rangle = \langle \hat{A}w - y', \hat{A}z - y \rangle + \langle \hat{A}w - y', y - \hat{A}w \rangle$$

$$\leq (\|\hat{A}w - y'\|_2 - \|\hat{A}z - y\|_2)\|\hat{A}z - y\|_2 \leq 0$$

    where we have used that $z \in \mathcal{K}_{y',A'}$ implies that $\|\hat{A}z - y\|_2 \leq \delta$. Thus $\langle d, z - w \rangle = \|g\|_\infty^{-1}\langle g, z - w \rangle = 2\|g\|_\infty^{-1}\langle \hat{A}^T(\hat{A}w - y'), z - w \rangle \leq 0$.

    c. If $\|w\|_2^2 > R^2$ then $\langle w, z - w \rangle \leq \|w\|_2(\|z\|_2 - \|w\|_2) < \|w\|_2(R - R)$ where we have used that $z \in \mathcal{K}_{y',A'}$ implies that $\|z\|_2 \leq R$. Hence $\langle d, z - w \rangle < 0$.

In all three cases we have constructed a $d \in \mathbb{R}^{2N}$ (rational in the Turing case) such that $\|d\|_\infty = 1$ and $\langle d, z - w \rangle < 0 < \xi$, for all $z \in S(\mathcal{K}_{y',A'}, -\xi)$.

It is easy to see that the runtime of 'WSS' is bounded by a polynomial in $\mathrm{Len}\big(\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}_{y',A'}, 2N, R)\big)$, $\mathrm{Len}(R)$, $\mathrm{Len}(w)$, $\mathrm{Len}(\xi)$, and N in the Turing case, and, in the BSS case, by a polynomial of $\dim\big((\mathrm{Data}^{\mathrm{BSS}}((\mathcal{K}_{y',A'}, 2N, R))\big)$. We hence deduce that $\mathscr{K}$ is both Turing-polynomially separable and BSS-polynomially separable with respect to the encoding functions $\mathrm{Data}^{\mathrm{Tur}}$ and $\mathrm{Data}^{\mathrm{BSS}}$.

We are now in a position to apply Theorem 20.2 to the weak optimisation problem $(\mathcal{K}_{y',A'}, R, \mathbf{1}_{2N}, 2^{-n_\varsigma})$, for all $(\mathcal{K}_{y',A'}, 2N, R) \in \mathscr{K}$ to complete the argument for this step in the Turing model. For the BSS case, in order to apply Theorem 20.3 to the same weak optimisation problem, however, we need to establish the existence of convex functions $f_1^{y',A'}, \ldots, f_{M(y',A')}^{y',A'} : \mathbb{R}^{2N} \to \mathbb{R}$ such that $\mathcal{K}_{y',A'} = \{z \in \mathbb{R}^{2N} \mid f_j^{\mathcal{K}_{y',A'}}(z) \leq 0, j = 1, \ldots, M(y', A')\}$, for all $(\mathcal{K}_{y',A'}, 2N, R) \in \mathscr{K}$. Recalling the definition of $\mathcal{K}_{y',A'}$, this is achieved easily by setting

$$f_j^{\mathcal{K}_{y',A'}}(z) = -z_j \quad j = 1, \ldots, 2N, \quad f_{2N+1}^{\mathcal{K}_{y',A'}}(z) = \|\hat{A}z - y'\|_2^2 - \delta'^2, \quad f_{2N+2}^{\mathcal{K}_{y',A'}}(z) = \|z\|_2^2 - R^2.$$
$$(20.12)$$

Step III - Proof of correctness

We will show that $\mathrm{dist}_\mathcal{M}(x_{\mathrm{out}}, \Xi_{\mathrm{BPDN}}(\iota)) \leq 10^{-K}$. Since z^* is a solution to the weak optimisation problem $(\mathcal{K}_{y',A'}, R', \mathbf{1}_{2N}, 2^{-n_\varsigma})$

    a) There exists a $\hat{z} \in \mathcal{K}_{y',A'}$ such that $\|z^* - \hat{z}\|_2 \leq 2^{-n_3}$. In particular, such a $\hat{z}$ must have non-negative entries and must satisfy $\|\hat{A}\hat{z} - y'\|_2 \leq \delta'$.

    b) For all $z \in S(\mathcal{K}, -2^{-n_3})$ we have $\langle \mathbf{1}_{2N}, z^* \rangle \leq \langle \mathbf{1}_{2N}, z \rangle + 2^{-n_3}$.

By a) we obtain

$$|\langle \mathbf{1}_{2N}, z^* \rangle - \|z^*\|_1| \leq |\langle \mathbf{1}_{2N}, z^* \rangle - \langle \mathbf{1}_{2N}, \hat{z} \rangle| + |\langle \mathbf{1}_{2N}, \hat{z} \rangle - \|\hat{z}\|_1| + |\|\hat{z}\|_1 - \|z^*\|_1|$$

$$\leq |\langle \mathbf{1}_{2N}, z^* - \hat{z} \rangle| + |\|\hat{z} - z^*\|_1| \leq 2\sqrt{2N}\|z^* - \hat{z}\|_2 \leq 2\sqrt{2N}2^{-n_3} \quad (20.13)$$

Combining (20.13) with b) as well as the previous result that $\hat{x}$ defined in (20.6) satisfies $\hat{x} \in S(\mathcal{K}_{y',A'}, -2^{-n_3})$ gives the following

$$\|x^*\|_1 \leq \|\{z_j^*\}_{j=1}^N\|_1 + \|\{z_j^*\}_{j=N+1}^{2N}\|_1 \underset{(20.13)}{\leq} \sqrt{2N}2^{1-n_3} + \langle \mathbf{1}_{2N}, z^* \rangle \leq \langle \mathbf{1}_{2N}, \hat{x} \rangle + (2\sqrt{2N} + 1)2^{-n_3}$$

$$= \|\hat{x}\|_1 + (2\sqrt{2N} + 1)2^{-n_3} \leq \|x\|_1 + (2N + 2\sqrt{2N} + 1)2^{-n_3} \leq \|x\|_1 + 6Nb_2 2^{-n_3}$$
$$(20.14)$$

where we have used the definition of $\hat{x}$ in the penultimate inequality and the final inequality uses the fact that $b_2 \geq 1$.

Next, returning to a) and using the definition of δ' and (20.9) we can estimate

$$\|A'x^* - y'\|_2 = \|\hat{A}z^* - y'\|_2 = \|\hat{A}\hat{z} - y' + \hat{A}(z^* - \hat{z})\|_2 \leq \|\hat{A}\hat{z} - y'\|_2 + 2\|A'\|_2\|z^* - \hat{z}\|_2$$

$$\leq \delta' + 4b_2\sqrt{N/m} \cdot 2^{-n_3} \leq \frac{10^{-K}(1-\rho)}{32\tau} + 2\varepsilon_1 + 8b_2 N 2^{-n_3}. \tag{20.15}$$

Therefore

$$\|Ax - Ax^*\|_2 \leq \|A - A'\|_2\|x - x^*\|_2 + \|A'x - y'\|_2 + \|A'x^* - y'\|_2$$

$$\leq \varepsilon_2\|x - x^*\|_2 + \frac{10^{-K}(1-\rho)}{16\tau} + 4\varepsilon_1 + 8b_A N 2^{-n_3} \tag{20.16}$$

where the bound from $\|A - A'\|_2$ comes from the definition of A' in the algorithm $\Gamma^{\mathscr{O}_{\mathrm{vec}}, \mathscr{O}_{\mathrm{mat}}}$, the bound for $\|A'x - y'\|_2$ comes from (20.10) and the bound for $\|A'x^* - y'\|_2$ from (20.15).

We now use Theorem B.7 applied to x and x^* as well as the fact that x is s-sparse to obtain

$$\|x - x^*\|_2 \leq \frac{(1+\rho)^2}{1-\rho}\frac{1}{\sqrt{s}}\left(\|x^*\|_1 - \|x\|_1\right) + \frac{(3+\rho)\tau}{1-\rho}\|Ax - Ax^*\|_2$$

$$\leq \frac{4\tau}{1-\rho}\left(\|x^*\|_1 - \|x\|_1 + \|Ax - Ax^*\|_2\right)$$

since $\tau, s \geq 1$ and $\rho \in [0, 1)$. Therefore employing equations (20.14) and (20.16) gives

$$\|x - x^*\|_2 \leq \frac{4\tau}{1-\rho}\left(14Nb_2 2^{-n_3} + \varepsilon_2\|x - x^*\|_2 + \frac{10^{-K}(1-\rho)}{16\tau} + 4\varepsilon_1\right). \tag{20.17}$$

From the definitions of $\varepsilon_1, \varepsilon_2$ and n_3, we also obtain

$$2^{-n_3} \leq 2^{-4K - \log_2(N) - \log_2(64) - \log_2(8\tau/(1-\rho)) - \log_2 b_2} \leq \frac{10^{-K}(1-\rho)}{512Nb_2\tau}$$

and

$$\frac{16\tau\varepsilon_1}{1-\rho} = \frac{16\tau}{1-\rho} \cdot 2^{-4K - 5 - \lceil\log_2(8\tau/(1-\rho))\rceil} \leq \frac{10^{-K}}{16}, \quad \frac{4\tau\varepsilon_2}{1-\rho} = \frac{4\tau}{1-\rho} \cdot 2^{-1 - \lceil\log_2(8\tau/(1-\rho))\rceil} \leq \frac{1}{4}.$$

Therefore (20.17) can be written as

$$\|x - x^*\|_2 \leq \frac{10^{-K}}{8} + \frac{\|x - x^*\|_2}{4} + \frac{10^{-K}}{4} + \frac{10^{-K}}{16} = \frac{\|x - x^*\|_2}{4} + \frac{7 \cdot 10^{-K}}{16} \tag{20.18}$$

Since $\epsilon \leq \delta$, the set $\Xi_{\mathrm{BPDN}}(\iota)$ is non-empty. Fix a solution $\xi^* \in \Xi_{\mathrm{BPDN}}(\iota)$. Then, as $\|Ax - y\|_2 \leq \varepsilon \leq \delta$, we have that x is feasible for the basis pursuit problem with ℓ^1 regularisation with parameter δ and input (y, A). In particular, since ξ^* is optimal for this problem, we have $\|\xi^*\|_1 \leq \|x\|_1$, and thus Theorem B.7 applied to ξ^* and x yields

$$\|\xi^* - x\|_2 \leq \frac{(3+\rho)\tau}{1-\rho}\left(\|A\xi^* - y\|_2 + \|Ax - y\|_2\right) \leq \frac{4\tau(\delta + \varepsilon)}{1-\rho} \leq \frac{10^{-K}}{4} \tag{20.19}$$

where we have once again made use of (20.5).

Next, we note that $x_{\mathrm{out}} = x^*$ in the BSS case, whereas

$$\|x_{\mathrm{out}} - x^*\|_2 \leq 2^{-n_4}\sqrt{N} = 2^{-\lceil\frac{\mathrm{Len}(N)}{2}\rceil - K - 3}\sqrt{N} \leq \frac{10^{-K}}{8}$$

in the Turing case, where we have used the fact that $\mathrm{Len}(n) \geq \log_2(n)$ for a natural number n. Combining this with (20.18) and (20.19) finally yields

$$\|x_{\mathrm{out}} - \xi^*\|_2 \leq \|x_{\mathrm{out}} - x^*\|_2 + \|x - x^*\|_2 + \|\xi^* - x\|_2 \leq \frac{10^{-K}}{8} + \frac{4 \cdot 7 \cdot 10^{-K}}{3 \cdot 16} + \frac{10^{-K}}{4} \leq 10^{-K}$$

Hence $\mathrm{dist}_{\mathcal{M}}\left(x_{\mathrm{out}}, \Xi_{\mathrm{BPDN}}(\iota)\right) \leq \|x_{\mathrm{out}} - \xi^*\|_2 \leq 10^{-K}$.

Step IV - Establishing a polynomial upper bound on the computational complexity of the algorithm in the Turing case

The initial computation of the integers n_1, n_2, n_3 and n_4 can be analysed as follows: firstly, computation of both $\mathrm{Len}(N)/2$ and $\mathrm{Len}(m)/2$ can be established in $\mathcal{O}(\mathrm{Len}(N) \vee \mathrm{Len}(m))$ bit operations. After that, we note that we require only finitely many multiplications and additions involving the quantities $\mathrm{Len}(N), \mathrm{Len}(m), K$ and the fixed quantities L_1 and L_2. Thus the complexity of computing n_1, n_2, n_3 and n_4 can be bound by a polynomial in $\mathrm{Len}(\mathrm{Len}(N)), \mathrm{Len}(\mathrm{Len}(m))$ and $\mathrm{Len}(K)$. This immediately implies that n_1, n_2, n_3 and n_4 each have lengths polynomial in $\mathrm{Len}(\mathrm{Len}(N)), \mathrm{Len}(\mathrm{Len}(m))$ and $\mathrm{Len}(K)$.

δ' is the sum of three fractions: the first, $10^{-K}(1-\rho)/(32\tau)$, can be computed as a pair of integers in bit complexity polynomial in K (as $(1-\rho)/(32\tau)$ is fixed. The second, $2^{-4K-3-L_1}$ can also be computed in bit complexity polynomial in K as L_1 is fixed. Finally, $4b_A N 2^{-n_3}$ can be computed in bit complexity polynomial in $N, \mathrm{Len}(m)$ and K as b_A is fixed and we have already established that the length of n_3 is polynomial in $\mathrm{Len}(\mathrm{Len}(N)), \mathrm{Len}(\mathrm{Len}(m))$ and $\mathrm{Len}(K)$. Thus the overall cost of computing the rational number δ' expressed as a pair of two integers is polynomial in K, N and $\mathrm{Len}(m)$.

Finally, R involves a multiplication between N and the fixed value L_3 and can thus be computed in time polynomial in N. Therefore the overall complexity of the first instruction is polynomial in K, N and $\mathrm{Len}(m)$.

Since n_1 and n_2 are both bounded from above by a polynomial in $\mathrm{Len}(m), \mathrm{Len}(N)$ and K, the cost of accessing the oracle to produce each y'_j and $A'_{j,k}$ is bounded above by a polynomial in $j, k, \mathrm{Len}(m)$, $\mathrm{Len}(N)$, and K (in both the arithmetic and standard Turing complexity models). As this process is repeated for $j \in \{1, 2, \ldots, m\}$ and $k \in \{1, 2, \ldots, N\}$ to obtain y' and A', the overall complexity of the second instruction for the algorithm is bounded above by a polynomial in m, N and K.

Moreover, we have already shown that n_1 and n_2 are such that both $\mathrm{Len}(n_1)$ and $\mathrm{Len}(n_2)$ are bounded from above by a polynomial in $\mathrm{Len}(\mathrm{Len}(N)), \mathrm{Len}(\mathrm{Len}(m))$ and $\mathrm{Len}(K)$. Furthermore, (20.9) tells us that y' and A' are bound above by polynomials in $\sqrt{N/m}$. Thus the resulting lengths of y'_k and $A'_{j,k}$ is also polynomial in $\mathrm{Len}(N), \mathrm{Len}(m)$ and $\mathrm{Len}(K)$.

By Theorem 20.2, the bit-complexity of the third instruction is polynomial in

$$\mathrm{Len}\big(\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}_{y',A'}, R, \mathbf{1}_{2N}, 2^{-n_3})\big), \ \mathrm{Len}(R), \ \mathrm{Len}(1_{2n}), \ \mathrm{Len}(2^{-n_3}) \text{ and } 2N.$$

Because of the way $\mathrm{Data}^{\mathrm{Tur}}$ has been defined in (20.11) we have

$$\mathrm{Len}\big(\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}_{y',A'}, R, \mathbf{1}_{2N})\big) = \mathrm{Len}(m) + \mathrm{Len}(N) + \mathrm{Len}(\delta')$$
$$+ \mathrm{Len}(R) + \sum_{j=1}^{m}\left(\mathrm{Len}(y'_j) + \sum_{k=1}^{N}\mathrm{Len}(A'_{j,k})\right)$$

and we have already established that each term on the right hand side is polynomial in m, N and $\mathrm{Len}(K)$. Furthermore, $\mathrm{Len}(R)$ is polynomial in $\mathrm{Len}(N)$, $\mathrm{Len}(1_{2N})$ is linear in N and $\mathrm{Len}(2^{-n_3}) = -n_3$ which is polynomial in $\mathrm{Len}(N)$ and K. Thus the bit-complexity of the third instruction is polynomial in m, N and K. The length of each component of the resulting vector z^* must therefore also have $\mathrm{Len}(z^*)$ polynomial in m, N and K.

Since the fourth instruction involves a subtraction between two portions of z^*, the bit-complexity of executing this instruction is polynomial in m, N and K. Finally, the division to convert the rational vector $x^* \in \mathbb{Q}^N$ into the output x_{out} expressed as a vector of dyadic rationals can be done in complexity polynomial in $\mathrm{Len}(x^*), \mathrm{Len}(n_4)$ and N using Newton-Raphson iterations (see [58, pp. 92-93]). As each of these quantities are bound above by a polynomial in m, N and K, we see that the bit-complexity of executing this final step is polynomial in m, N and K.

We have therefore shown that each instruction can be executed in bit complexity bound above by a polynomial in m, N and K and thus as there are only five instructions executed in the algorithm the

overall complexity can be bound above by a polynomial in m, N and K, completing the proof in the Turing case.

Step V - Establishing a polynomial upper bound on the computational complexity of the algorithm in the BSS case

For a natural number n, it is possible to compute $\mathrm{Len}(n)$ on a BSS machine by incrementing a value k (starting at $k = 1$) and stopping when $2^k > n$. This can be done in $\mathrm{Len}(n)$ arithmetic operations and thus the BSS complexity of computing n_1 through n_4 is bounded by a polynomial in $\mathrm{Len}(N)$ and $\mathrm{Len}(m)$. The computation of δ' can be done with BSS complexity bounded above by a polynomial in K and n_3. Since n_3 is itself linear in K and $\mathrm{Len}(N)$, the BSS complexity of computing δ' is bounded above by a polynomial in K and $\mathrm{Len}(N)$. Finally, computing R requires a single arithmetic operation. Thus the computational cost of executing the first instruction on a BSS machine is polynomial in $\mathrm{Len}(N), \mathrm{Len}(m)$ and K.

In the same way as in the arithmetic Turing case, the cost of calling the oracles in the second instruction of the algorithm is polynomial in m, N and K. In the context of our application of Theorem 20.3 in the third instruction of the algorithm, the parameter V is equal to

$$V = R\|\mathbf{1}_{2N}\|_2 \vee \max_{1 \leq j \leq 2N+2} g_j, \quad \text{where} \quad g_j := \max_{\|z\|_2 \leq R} f_j^{\mathcal{K}_{y',A'}}(z) - \min_{\|z\|_2 \leq R} f_j^{\mathcal{K}_{y',A'}}(z)$$

and $f^{\mathcal{K}_{y',A'}}$ are defined as in (20.12). From these definitions, it follows that $g_j = \max_{\|z\|_2 \leq R} z_j - \min_{\|z\|_2 \leq R} z_j = 2R$ for $j = 1, 2, \ldots, 2N$, that

$$g_{2N+1} = \max_{\|z\|_2 \leq R} \|Az - y'\|_2^2 - \min_{\|z\|_2 \leq R} \|Az - y'\|_2^2$$

$$\leq (\|A'\|_2 \cdot 2\|z\|_2 + \|y'\|_2)^2 \leq \left(4b_2\sqrt{\frac{N}{m}}R + 2b_1\sqrt{\frac{N}{m}}\right)^2 \leq 16NR^2(b_1 + b_2)^2$$

and that $g_{2N+2} = \max_{\|z\|_2 \leq R} \|z\|_2^2 - \min_{\|z\|_2 \leq R} \|z\|_2^2 = R^2$. Therefore $V \leq R\sqrt{2N} \vee 2R \vee 16NR^2(b_1 + b_2)^2 \vee R^2 = 16NR^2(b_1 + b_2)^2$. Thus $V \leq C_{\mathrm{BSS}}N^3$ where the real number C_{BSS} exceeding 16 depends only on b_1, b_2, τ (crucially, C_{BSS} is independent of m, N and K).

Thus $\log(2 + V/2^{-n_3}) \leq \log(2C_{\mathrm{BSS}}N^3 2^{-n_3}) = n_3 \log(2) + 3\log(N) + \log(2C_{\mathrm{BSS}})$. In particular, using the definition of n_3, we note that $\log(2 + V/2^{-n_3})$ can be written as a polynomial in $\mathrm{Len}(N)$ and K. The dimension of $\mathrm{Data}^{\mathrm{BSS}}((\mathcal{K}_{y',A'}, 2N, R))$ is $4 + m + mN$ and thus we conclude via Theorem 20.3 that the BSS complexity of the third instruction of the algorithm is polynomial in m, N and K.

The fourth instruction takes N arithmetic operations and the final instruction is constant time on a BSS machine. Thus the overall complexity of the algorithm is polynomial in m, N and K, as claimed. $\qquad \square$

21. Proof of Theorem 7.1: part (iv)

21.1. Preliminaries - distance to several minimisers for the ℓ^1 BP problem.

We begin by proving the following lemma that allows us to create examples of problems with infinite condition number for the basis pursuit problem with $\delta = 0$.

Lemma 21.1. *Let $\{\Xi, \Omega\}$ be the computational problem of basis pursuit with $\delta = 0$. Suppose that $A \in \mathbb{R}^{m \times N}$ is a matrix such that for every $y' \in \mathbb{R}^m$, $(y', A) \in \Omega^{act}$. Furthermore, suppose that there is a non-empty set $S \subseteq \{1, 2, \ldots, N\}$ and a vector v in the row span of A such that the following conditions hold:*

(1) there is a non-zero $\hat{\xi} \in \mathbb{R}^N$ such that $\mathrm{supp}(\hat{\xi}) \subseteq S$ and $A\hat{\xi} = 0$, and
(2) $\|v_{S^c}\|_\infty \leq 1$, and $|v_i| = 1$ for all $i \in S$.

If $T \subseteq S$ is a set such that

    (3) if $A\rho = 0$ and $supp(\rho) \cap T \neq \varnothing$ for some vector $\rho \in \mathbb{R}^N$, then there is a non-zero ρ' with
        $supp(\rho) \cap T \subseteq supp(\rho') \subseteq S$ and $A\rho' = 0$,

then there exists an $x \in \mathbb{R}^N$ with $supp(x) = T$ so that $\mathrm{dist}_2[(Ax, A), \Sigma^{\mathrm{RCC}} \cup \Omega^{act}] = 0$ and that $\|Ax\|_2 \leq 1$. Note that condition (3) is trivially satisfied if $S = \{1, 2, \ldots, N\}$ (with $\rho' = \rho$) or $T = \varnothing$ (where there is nothing to show).

Proof. Let $\xi \in \mathbb{R}^N$ be a vector such that $A\xi = 0$, $supp(\xi) \subseteq S$, and ξ has maximal support in S, in the sense that

    If $\xi' \in \mathbb{R}^N$ is such that, if $A\xi' = 0$ and $supp(\xi) \subseteq supp(\xi') \subseteq S$, then $supp(\xi') = supp(\xi)$.   (21.1)

The existence of ξ is guaranteed by condition (1) in the statement of the lemma: indeed, the set of supports of all vectors in the nullspace of A is a finite, non-empty (here we use 1), partially ordered set under the set inclusion ordering and thus such a set of supports contains at least one maximal element (one of which we choose arbitrarily and call R). Thus there must exist a vector ξ in the nullspace of A with support R satisfying (21.1).

    Next, set $c = (\sqrt{N}\|A\|_2)^{-1} \wedge (\sqrt{N}\|\xi\|_\infty \|A\|_2)^{-1}$ (note that c is well defined as both ξ and A are non-zero) and partition $R = supp(\xi)$ into sets J_+ and J_- according to

$$J_+ = \{i \in supp(\xi) \,|\, v_i\xi_i > 0\}, \qquad J_- = \{i \in supp(\xi) \,|\, v_i\xi_i < 0\},$$

and let

$$x_i = \begin{cases} \mathrm{sgn}(v_i\xi_i)\, c\xi_i & \text{if } i \in T \cap supp(\xi) \\ c & \text{if } i \in T \setminus supp(\xi) \\ 0 & \text{otherwise} \end{cases}.$$

Then clearly $supp(x) = T$ and $\|Ax\|_2 \leq \|A\|_2\sqrt{N}\|x\|_\infty \leq 1$. We now proceed to establish the following for every sufficiently small $\epsilon > 0$:

    (I) $A(x + \epsilon\xi_{J_+}) = A(x - \epsilon\xi_{J_-})$,
    (II) $\|x + \epsilon\xi_{J_+}\|_1 = \|x - \epsilon\xi_{J_-}\|_1$, and
    (III) if $\hat{\xi} \in \mathbb{R}^N$ is such that $A(\hat{\xi} + x + \epsilon\xi_{J_+}) = A(x + \epsilon\xi_{J_+})$, then $\|x + \epsilon\xi_{J_+} + \hat{\xi}\|_1 \geq \|x + \epsilon\xi_{J_+}\|_1$.

This will imply that $\{x + \epsilon\xi_{J_+}, x + \epsilon\xi_{J_-}\} \subset \Xi\big(A(x + \epsilon\xi_{J_+}), A\big)$ and thus that $\big(A(x + \epsilon\xi_{J_+}), A\big) \in \Sigma^{\mathrm{RCC}}$. Indeed, from (III), $x + \epsilon\xi_{J_+}$ is a minimiser, whereas (I) implies that $x - \epsilon\xi_{J_-}$ is feasible and (II) implies that $x - \epsilon\xi_{J_-}$ is also a minimiser. Consequently, using the assumption that $\big(A(x + \epsilon\xi_{J_+}), A\big) \in \Omega^{\mathrm{act}}$, we obtain

$$\mathrm{dist}_2[(Ax, A), \Sigma^{\mathrm{RCC}} \cap \Omega^{\mathrm{act}}] \leq \|A(x + \epsilon\xi_{J_+}) - Ax\|_2 \leq \epsilon\|\xi\|_2\|U\|_2$$

and, as ϵ was arbitrary and ξ and A are independent of ϵ, we must have $C_{\mathrm{RCC}}(Ax, A) = \infty$.

    It hence suffices to establish (I), (II), and (II) in order to complete the proof. To this end, note that $0 = A\xi = A(\xi_{J_+} + \xi_{J_-}) = A\xi_{J_+} + A\xi_{J_-}$, and so (I) follows immediately. Next, since ξ is in the nullspace of A and v is in the row span of A, we have $\langle v, \xi \rangle = 0$ and, as $v_i = \mathrm{sgn}(v_i)$ for $i \in supp(\xi) \subset S$, we obtain

$$0 = \sum_{i \in supp(\xi)} v_i\xi_i = \sum_{i \in supp(\xi)} \mathrm{sgn}(v_i)\mathrm{sgn}(\xi_i)|\xi_i| = \sum_{i \in supp(\xi)} \mathrm{sgn}(v_i\xi_i)|\xi_i| = \|\xi_{J_+}\|_1 - \|\xi_{J_-}\|_1$$

i.e. that $\|\xi_{J_+}\|_1 = \|\xi_{J_-}\|_1$. Next, we have

$$
\left(x + \epsilon \xi_{J_+}\right)_i = \begin{cases} (c+\epsilon)\xi_i & \text{if } i \in T \cap J_+ \\ -c\xi_i & \text{if } i \in T \cap J_- \\ c & \text{if } i \in T \setminus \mathrm{supp}(\xi) \\ \epsilon\xi_i & \text{if } i \in J_+ \setminus T \\ 0 & \text{otherwise} \end{cases}, \quad \left(x_i - \epsilon \xi_{J_-}\right)_i = \begin{cases} c\xi_i & \text{if } i \in T \cap J_+ \\ -(c+\epsilon)\xi_i & \text{if } i \in T \cap J_- \\ c & \text{if } i \in T \setminus \mathrm{supp}(\xi) \\ -\epsilon\xi_i & \text{if } i \in J_- \setminus T \\ 0 & \text{otherwise} \end{cases}
$$
(21.2)

and therefore $\|x + \epsilon\xi_{J_+}\|_1 = \|x\|_1 + \epsilon\|\xi_{J_+}\|_1 = \|x\|_1 + \epsilon\|\xi_{J_-}\|_1 = \|x - \epsilon\xi_{J_-}\|_1$, establishing (II).

Now consider an arbitrary $\hat\xi$ such that $A\hat\xi = 0$. We claim that $\mathrm{supp}(\hat\xi) \subset (T \setminus \mathrm{supp}(\xi))^c$. To prove this, suppose by way of contradiction that there exists a $t \in \mathrm{supp}(\hat\xi) \cap (T \setminus \mathrm{supp}(\xi))$. Then property (3) implies that we can find a $\rho' \neq 0$ such that $A\rho' = 0$ and $\mathrm{supp}(\hat\xi) \cap T \subset \mathrm{supp}(\rho') \subset S$. But then $A(\xi + \kappa\rho') = 0$ and $\mathrm{supp}(\xi + \kappa\rho') \supseteq \mathrm{supp}(\xi) \cup \{t\} \supsetneq \mathrm{supp}(\xi)$, for sufficiently small $\kappa > 0$, contradicting (21.1). Therefore $\mathrm{supp}(\hat\xi) \subset (T \setminus \mathrm{supp}(\xi))^c$.

Next, by inspecting the explicit expression for $x + \epsilon\xi_{J_+}$ in (21.2) and recalling that $\|v\|_\infty \leq 1$, we see that $(x + \epsilon\xi_{J_+} + \hat\xi)_i = (x + \epsilon\xi_{J_+})_i = c$, for $i \in T \setminus \mathrm{supp}(\xi)$, and $\mathrm{sgn}\big((x + \epsilon\xi_{J_+})_i\big) \supset \{v_i\}$, for all $i \in (T \setminus \mathrm{supp}(\xi))^c$, and therefore

$$
\|x + \epsilon\xi_{J_+} + \hat\xi\|_1 - \|x + \epsilon\xi_{J_+}\|_1 = \sum_{i \in (T \setminus \mathrm{supp}(\xi))^c} |(x + \epsilon\xi_{J_+} + \hat\xi)_i| - |(x + \epsilon\xi_{J_+})_i|
$$
$$
\geq \sum_{i \in (T \setminus \mathrm{supp}(\xi))^c} \hat\xi_i v_i = \langle \hat\xi, v \rangle = 0,
$$

where the last equality is due to the fact that v is in the row span of A and $A\hat\xi = 0$. This establishes (III) and concludes the proof. $\qquad\square$

For the purpose of proving part (iv) of Theorem 7.1, we will apply Lemma 21.1 to both the case where A is a subsampled Hadamard matrix and the case where A is a subsampled Hadamard to Haar matrix, as in §B.2. We do this in the following result:

Lemma 21.2. *Let $\{\Xi, \Omega\}$ be the computational problem of basis pursuit with $\delta = 0$, and let $A \in \mathbb{R}^{m \times N}$ with $1 \leq m < N$ be either*

  (i) *a row-subsampled Hadamard matrix, or*

  (ii) *a row-subsampled Hadamard-to-Haar matrix as given in (B.10), with $S_i \neq \varnothing$, for $i = 1, 2 \ldots, n-1$.*

Additionally, assume that $(y', A) \in \Omega^{act}$ for all $y' \in \mathbb{R}^m$. Then, for every set $T \subseteq \{1, 2, \ldots, N\}$, there is a vector $x \in \mathbb{R}^N$ with support T so that $\mathrm{dist}_2[(Ax, A), \Sigma^{\mathrm{RCC}} \cap \Omega^{act}] = 0$ and $\|Ax\|_2 \leq 1$.

Proof of Lemma 21.2. Our aim will be to verify that A satisfies the desired conditions of Lemma 21.1.

It suffices to verify the conditions (1) and to construct a suitable v satisfying (2) with $S = \{1, \ldots, N\}$, noting the observation that (3) holds trivially for such an S and any T as defined in this lemma. To this end, we first observe that (1) holds simply as $m < N$.

Next, in order to establish (2), we construct a vector $v \in \mathbb{R}^N$ in the row span of A such that $|v_i| = 1$, for all $i = 1, \ldots, N$. If A is a row-subsampled Hadamard matrix, we can simply take v to be the first row of A. Suppose now that A is a row-subsampled Hadamard-to-Haar matrix as given in (B.10), with $S_i \neq \varnothing$, for $i = 1, 2 \ldots, n-1$. In this case we select an arbitrary $j_i \in S_i$, for every $i = 1, 2 \ldots, n-1$, and let $v = (1) \oplus \bigoplus_{i=0}^{n-1} \sqrt{2^i}(X^i)^*_{j_i}$, where $(X^i)^*_{j_i}$ denotes the j_i-th row of X^i. Since $|(X^i)_{j,k}| = 1/\sqrt{2^i}$, for $i = 0, 1, 2, \ldots, n-1$ we have $|v_i| = 1$ for $i = 1, 2, \ldots, N$.

We can now apply Lemma 21.1 to complete the proof, where we use the assumption that $(y', A) \in \Omega^{act}$ for all $y' \in \mathbb{R}^m$. $\qquad\square$

We are now ready to prove part (iv) of Theorem 7.1. The argument proceeds as follows - first, we use existing results from the literature (Theorem B.5 and Theorem B.3) to argue that if N is sufficiently large and provided m is sufficiently large relative to N, then, with strictly positive probability, a randomly subsampled Hadamard or Hadamard-to-Haar matrix will satisfy the RIP and hence the robust nullspace property. This will, in particular, imply the existence of at least one such matrix A, to which Lemma 21.2 will be applied to conclude the proof.

21.2. **Proof of part (iv) of Proposition 9.36.** In the language of the SCI hierarchy, Theorem 7.1 part (iv) is written as Proposition 9.36 part (iv). We will prove this proposition.

Proof of Proposition 9.36 part (iv). Fix an s as in the theorem statement and set $K = 1$. Let C_4 be the constant in Theorem B.5 corresponding to K. We proceed assuming that $N = 2^n$ for some natural number $n \geq 2$ satisfying $N - 2 \geq 2C_4 s \log(N) \log(s \log(N)) \log(s)^2$ and we set $m = N - 1$. Clearly, there are infinitely many such pairs (m, N).

Next we let $U \in \mathbb{R}^{N \times N} = \mathbb{R}^{2^n \times 2^n}$ be either a Hadamard matrix or a Hadamard to Haar matrix. We will produce a subsampled matrix A, a vector $x \in \mathbb{R}^N$ and a vector $y \in \mathbb{R}^m$ with the following properties:

(1) $y = Ax$ with $|\mathrm{supp}(x)| = s$
(2) A satisfies the ℓ^2-RNP of order s with parameters (ρ, τ).
(3) $\|y\|_2 \leq 1$ and $\|A\|_2 \leq \sqrt{\frac{N}{m}}$
(4) $\mathrm{dist}_2[(y, A), \Sigma^{\mathrm{RCC}} \cap (\Omega^0_{s,m,N})^{\mathrm{act}}] = 0$.

This will complete the proof. Indeed, if we set $\iota = (y, A)$ then (1)-(3) proves that $\iota \in \Omega^0_{s,m,N}$ and (4) proves that $C_{\mathrm{RCC}}(\iota) = \infty$.

We divide into two cases depending on whether U is a Hadamard matrix or a Hadamard to Haar matrix.

- If U is a Hadamard matrix, we start by noting that

$$m = N - 1 \geq N - 2 \geq C_4 s \log(N) \log(s \log(N)) \log(s)^2$$

  and thus we can employ Theorem B.5 to conclude that there exists a set S with $|S| = m$ such that $A := P_S \sqrt{\frac{N}{m}} U$ satisfies the RIP of order $2s$ with constant $\delta_{2s} \leq 1/5$. Moreover, note that since U is unitary, $\|A\|_2 = \sqrt{N/m} \leq b_2 \sqrt{N/m}$.
- If U is a Hadamard-to-Haar matrix then U can be decomposed as $U = (1) \oplus \bigoplus_{i=0}^{n-1} X^i$ by the argument in (B.10). By observing that

$$2^{n-1} - 1 = N/2 - 1 \geq C_4 s \log(N) \log(s \log(N)) \log(s)^2$$

  we can employ Theorem B.5 to see that there exists a set S_{n-1} of size $2^{n-1} - 1$ such that $c_{n-1} P_{S_{n-1}} X^{n-1}$ with $c_{n-1} = \sqrt{\frac{2^{n-1}}{2^{n-1}-1}}$ satisfies the RIP of order $2s$ with constant $\delta_{2s} \leq 1/5$. We choose

$$A = (1) \oplus \bigoplus_{i=0}^{n-2} X^i \oplus c_{n-1} P_{S_{n-1}} X^{n-1}.$$

  Since X^i are unitary for $i = 0, 1, \ldots, n-2$ and $c_{n-1} P_{S_{n-1}} X^{n-1}$ obeys the RIP of order $2s$ with constant $\delta_{2s} \leq 1/5$ the matrix A also obeys the RIP of order $2s$ with constant $\delta_{2s} \leq 1/5$. Furthermore,

$$\|A\|_2 = c_{n-1} = \sqrt{\frac{2^{n-1}}{2^{n-1}-1}} \leq \sqrt{2}\sqrt{\frac{2^n}{2^n-1}} \leq b_2 \sqrt{\frac{N}{m}}$$

In either case, we deduce that there is a matrix $A \in \mathbb{R}^{m \times N}$ subsampled from U such that A obeys the RIP of order $2s$ with constant $\delta_{2s} \leq 1/5$ and $\|A\|_2 \leq b_2 \sqrt{N/m}$. By Theorem B.3, A satisfies the ℓ^2-RNP of order s with parameters

$$\frac{1/4}{\sqrt{1 - (1/4)^2} - (1/4)/4} < \frac{1}{3} < \rho \quad \text{and} \quad \frac{\sqrt{1 + 1/4}}{\sqrt{1 - (1/4)^2} - (1/4)/4} < 10 < \tau. \tag{21.3}$$

Furthermore, the construction of A is such that A contains a non-zero entry in both the Hadamard case or the Hadamard to Haar case (here we have used the fact that S_{n-1} is non-empty and that every element of X^i is non-zero). For a given $j \in \{1, 2, \ldots, m\}$, let $k \in \{1, 2, \ldots, N\}$ be such that $A_{j,k} \neq 0$. Using (21.3) and the bound $\|A\|_2 \leq b_2 \sqrt{N/m}$, we note that for sufficiently small $\epsilon > 0$ both $(\epsilon A e_k, A), (-\epsilon A e_k, A) \in \Omega^0_{s,m,N}$.

Since the jth entry of the vectors $A(\epsilon e_k)$ and $-A(\epsilon e_k)$ are not equal, we have shown that the jth coordinate of the input vector is an active coordinate of $\Omega^0_{s,m,N}$, i.e. the jth coordinate of the input vector is in $\mathcal{A}(\Omega^0_{s,m,N})$ (see (9.1)). We have thus shown that for every $y' \in \mathbb{R}^m$, we have $(y', A) \in (\Omega^0_{s,m,N})^{\text{act}}$.

Thus the conditions of Lemma 21.2 apply and we obtain x such that $\|Ax\|_2 \leq 1$, $\text{dist}_2[(Ax, A), \Sigma^{\text{RCC}} \cap (\Omega^0_{s,m,N})^{\text{act}}] = 0$ and $|\text{supp}(x)| = s$. Setting $y = Ax$ completes the proof. $\square$

22. PROOF OF THEOREM 8.1 AND THEOREM 8.3

We first prove Theorem 8.3 and then Theorem 8.1. For both theorems the proof relies on the points $x^{k,\delta} \in \mathbb{R}^{N_0}$, defined for $k \in \mathbb{N}$, $\delta \geq 0$, $\kappa \in [1/4, 3/4]$, and $a \in [1/2, 1]$ as follows:

$$x^{k,\delta} = \begin{cases} (a(k + 1 - \kappa)^{-1}, 0, \ldots, 0), & \text{if } k \text{ is odd} \\ (a(k + 1 - \kappa)^{-1}, \delta, 0, 0, \ldots, 0), & \text{if } k \text{ is even} \end{cases}. \tag{22.1}$$

The following two lemmas will be useful in both proofs.

Lemma 22.1. *Let $a \in [1/2, 1]$ and define $f_a : \mathbb{R}^{N_0} \to \{0, 1\}$ by $f_a(x) = 1$ if $\lceil a/x_1 \rceil$ is an odd integer and $f_a(x) = 0$ otherwise. Let $\kappa \in [1/4, 3/4]$ and $\delta \geq 0$, and consider the points $x^{k,\delta}$ as given in (22.1). Then, for every $K \in \mathbb{N}$, we have $\{x^{1,\delta}, \ldots, x^{K,\delta}\} \subset \mathcal{S}^{f_a}_{\varepsilon'(K)}$, where $\varepsilon'(n) := [(4n + 3)(4n + 4)]^{-1}$.*

Proof. We must verify that $\min_{i \neq j} \|x^{i,\delta} - x^{j,\delta}\|_\infty \geq 2\varepsilon'(r)$ and that for $k \leq K$ and vectors $y \in \mathbb{R}^{N_0}$ with $\|y\|_\infty < \varepsilon'(r)$ we have $f_a(x^{k,\delta} + y) = f_a(x^{k,\delta})$.

For the first part, note that for distinct $i, j \leq K$ we have

$$\|x^{i,\delta} - x^{j,\delta}\|_\infty \geq \left| \frac{a}{i + 1 - \kappa} - \frac{a}{j + 1 - \kappa} \right| = \frac{|a(j - i)|}{(i + 1 - \kappa)(j + 1 - \kappa)} \geq \frac{1}{2(K + 1 - \kappa)(K - \kappa)} \tag{22.2}$$

since $a|j - i| \geq a \geq 1/2$ and the condition that $i, j \leq K$ with at least one bounded by $K - 1$ implies that $(i + 1 - \kappa)^{-1}(j + 1 - \kappa)^{-1} \geq (K + 1 - \kappa)^{-1}(K - \kappa)^{-1}$. Since $\kappa \geq 1/4$, we get $\|x^{i,\delta} - x^{j,\delta}\|_\infty \geq [2(K + 1 - 1/4)(K - 1/4)]^{-1} \geq 2\varepsilon'(K)$.

Next, we let $k \leq K$ and $y \in \mathbb{R}^{N_0}$ be such that $\|y\|_\infty \leq \varepsilon'(K)$. We will establish that $f_a(x^{k,\delta} + y) = f_a(x^{k,\delta})$. Since $k \leq K$, we have

$$\frac{a(1 - \kappa)}{(k + 1 - \kappa)k} > \frac{1}{(4K + 3)(2K + 2)} \geq y_1 \geq \frac{-1}{(4K + 3)(2K + 2)} \geq \frac{-a\kappa}{(k + 1 - \kappa)(k + 1)}.$$

We claim that this implies $a(x_1^{k,\delta} + y_1)^{-1} \in (k, k + 1]$. For the upper bound, note that

$$\frac{y_1}{a} \geq \frac{-\kappa}{(k + 1 - \kappa)(k + 1)} = \frac{1}{k + 1} - \frac{1}{k + 1 - \kappa} = \frac{1}{k + 1} - \frac{x_1^{k,\delta}}{a}.$$

Similarly, for the lower bound, we have

$$\frac{y_1}{a} < \frac{1-\kappa}{k(k+1-\kappa)} = k^{-1}\left(\frac{k+1-\kappa}{k+1-\kappa} - \frac{k}{k+1-\kappa}\right) = \frac{1}{k} - \frac{x_1^{k,\delta}}{a}.$$

Therefore $\lceil a/(x_1^{k,\delta}+y_1)\rceil = k+1$. Thus, for all $\|y\|_\infty < \varepsilon'(K)$, we have $f_a(x^{k,\delta}+y) = f_a(x^{k,\delta}) = 1$ for even k and $f_a(x^{k,\delta}+y) = f_a(x^{k,\delta}) = 0$ for odd k, therefore establishing $\{x^{1,\delta},\ldots,x^{K,\delta}\} \subset \mathcal{S}_{\varepsilon'(K)}^{f_a}$. $\quad\square$

Lemma 22.2. *Fix* $m,n \in \mathbb{N}$, $A \in \mathbb{R}^{N\times N_0}$, $B \in \mathbb{R}^{m\times N}$, *and* $w \in \mathbb{R}^N$. *Suppose that*

$$R = \{x^q, x^{q+1}, x^{q+2}, \ldots, x^{q+r-1}\} \subset \mathbb{R}^{N_0}$$

is a set such that $|R|$ *is divisible by* $N+1$, *the sequence* $\{x_1^k\}_{k=q}^{q+r-1}$ *is strictly increasing and* $x_j^k = 0$ *for* $j > 1$ *and all* k. *Then there exist a matrix* $C \in \mathbb{R}^{m\times N_0}$, *a vector* $v \in \mathbb{R}^m$, *and a set* $S \subseteq R$ *of the form* $S = \{x^s, x^{s+1}, \ldots, x^{s+t-1}\}$ *such that* $|S| \geq |R|/(N+1)$ *and* $B\rho(Ax+w) = Cx+v$, *for all* $x \in S$.

Proof. Write $B = (b_{j,k})_{j=1,k=1}^{j=m,k=N}$, $A = (a_{j,k})_{j=1,k=1}^{j=N,k=N_0}$. We claim that the set SS defined by

$$SS = \{(\mathrm{sgn}(a_{1,1}x_1 + w_1), \mathrm{sgn}(a_{2,1}x_1 + w_2), \ldots, \mathrm{sgn}(a_{N,1}x_1 + w_N)) \mid x \in R\}.$$

contains at most $N+1$ (unique) elements, i.e., $|SS| \leq N+1$, where we define $\mathrm{sgn}(x) = 1$ for $x \geq 0$ and $\mathrm{sgn}(x) = -1$ for $x < 0$. To see this, note that if we allow x_1 to vary over $\mathbb{R}$, then each of the lines $y = a_{1,1}x_1 + w_1$, $y = a_{2,1}x_1 + w_2, \ldots, y = a_{N,1}x_1 + w_N$ intersect the line $y = 0$ at most once. Between each of these intersections the vector $(\mathrm{sgn}(a_{1,1}x_1 + w_1), \mathrm{sgn}(a_{2,1}x_1 + w_2), \ldots, \mathrm{sgn}(a_{N,1}x_1 + w_N))$ is constant. As there are at most N such intersections, the proof that $|SS| \leq N+1$ is complete by noticing that partitioning a line by at most N intersections gives at most $N+1$ regions between the intersections.

We can now define S. By the pigeonhole principle and the fact that $|SS| \leq N+1$, there exists a subset of R with cardinality at least $|R|/(N+1)$ such that the vector

$$\mathrm{sgn}(a_{\cdot,1}x_1 + w) = (\mathrm{sgn}(a_{1,1}x_1 + w_1), \mathrm{sgn}(a_{2,1}x_1 + w_2), \ldots, \mathrm{sgn}(a_{N,1}x_1 + w_N))$$

is constant over x in this subset. Let S be a subset of R of maximal cardinality satisfying this constant sign condition. Then clearly $|S| \geq |R|/(N+1)$. To see that $S = \{x^s, x^{s+1}, \ldots, x^{s+t-1}\}$, for some s and t, suppose by way of contradiction that no such s and t exist. Then there are j_1 and k_1 such that $j_1 + 1 < k_1$, $x^{j_1}, x^{k_1} \in S$ and $x^{j_1+1} \notin S$. But then, as S is assumed to be of maximal cardinality, there must be an ℓ for which $\mathrm{sgn}(a_{\ell,1}x_1^{j_1} + w_1) = \mathrm{sgn}(a_{\ell,1}x_1^{k_1} + w_1) \neq \mathrm{sgn}(a_{\ell,1}x_1^{j_1+1} + w_1)$. However, since $\{x_1^j\}_{j=j_1}^{k_1}$ is a strictly increasing sequence by assumption, we see that if $a_{\ell,1} \geq 0$ then $a_{\ell,1}x_1^j + w_1 \leq a_{\ell,1}x_1^{j+1} + w_1 \leq a_{\ell,1}x_1^k + w_1$ and similarly if $a_{\ell,1} < 0$ then $a_{\ell,1}x_1^j + w_1 > a_{\ell,1}x_1^{j+1} + w_1 > a_{\ell,1}x_1^k + w_1$ which is a contradiction. This establishes that $S = \{x^s, x^{s+1}, \ldots, x^{s+t-1}\}$, for some s and t.

We now show how to construct C and v. Recall that, for all $x \in S$, $x_2 = x_3 = \cdots = x_{N_0} = 0$, and so the i-th row of $B\rho(Ax+w)$ is given by $\sum_{j=1}^N b_{i,j}\rho(a_{j,1}x_1 + w_j)$. Since $\mathrm{sgn}(a_{j,1}x_1 + w_j)$ is constant over $x \in S$, we must have that for each j either $\rho(a_{j,1}x_1 + w_j) = 0$ or $\rho(a_{j,1}x_1 + w_j) = a_{j,1}x_1 + w_j$, for all $x \in S$. In the former case, we define $d_{i,j} = 0$ and $y_{i,j} = 0$ and in the latter case we define $d_{i,j} = b_{i,j}a_{j,1}$ and $y_{i,j} = b_{i,j}w_j$. Therefore, by construction, the i-th row of $B\rho(Ax+w)$ is given by $\sum_{j=1}^N (d_{i,j}x_1 + y_{i,j})$. Thus, defining the matrix $C = (c_{i,j})_{i=1,j=1}^{i=m,j=N_0}$ and the vector $v \in \mathbb{R}^m$ according to

$$c_{i,1} = \sum_{k=1}^N d_{i,k}, \quad c_{i,j} = 0, \text{ for } j > 1, \qquad \text{and} \qquad v_i = \sum_{k=1}^N y_{i,k}$$

immediately yields that the i-th row of $B\rho(Ax+w)$ satisfies $\sum_{k=1}^N (d_{i,k}x_1 + y_{i,k}) = \sum_{k=1}^N c_{i,k}x_k + v_i$. As i and $x \in S$ were arbitrary, this implies that $B\rho(Ax+w) = Cx+v$, for all $x \in S$, thereby concluding the proof of the lemma. $\quad\square$

Proof of Theorem 8.3. Theorem 8.3 is formally stated in Proposition 9.44, which we now prove. Let $a \in [1/2, 1]$ and define $f_a : \mathbb{R}^{N_0} \to \{0, 1\}$ by $f_a(x) = 1$ if $\lceil a/x_1 \rceil$ is an odd integer and $f_a(x) = 0$ otherwise. Note that this definition of f_a yields uncountably many unique classification functions. As with the other non-computability results, the result will follow by verifying the conditions (a) – (c) of Proposition 10.5. To this end, fix arbitrary $\kappa \in [1/4, 3/4]$, $r \geq 3 \cdot (N_1 + 1) \cdots (N_{L-1} + 1)$, and for $\delta \in [0, \varepsilon'(r))$ define $\mathcal{G}_\delta^\kappa := \{x^{1,\delta}, x^{2,\delta}, \ldots, x^{r,\delta}\}$, where the $x^{k,\delta}$ are as given in (22.1). We will prove the claim of the proposition with $\mathcal{T} = \mathcal{G}_0^\kappa$. As κ is chosen from an interval, this will provide uncountably many sets $\mathcal{T}$ for which the proposition holds. First notice that, due to Lemma 22.1, we have $\mathcal{G}_0^\kappa \in \mathcal{S}_{\varepsilon'(r)}^{f_a}$.

Now, fix a $\delta \in (0, \varepsilon'(r))$ and consider an arbitrary neural network $\phi_0 \in \mathcal{NN}_{\mathbf{N}, L}$ as well as an arbitrary

$$\phi_\delta \in \underset{\phi \in \mathcal{NN}_{\mathbf{N}, L}}{\mathrm{argmin}} \ \mathcal{R}\left(\{\phi(x^{j,\delta})\}_{j=1}^r, \{f_a(x^{j,\delta})\}_{j=1}^r\right) \tag{22.3}$$

We now make three claims about such ϕ_0 and ϕ_δ. The first relates to ϕ_δ.

Claim I: $\phi_\delta(x^{k,\delta}) = f_a(x^{k,\delta})$ for all $k \in \{1, \ldots, r\}$. To see this, we start by defining the neural network

$$\tilde{\phi} = W^L \rho W^{L-1} \rho W^{L-2} \ldots \rho W^1$$

where $W^\ell x = A^\ell x + b^\ell$ and $A^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$, $b^\ell \in \mathbb{R}^{N_\ell}$ are defined as follows: let $A_{1,1}^1 = 0$, $A_{1,2}^1 = \delta^{-1}$ and $A_{i,j}^1 = 0$ otherwise, and, for $\ell > 1$, $A_{1,1}^\ell = 1$ and $A_{i,j}^\ell = 0$ otherwise, and $b^\ell = 0$ for every ℓ. Clearly

$$W^1 x^{k,\delta} = \begin{cases} e_1 \in \mathbb{R}^{N_1} & \text{if } k \text{ is even} \\ \mathbf{0} \in \mathbb{R}^{N_1} & \text{if } k \text{ is odd} \end{cases}$$

and it is therefore easy to see that $\tilde{\phi}(x^{k,\delta}) = 1$ if k is even and $\tilde{\phi}(x^{k,\delta}) = 0$ if k is odd. Now, as $\|x^{k,\delta} - x^k\|_\infty \leq \delta < \varepsilon'(r)$, it follows from $\mathcal{G}_0^\kappa \in \mathcal{S}_{\varepsilon'(r)}^{f_a}$ that $f_a(x^{k,\delta}) = 1$ if k is even and $f_a(x^{k,\delta}) = 0$ if k is odd, and therefore $\tilde{\phi}(x^{k,\delta}) = f_a(x^{k,\delta})$ for all k. Since $\mathcal{R}(v, w) = 0$ if $v = w$, we must have that the objective function of (22.3) at the minimiser is at most 0. Thus, since $\mathcal{R}$ is non-negative, the objective function at a minimiser is exactly 0. However, since $\mathcal{R}(v, w) \neq 0$ if $v \neq w$, if there is a k such that $\phi_\delta(x^{k,\delta}) \neq f_a(x^{k,\delta})$ then the objective function evaluated at ϕ_δ would be greater than 0, contradicting the assumption (22.3) that ϕ_δ be a minimiser. This concludes the proof of Claim I.

The other two claims are about ϕ_0. In the following we write x^k for $x^{k,0}$ to lighten the notation.

Claim II: There exist a set

$$\mathcal{S} = \{x^s, x^{s+1}, x^{s+2}\} \subset \mathcal{G}_0^\kappa$$

for some $s \in \mathbb{N}$ with $s \leq r - 2$, a matrix $M \in \mathbb{R}^{1 \times N_0}$ and a $z \in \mathbb{R}$ such that, for all $x \in S$, we have $\phi_0(x) = Mx + z$. We proceed inductively by showing that there are sets $\mathcal{S}_\ell \subset \{x^1, x^2, \ldots, x^r\}$, matrices $M^\ell \in \mathbb{R}^{N_\ell \times N_0}$ and vectors $z^\ell \in \mathbb{R}^{N_\ell}$ for $\ell = 1, \ldots, L$ such that

   (i) $|\mathcal{S}_\ell| \geq 3 \cdot (N_\ell + 1) \cdots (N_{L-1} + 1)$,
   (ii) $\mathcal{S}_\ell = \{x^{s_\ell}, x^{s_\ell+1}, \ldots, x^{s_\ell+t_\ell}\}$ for some $s_\ell, t_\ell \in \mathbb{N}$.
   (iii) $\phi(x) = W^L \rho W^{L-1} \rho W^{L-2} \ldots W^{\ell+1} \rho(M^\ell x + z^\ell)$ whenever $x \in \mathcal{S}_\ell$, where the W^i are affine maps and ρ is applied coordinatewise.

The induction base is obvious by taking $\mathcal{S}^1 = \mathcal{G}_0^\kappa$, $M^1 = A^1$, and $z^1 = b^1$. The induction step will follow with the help of Lemma 22.2. Indeed, assuming the existence of $\mathcal{S}_\ell$, M^ℓ, and z^ℓ for some $\ell < L$, we apply Lemma 22.2 with $B = A^{\ell+1}, A = M^\ell, R = \mathcal{S}_\ell$ and $w = z^\ell$ to obtain some set $\mathcal{S}_{\ell+1}$, a matrix $M^{\ell+1}$ and a vector $v^{\ell+1}$ for which $A^{\ell+1}\rho(M^\ell x + z^\ell) = M^{\ell+1}x + v^{\ell+1}$, for $x \in \mathcal{S}_{\ell+1}$, and thus $W^{\ell+1}\rho(M^\ell x + z^\ell) = M^{\ell+1}x + z^{\ell+1}$, where we set $z^{\ell+1} = v^{\ell+1} + b^{\ell+1}$. This completes the proof of Claim II.

Our final claim is as follows:

Claim III: There exists a $x \in \mathcal{G}_0^\kappa$ such that $|\phi_0(x) - f_a(x)| \geq 1/2$. Suppose by way of contradiction that no such x exists. Then, using Claim II, we have $\phi_0(x^{s+1}) = aM_{1,1}(s+2-\kappa)^{-1} + z \in [\alpha, \beta]$ where

$$\alpha := (aM_{1,1}(s+1-\kappa)^{-1} + z) \wedge (aM_{1,1}(s+3-\kappa)^{-1} + z) = \phi_0(x^s) \wedge \phi_0(x^{s+2})$$

$$\beta := (aM_{1,1}(s+1-\kappa)^{-1} + z) \vee (aM_{1,1}(s+3-\kappa)^{-1} + z) = \phi_0(x^s) \vee \phi_0(x^{s+2})$$

since the function $t \mapsto aM_{1,1}t^{-1} + y_1$ is monotonic on $(0, \infty)$. Recall that $f_a(x^k) = 1$ if k is even and $f_a(x^k) = 0$ if k is odd. We now consider two cases depending on the parity of s. Suppose first that s is even. Then we have $f_a(x^s) = f_a(x^{s+2}) = 1$ and thus $|\phi_0(x) - f_a(x)| < 1/2$ for $x \in \mathcal{G}_0^\kappa$ implies that $\phi_0(x^s) > 1/2$ and $\phi_0(x^{s+2}) > 1/2$. But then $\alpha > 1/2$ and so $\phi_0(x^{s+1}) > 1/2$, contradicting $f_a(x^{s+1}) = 0$. If instead s is odd, we have $f_a(x^s) = f_a(x^{s+2}) = 0$ and thus $|\phi_0(x) - f_a(x)| < 1/2$ for $x \in \mathcal{G}_0^\kappa$ implies that $\phi_0(x^s) < 1/2$ and $\phi_0(x^{s+2}) < 1/2$. Therefore $\beta < 1/2$ and so $\phi_0(x^{s+1}) < 1/2$, contradicting $f_a(x^{s+1}) = 1$. This completes the proof of Claim III.

Recall that we chose $\mathcal{T} = \mathcal{G}_0^\kappa$. Now, in order to apply Proposition 10.5, for $n \in \mathbb{N}$ we define $\iota_n^1 = \iota^0 = (\{x^j\}_{j=1}^r, \{x^j\}_{j=1}^r) \in \Omega_{\mathcal{T}, \nu}$ as well as $\iota_n^2 = (\{x^{j, \nu/n}\}_{j=1}^r, \{x^{j, \nu/n}\}_{j=1}^r) \in \Omega_{\mathcal{T}, \nu}$. Moreover, define $S_1 = \Xi(\iota_0)$, and $S_2 = \bigcup_{n \in \mathbb{N}} \Xi(\iota_n^2)$. Note that for any $\{\phi_0(x^j)\}_{j=1}^r \in S_1$ we have by Claim III that $\max_{x^j \in \mathcal{G}_0^\kappa} |\phi_0(x^j) - f_a(x^j)| \geq 1/2$. However, for any $\{\phi_{\nu/n}(x^{j, \nu/n})\}_{j=1}^r \in S_2$ we have that $\phi_{\nu/n}$ satisfies (22.3) with $\delta = \nu/n$, and hence Claim I implies that $\phi_{\nu/n}(x^{j, \nu/n}) = f_a(x^{j, \nu/n})$, for all j and n. Since $\mathcal{G}_0^\kappa \in \mathcal{S}_{\varepsilon'(r)}^{f_a}$ and $\|x^{j, \nu/n} - x^j\|_\infty \leq \nu/n < \varepsilon'(r)$, it follows that $f_a(x^{j, \nu/n}) = f_a(x^j)$, for all j and n, and hence

$$\inf_{\xi_1 \in S^1, \xi_2 \in S^2} d_\mathcal{M}(\xi_1, \xi_2) \geq 1/2.$$

Hence, the conditions (a) – (c) of Proposition 10.5 hold, implying the existence of a $\hat{\Lambda}_{d,r,r} \in \mathcal{L}^1(\Lambda_{d,r,r})$ so that for the computational problem $\{\Xi, \Omega_{\mathcal{T}, \nu}, \mathbb{R}^r, \Lambda_{d,r,r}\}$ one has $\epsilon_{\mathbb{P}hB}^s(\mathrm{p}) \geq 1/4$, for all $\mathrm{p} \in [0, 1/2)$, thereby concluding the proof. $\square$

Lemma 22.3. *For $\gamma \in (1, 2)$, define the probability distribution $\mathcal{P} = \{\mathrm{p}_j\}_{j=1}^\infty$ on $\mathbb{N}$ by $\mathrm{p}_{2j-1} = \mathrm{p}_{2j} = \frac{1}{2} C_\zeta(\gamma) j^{-\gamma}$, for $j \in \mathbb{N}$, where $C_\zeta(\gamma) := \left(\sum_{j=1}^\infty j^{-\gamma} \right)^{-1}$ is a normalising factor.*

Fix $r \in \mathbb{N}$ and let $X_1, X_2, \ldots, X_r$ be i.i.d. random variables in $\mathbb{N}$ distributed according to $\mathcal{P}$. Next, consider the random set whose elements are the values of $X_1, X_2, \ldots, X_r$ and enumerate it as $S = \{Z_1 < Z_2 < \cdots < Z_N\}$ (note that N, the number of distinct elements of S, is an integer-valued random variable such that $N \leq r$). Then, setting $c_1 = (1 - e^{-C_\zeta(\gamma)})/2$ and $c_2 = C_\zeta(\gamma)/(\gamma - 1)$, we have

(i) $\mathbb{P}\left(N \geq c_1 r^{1/\gamma} \right) \geq 1 - c_1^{-2} r^{-(2/\gamma - 1)}$,

(ii) $\mathbb{P}(\max S \leq n) \geq 1 - c_2 r \lfloor n/2 \rfloor^{1-\gamma}$, *for all $n \in \mathbb{N}$, and*

(iii) $\mathbb{P}\left(\sum_{j=1}^{N-1} \chi_{\{Z_{j+1} - Z_j \text{ odd}\}} \leq n/5 \,\middle|\, N = n \right) \leq e^{-n/100}$, *for all integers n such that $10 \leq n \leq r$.*

Proof. For item (i), define the random variable M_r to be the number of different unique values taken by the random variables $\lceil X_1/2 \rceil, \ldots, \lceil X_r/2 \rceil$ and note that $\mathbb{P}(N < m) \leq \mathbb{P}(M_r < m)$, for $m \in \mathbb{N}$. Now, as the random variables $\lceil X_j/2 \rceil$, $j = 1, \ldots, r$, are i.i.d. and distributed according to the zeta distribution with parameter γ, it follows from [59, Lem. 4, Lem. 3] that $\mathbb{E}[M_r] > (1 - e^{-C_\zeta(\gamma)}) r^{1/\gamma}$ and $\sigma^2 := \mathrm{Var}[M_r] \leq \mathbb{E}[M_r] \leq r$, and hence Chebyshev's inequality yields

$$\mathbb{P}\left(N < \frac{1 - e^{-C_\zeta(\gamma)}}{2} r^{1/\gamma} \right) \leq \mathbb{P}\left(M_r < \frac{1 - e^{-C_\zeta(\gamma)}}{2} r^{1/\gamma} \right)$$

$$\leq \mathbb{P}\left(|M_r - \mathbb{E}[M_r]| > \frac{1 - e^{-C_\zeta(\gamma)}}{2\sigma} r^{1/\gamma} \cdot \sigma \right) \leq \left(\frac{1 - e^{-C_\zeta(\gamma)}}{2\sigma} r^{1/\gamma} \right)^{-2} \leq \frac{4 r^{-(2/\gamma - 1)}}{(1 - e^{-C_\zeta(\gamma)})^2},$$

which implies item (i).

The proof of item (ii) is simple. Note that $\{\max S \leq n\} = \bigcap_{j=1}^{r}\{X_j \leq n\}$ and, for each j,

$$\mathbb{P}(X_j \leq n) = \sum_{j=1}^{n} \mathrm{p}_j \geq \sum_{j=1}^{\lfloor n/2 \rfloor} C_\zeta(\gamma)j^{-\gamma} \geq 1 - C_\zeta(\gamma)\int_{\lfloor n/2 \rfloor}^{\infty} t^{-\gamma}\mathrm{d}t \geq 1 - \frac{C_\zeta(\gamma)}{\gamma - 1}\lfloor n/2 \rfloor^{1-\gamma},$$

and hence, as the X_j are independent,

$$\mathbb{P}(\max S \leq n) = \mathbb{P}(X_j \leq n)^r \geq \left(1 - \frac{C_\zeta(\gamma)}{\gamma - 1}\lfloor n/2 \rfloor^{1-\gamma}\right)^r \geq 1 - \frac{C_\zeta(\gamma)}{\gamma - 1}r\lfloor n/2 \rfloor^{1-\gamma}$$

where the last inequality follows by Bernoulli's inequality.

Item (iii) is somewhat more involved. We start by outlining the strategy: the set S may contain pairs of the form $(Z_j, Z_{j+1}) = (2i - 1, 2i)$, i.e., an odd natural number followed by the next even one. We will condition on the set of j where (Z_j, Z_{j+1}) is such a pair, as well as the specific value of Z_j.

More precisely, for fixed sets $\mathcal{I}$ and $\mathcal{J}$ with $|\mathcal{I}| = |\mathcal{J}|$, enumerated by

$$\mathcal{I} = \{i_1, i_2, \ldots, i_m\} \text{ and } \mathcal{J} = \{j_1, j_2, \ldots, j_m\},$$

let $\mathcal{A} = \{1, \ldots, N\} \setminus \big(\mathcal{J} \cup (\mathcal{J} + 1)\big)$. We will condition on the event $F_{\mathcal{I},\mathcal{J}}$ which occurs precisely when $N = n$, $(Z_{j_\ell}, Z_{j_\ell+1}) = (2i_\ell - 1, 2i_\ell)$ for $\ell \in \{1, 2, \ldots, m\}$, and, on the indices in $\mathcal{A}$, the set S contains no odd-even pairs, i.e., $(Z_\ell, Z_{\ell+1}) \notin \{(2i - 1, 2i) \,|\, i \in \mathbb{N}\}$ for all $\ell \in \mathcal{A}$. With varying $\mathcal{I}$ and $\mathcal{J}$, these sets $F_{\mathcal{I},\mathcal{J}}$ partition the event $\{N = n\}$.

The intuition behind this construction is as follows: conditional on $F_{\mathcal{I},\mathcal{J}}$, whenever $j \in \mathcal{J}$ we have $Z_{j+1} - Z_j = 1$ and hence $\chi_{\{Z_{j+1}-Z_j \text{ odd}\}} = 1$. Thus for sets $\mathcal{J}$ with $|\mathcal{J}| \geq n/5$, we are done. If instead $|\mathcal{J}|$ is small then $|\mathcal{A}|$ will be relatively large. For $a \in \mathcal{A}$, we will argue that every Z_a has equal probability of being an odd number or the even number following it, owing to the assumption that $\mathrm{p}_{2i-1} = \mathrm{p}_{2i}$ and the assumption that $(Z_a, Z_{a+1}) \notin \{(2i - 1, 2i) \,|\, i \in \mathbb{N}\}$, for all $i \in \mathbb{N}$. This will allow us to conclude that the indicators $\chi_{\{Z_a \text{ odd}\}}$, for $a \in \mathcal{A}$ are independent symmetric Bernoulli random variables. The desired bound will follow by an application of Hoeffding's inequality.

We are now ready to present the formal proof. If $r < 10$ there is nothing to prove, so assume that $r \geq 10$ and fix an n such that $10 \leq n \leq r$. Consider arbitrary sets $\mathcal{I} \subset \mathbb{N}$ and $\mathcal{J} \subset \{1, \ldots, n - 1\}$ so that

$$|\mathcal{I}| = |\mathcal{J}| = m < n \text{ and } \mathcal{J} \cap (\mathcal{J} + 1) = \varnothing, \tag{22.4}$$

where $\mathcal{J} + 1 := \{j + 1 \,|\, j \in \mathcal{J}\}$, and define $\mathcal{A} = \{1, \ldots, N\} \setminus \big(\mathcal{J} \cup (\mathcal{J} + 1)\big)$. Enumerate $\mathcal{I} = \{i_1 < \cdots < i_m\}$, $\mathcal{J} = \{j_1 < \cdots < j_m\}$, and $\mathcal{A} = \{a_1 < \cdots < a_{n-2m}\}$ and define the event

$$F_{\mathcal{I},\mathcal{J}} = \{N = n\} \cap \bigcap_{\ell=1}^{m}\{(Z_\ell, Z_{\ell+1}) = (2i_\ell - 1, 2i_\ell)\} \cap \bigcap_{\substack{a \in \mathcal{A}, a < n \\ i \in \mathbb{N}}}\{(Z_a, Z_{a+1}) \neq (2i - 1, 2i)\}.$$

Note that, for every $n \in \mathbb{N}$, we have

$$\{N = n\} = \bigcup_{\substack{\mathcal{I} \subset \mathbb{N}, \mathcal{J} \subset \{1, \ldots, n-1\} \\ \text{satisfying } (22.4)}} F_{\mathcal{I},\mathcal{J}}, \tag{22.5}$$

i.e., the events $F_{\mathcal{I},\mathcal{J}}$ for different $\mathcal{I}$ and $\mathcal{J}$ partition the event $\{N = n\}$, and thus our strategy will be to prove the bound $\mathbb{P}\left(\sum_{j=1}^{N-1} \chi_{\{Z_{j+1}-Z_j \text{ odd}\}} \leq n/5 \,\Big|\, F_{\mathcal{I},\mathcal{J}}\right) \leq e^{-n/100}$ for each of these events.

The argument relies on bounding from below the number of indices j such that $Z_{j+1} - Z_j$ is odd. For $j \in \mathcal{J}$ this will be easy, as $Z_{j+1} - Z_j = 2i_j - (2i_j - 1) = 1$ is always odd, by definition of $F_{\mathcal{I},\mathcal{J}}$. For $j \in \mathcal{A}$, we will need the following claim which we prove last.

Claim: For any $\mathcal{I}, \mathcal{J}$, and $\mathcal{A}$ as above, the indicator random variables $\chi_{\{Z_a \text{ odd}\}}$, $a \in \mathcal{A}$, conditional on $F_{\mathcal{I},\mathcal{J}}$ are independent symmetric Bernoulli variables.

Armed with the claim, the counting argument is as follows. Note that, on the event $F_{\mathcal{I},\mathcal{J}}$, for $k \in \{1,\ldots,n-2m\}$ such that $a_{k+1} > a_k + 1$, we have that $\{Z_{a_k},\ldots,Z_{a_{k+1}}\} = \{Z_{a_k} < 2i_t - 1 < 2i_t < 2i_{t+1} - 1 < 2i_{t+1} < \cdots < 2i_{t+s} - 1 < 2i_{t+s} < Z_{a_{k+1}}\}$, for $s = |\mathcal{J} \cap \{a_k,\ldots,a_{k+1}-1\}|$ and some $t \in \{1,\ldots,m\}$, and hence

$$\sum_{\ell=a_k}^{a_{k+1}-1} \chi_{\{Z_{\ell+1}-Z_\ell \text{ odd}\}} \geq \chi_{\{2i_t-1-Z_{a_k} \text{ odd}\}} + \sum_{\ell=1}^{s} \chi_{\{(2i_{t+\ell})-(2i_{t+\ell}-1) \text{ odd}\}} + \chi_{\{Z_{a_{k+1}}-2i_{t+s} \text{ odd}\}}$$

$$\geq |\mathcal{J} \cap \{a_k,\ldots,a_{k+1}-1\}| + \chi_{\{Z_{a_{k+1}}-Z_{a_k} \text{ even}\}}, \tag{22.6}$$

where we used the simple observation that $\chi_{\{2i_t-1-Z_{a_k} \text{ odd}\}} + \chi_{\{Z_{a_{k+1}}-2i_{t+s} \text{ odd}\}} \geq \chi_{\{Z_{a_{k+1}}-Z_{a_k} \text{ even}\}}$. This motivates defining random variables E_{a_k}, $k \in \{1,\ldots,n-2m-1\}$ on the event $F_{\mathcal{I},\mathcal{J}}$ according to

$$E_{a_k} = \begin{cases} 1, & Z_{a_k+1} - Z_{a_k} \text{ is odd} \\ 0, & Z_{a_k+1} - Z_{a_k} \text{ is even} \end{cases}, \quad \text{for } k \text{ s.t. } a_{k+1} = a_k + 1, \text{ and}$$

$$E_{a_k} = \begin{cases} 0, & Z_{a_{k+1}} - Z_{a_k} \text{ is odd} \\ 1, & Z_{a_{k+1}} - Z_{a_k} \text{ is even} \end{cases}, \quad \text{for } k \text{ s.t. } a_{k+1} > a_k + 1,$$

which, as a consequence of the Claim, are themselves independent symmetric Bernoulli random variables. Thus, writing $U := \sum_{k=1}^{N-1} \chi_{\{Z_{k+1}-Z_k \text{ odd}\}}$, on the event $F_{\mathcal{I},\mathcal{J}}$ we have

$$U = \sum_{\ell < a_1 \text{ or } \ell \geq a_{n-2m}} \chi_{\{Z_{\ell+1}-Z_\ell \text{ odd}\}} + \sum_{k=1}^{n-2m-1} \sum_{\ell=a_k}^{a_{k+1}-1} \chi_{\{Z_{\ell+1}-Z_\ell \text{ odd}\}}$$

$$\geq |\mathcal{J} \cap \{1,\ldots,a_1-1\}| + |\mathcal{J} \cap \{a_{n-2m},\ldots,n\}|$$

$$+ \sum_{\substack{1 \leq k \leq n-2m-1 \\ a_{k+1}=a_k+1}} \chi_{\{Z_{k+1}-Z_k \text{ odd}\}} + \sum_{\substack{1 \leq k \leq n-2m-1 \\ a_{k+1}>a_k+1}} \sum_{\ell=a_k}^{a_{k+1}-1} \chi_{\{Z_{\ell+1}-Z_\ell \text{ odd}\}}$$

$$\geq |\mathcal{J} \cap \{1,\ldots,a_1-1\}| + |\mathcal{J} \cap \{a_{n-2m},\ldots,n\}|$$

$$+ \sum_{\substack{1 \leq k \leq n-2m-1 \\ a_{k+1}=a_k+1}} E_{a_k} + \sum_{\substack{1 \leq k \leq n-2m-1 \\ a_{k+1}>a_k+1}} \left(|\mathcal{J} \cap \{a_k,\ldots,a_{k+1}-1\}| + E_{a_k}\right)$$

$$= |\mathcal{J}| + \sum_{k=1}^{n-2m-1} E_{a_k} = m + \sum_{k=1}^{n-2m-1} E_{a_k}, \tag{22.7}$$

where the second inequality is due to (22.6).

Now, for sets $\mathcal{I} \subset \mathbb{N}$ and $\mathcal{J} \subset \{1,\ldots,n-1\}$ satisfying (22.4) as well as $m = |\mathcal{I}| = |\mathcal{J}| \leq n/5$, we have that (22.7) implies $U \geq \sum_{j=1}^{n-2m-1} E_{a_k}$, which together with Hoeffding's inequality yields

$$\mathbb{P}\left(U \leq n/5 \,\Big|\, F_{\mathcal{I},\mathcal{J}}\right) \leq \mathbb{P}\left(\sum_{j=1}^{n-2m-1} E_{a_k} \leq n/5 \,\Big|\, F_{\mathcal{I},\mathcal{J}}\right)$$

$$\leq \exp\left(-2\left(\frac{1}{2} - \frac{n/5}{n-2m-1}\right)^2 (n-2m-1)\right) \leq \exp\left(-n/100\right)$$

where in the last inequality we used $n - 2m - 1 \geq n/2$ (recall that $n \geq 10$). On the other hand, in the case when $m = |\mathcal{I}| = |\mathcal{J}| > n/5$ we have $\mathbb{P}\left(U \leq n/5 \,\Big|\, F_{\mathcal{I},\mathcal{J}}\right) = 0$ directly from (22.7).

Therefore, we have shown that for any $\mathcal{I}, \mathcal{J}$ satisfying (22.4), $\mathbb{P}\left(U \leq n/5 \,\middle|\, F_{\mathcal{I},\mathcal{J}}\right) \leq \exp(-n/100)$ and so using (22.5)

$$\mathbb{P}\left(U \leq n/5, N = n\right) = \sum_{\substack{\mathcal{I} \subset \mathbb{N}, \mathcal{J} \subset \{1,\ldots,n-1\} \\ \text{satisfying (22.4)}}} \mathbb{P}\left(U \leq n/5 \,\middle|\, F_{\mathcal{I},\mathcal{J}}\right) \mathbb{P}(F_{\mathcal{I},\mathcal{J}})$$

$$\leq \exp\left(-n/100\right) \mathbb{P}\left(\bigcup_{\substack{\mathcal{I} \subset \mathbb{N}, \mathcal{J} \subset \{1,\ldots,n-1\} \\ \text{satisfying (22.4)}}} F_{\mathcal{I},\mathcal{J}}\right) = \exp\left(-n/100\right) \mathbb{P}(N = n),$$

which yields the desired bound after dividing both sides by $\mathbb{P}(N = n)$.

It remains to prove the Claim. To this end, fix n, $\mathcal{I} = \{i_1 < \cdots < i_m\}$, $\mathcal{J} = \{j_1 < \cdots < j_m\}$, and $\mathcal{A} = \{a_1 < \cdots < a_{n-2m}\}$ satisfying (22.4). Then, conditional on $F_{\mathcal{I},\mathcal{J}}$ we can write $Z_a = 2\lceil Z_a/2 \rceil - \chi_{\{Z_a \text{ odd}\}}$, for $a \in \mathcal{A}$, where the $\chi_{\{Z_a \text{ odd}\}}$ are random variables taking values in $\{0, 1\}$ and the $\lceil Z_a/2 \rceil$ are random variables taking values in $\mathbb{N} \setminus \mathcal{I}$ and moreover $\lceil Z_{a_1}/2 \rceil < \cdots < \lceil Z_{a_{n-2m}}/2 \rceil$. Now, for a set $\mathcal{U} = \{u_1 < \cdots < u_{n-2m}\} \subset \mathbb{N} \setminus \mathcal{I}$ denote $\mathring{F}_{\mathcal{U}} = \bigcap_{j=1}^{n-2m} \{\lceil Z_{a_j}/2 \rceil = u_j\}$ so that

$$\mathbb{P}\left(\{\chi_{\{Z_{a_1} \text{ odd}\}} = b_1, \ldots, \chi_{\{Z_{a_{n-2m}} \text{ odd}\}} = b_{n-2m}\} \,\middle|\, F_{\mathcal{I},\mathcal{J}}\right)$$

$$= \sum_{\mathcal{U} \subset \mathbb{N} \setminus \mathcal{I}} \mathbb{P}\left(\{\chi_{\{Z_{a_1} \text{ odd}\}} = b_1, \ldots, \chi_{\{Z_{a_{n-2m}} \text{ odd}\}} = b_{n-2m}\} \,\middle|\, F_{\mathcal{I},\mathcal{J}} \cap \mathring{F}_{\mathcal{U}}\right) \mathbb{P}(\mathring{F}_{\mathcal{U}} \,|\, F_{\mathcal{I},\mathcal{J}})$$

$$= \sum_{\mathcal{U} \subset \mathbb{N} \setminus \mathcal{I}} \mathbb{P}\left(\{\chi_{\{Z_{a_1} \text{ odd}\}} = 0, \ldots, \chi_{\{Z_{a_{n-2m}} \text{ odd}\}} = 0\} \,\middle|\, F_{\mathcal{I},\mathcal{J}} \cap \mathring{F}_{\mathcal{U}}\right) \mathbb{P}(\mathring{F}_{\mathcal{U}} \,|\, F_{\mathcal{I},\mathcal{J}}) \qquad (22.8)$$

$$= \mathbb{P}\left(\{\chi_{\{Z_{a_1} \text{ odd}\}} = 0, \ldots, \chi_{\{Z_{a_{n-2m}} \text{ odd}\}} = 0\} \,\middle|\, F_{\mathcal{I},\mathcal{J}}\right),$$

for all $b \in \{0,1\}^{n-2m}$, where in (22.8) we used the fact that $\mathrm{p}_{2j-1} = \mathrm{p}_{2j}$, for all $j \in \mathbb{N}$. It hence follows that the $\chi_{\{Z_{a_j} \text{ odd}\}}$, $1 \leq j \leq n - 2m$, conditional on $F_{\mathcal{I},\mathcal{J}}$ are independent symmetric Bernoulli variables, establishing the Claim. $\square$

Lemma 22.4. *Fix an even $K \in \mathbb{N}$ and let $\{\alpha_j\}_{j=1}^K$ be such that $0 < \alpha_{j+1} < \alpha_j < 1$ for all $1 \leq j \leq K - 1$. Furthermore, let $N_0 \in \mathbb{N}$. Then there exists a neural network $\psi : \mathbb{R}^{N_0} \to \mathbb{R}$ with the ReLU nonlinearity $\rho(t) = \max\{0, t\}$ such that*

$$\psi(x) = \begin{cases} 0 & \text{whenever } x_1 \in [\alpha_{4\ell-2}, \alpha_{4\ell-3}] \\ 1 & \text{whenever } x_1 \in [\alpha_{4\ell}, \alpha_{4\ell-1}] \end{cases}, \qquad \text{for all } x \in \mathbb{R}^{N_0}. \qquad (22.9)$$

Proof. We may w.l.o.g. assume that K is divisible by 4. Indeed, if K is not divisible by 4, we can extend the sequence $\{\alpha_j\}_{j=1}^K$ by adjoining two new elements (say $\alpha_K/2$ and $\alpha_K/4$) at the end of the sequence. We additionally set $\alpha_{K+1} = 0$ for convenience. Now, for $\ell \in \{1, \ldots, K/4\}$, define the single-layer neural network

$$\psi_\ell(x) = (\alpha_{4\ell-2} - \alpha_{4\ell-1})^{-1} \left(\rho(\alpha_{4\ell-2} - x_1) - \rho(\alpha_{4\ell-1} - x_1)\right)$$

$$- (\alpha_{4\ell} - \alpha_{4\ell+1})^{-1} \left(\rho(\alpha_{4\ell} - x_1) - \rho(\alpha_{4\ell+1} - x_1)\right), \quad \text{for } x \in \mathbb{R}^{N_0}.$$

One now easily verifies that $\psi_\ell(x) = 1$ whenever $x_1 \in [\alpha_{4\ell}, \alpha_{4\ell-1}]$ and $\psi_\ell(x) = 0$ whenever $x_1 \in \mathbb{R} \setminus (\alpha_{4\ell+1}, \alpha_{4\ell-2})$. Hence, setting $\psi(x) = \sum_{\ell=1}^{K/4} \psi_\ell(x)$ yields the desired network. $\square$

Proof of Theorem 8.1. The proof relies on the use Lemma 22.3 to show that the training and validation sets have various desired properties with large probability, after which the proof stays close to the proof of Theorem 8.3.

To start with we define the uncountable collection of functions $f_a : \mathbb{R}^{N_0} \to \{0, 1\}$ for $a \in [1/2, 1]$ by $f_a(x) = 1$ if $\lceil a/x_1 \rceil$ is an odd integer and $f_a(x) = 0$ otherwise. For $\delta \in (0, \epsilon)$, $\kappa \in [1/4, 3/4]$, and $k \in \mathbb{N}$ define $x^{k,\delta}$ according to (22.1).

Let c_1, c_2, and $C_\zeta(3/2)$ be the constants defined in Lemma 22.3 with γ set to $3/2$. We choose the constant C so that the following hold:

$$C \geq 4c_1^{-6}, \tag{22.10}$$

$$C \geq (200\log(8)^{3/2}c_1^{-3/2}), \text{ and} \tag{22.11}$$

$$C \geq 4 \cdot (8c_2)^2. \tag{22.12}$$

Next, let $\mathrm{p}_{2j-1} = \mathrm{p}_{2j} = C_\zeta(3/2)j^{-3/2}$, for $j \in \mathbb{N}$, and define the distribution $\mathcal{D}$ on $[0,1]^{N_0}$ according to

$$X \sim \mathcal{D} \quad \Longleftrightarrow \quad \mathbb{P}(X = x) = \begin{cases} \mathrm{p}_k, & \text{if } x = x^{k,\delta} \\ 0, & \text{else} \end{cases}.$$

As κ and δ are selected from intervals, this gives us uncountably many distributions.

Let $\mathcal{T} = \{x^1, \ldots, x^r\}$ and $\mathcal{V} = \{y^1, \ldots, y^s\}$ be the random multisets drawn from this distribution as in the statement of the theorem.

Since $C/2 \geq 2 \cdot (8c_2)^2$ (by (22.12)) and $Cr^2/(2p^2) \geq 4c_1^{-6}/2 \geq 2$ (by (22.10) and the facts that $r/p \geq 1$ and $c_1^{-1} \geq 1$) we obtain $C^2r^2/p^2 \geq 2[(8c_2r/p)^2 + 1]$ and thus item (ii) of Lemma 22.3 with $\gamma = 3/2$ yields

$$\mathbb{P}\big(\max\{k \in \mathbb{N} \mid x^{k,\delta} \in \mathcal{T}\} \leq \lceil C(r/\mathrm{p})^2\rceil\big) \geq \mathbb{P}\Big(\max\{k \in \mathbb{N} \mid x^{k,\delta} \in \mathcal{T}\} \leq 2\big((8c_2r/\mathrm{p})^2 + 1\big)\Big)$$

$$\geq 1 - \frac{c_2r}{\lfloor(8c_2r/\mathrm{p})^2 + 1\rfloor^{1/2}} \geq 1 - \mathrm{p}/8. \tag{22.13}$$

On the other hand, (22.11) yields $C^2t^2 \geq (4\lceil t\rceil + 3)(4\lceil t\rceil + 4)$, for all $t \geq 1$. Applying this inequality with $t = C(r/\mathrm{p})^2 \geq 1$, we deduce

$$\varepsilon[(r \vee s)/\mathrm{p}] \leq C^{-2}(C(r/\mathrm{p})^2)^{-2} \leq [(4\lceil C(r/\mathrm{p})^2\rceil + 3)(4\lceil C(r/\mathrm{p})^2\rceil + 4)]^{-1} = \varepsilon'\big(C(r/\mathrm{p})^2\big),$$

where $\varepsilon'(n) = [(4n + 3)(4n + 4)]^{-1}$. Therefore, on the event $\{\max\{k \in \mathbb{N} \mid x^{k,\delta} \in \mathcal{T}\} \leq \lceil C(r/\mathrm{p})^2\rceil\}$, Lemma 22.1 yields $\mathcal{T} \subset \{x^{1,\delta}, \ldots, x^{\lceil C(r/\mathrm{p})^2\rceil,\delta}\} \subset \mathcal{S}^{f_a}_{\varepsilon'(\lceil C(r/\mathrm{p})^2\rceil)} \subset \mathcal{S}^f_{\varepsilon((r\vee s)/\mathrm{p})}$, and hence (22.13) implies

$$\mathbb{P}(\mathcal{T} \in \mathcal{S}^f_{\varepsilon((r\vee s)/\mathrm{p})}) \geq 1 - \mathrm{p}/8. \tag{22.14}$$

A completely analogous argument also yields

$$\mathbb{P}(\mathcal{V} \in \mathcal{S}^f_{\varepsilon((r\vee s)/\mathrm{p})}) \geq 1 - \mathrm{p}/8. \tag{22.15}$$

Now, let $S = \{z_1^1, \ldots, z_1^{|S|}\}$ denote the set of (unique) first coordinates of the elements of $\mathcal{T}$, enumerated so that $z_1^1 > z_1^2 > \cdots > z_1^{|S|}$, and define $z^j = z_1^j \oplus \mathbf{0}_{N_0-1} \in \mathbb{R}^{N_0}$, for $j = 1, \ldots, |S|$. Note that, for every $j \in \{1, \ldots, r\}$ such that $x_1^j = z_1^j$, we have $f_a(x^j) = f_a(z^j)$. By item (i) of Lemma 22.3 with $\gamma = 3/2$, we have

$$\mathbb{P}\big(|S| \geq c_1 r^{2/3}\big) \geq 1 - c_1^{-2}r^{-1/3} > 1 - \mathrm{p}/4, \tag{22.16}$$

where the final bound follows because $r \geq Cp^{-3}$ (which, in turn, is due to the assumption (8.3)) and (22.10).

Now, writing $N_{\mathrm{prod}} := (N_1 + 1) \cdots (N_{L-1} + 1)$, by the assumption (8.3) and (22.11), we obtain

$$\lfloor c_1 r^{2/3}\rfloor \geq \lfloor C^{2/3}c_1 N_{\mathrm{prod}}\rfloor \geq \lfloor 1000^{2/3}\log(8)^2 N_{\mathrm{prod}}\rfloor \geq 15 N_{\mathrm{prod}}.$$

Therefore we can apply item (iii) of Lemma 22.3 to see that

$$
\mathbb{P}\Big(\sum_{i=1}^{|S|-1} \chi_{\{f_a(z^{j+1}) \neq f_a(z^j)\}} > 3 N_{\text{prod}} \Big)
$$

$$
\geq \sum_{n=\lfloor c_1 r^{2/3} \rfloor}^{r} \mathbb{P}\Big(\sum_{j=1}^{n-1} \chi_{\{f_a(z^{j+1}) \neq f_a(z^j)\}} > \frac{n}{5} \,\Big|\, |S| = n \Big) \mathbb{P}(|\mathcal{T}| = n)
$$

$$
\geq \Big(1 - \exp\big(-\lfloor c_1 r^{2/3} \rfloor / 100 \big) \Big) \cdot \mathbb{P}(|S| \geq \lfloor c_1 r^{2/3} \rfloor) \geq \Big(1 - e^{1/100} \mathrm{p}/8 \Big)(1 - p/4) > 1 - \mathrm{p}/2
$$

$$(22.17)$$

where the penultimate inequality follows from (22.16) and the bound $r \geq C\mathrm{p}^{-3} > \big(100 c_1^{-1} \log(8/\mathrm{p})\big)^{3/2}$ (which, in turn, is due to (22.11), the bound $\log(8/p) \leq 2\log(8)\log(\mathrm{p})$ and the bound $\log(t) \leq t$ for $t \geq 1$).

Therefore, combining (22.16), (22.14), (22.15), and (22.17), the probability that all of

$$
|S| \geq c_1 r^{2/3}, \quad \mathcal{T}, \mathcal{V} \in \mathcal{S}^f_{\varepsilon((r \vee s)/\mathrm{p})}, \quad \text{and} \quad \sum_{j=1}^{|S|-1} \chi_{\{f_a(z^{j+1}) \neq f_a(z^j)\}} > 3 N_{\text{prod}} \tag{22.18}
$$

occur is at least $1 - (\mathrm{p}/4 + \mathrm{p}/8 + \mathrm{p}/8 + \mathrm{p}/2) = 1 - \mathrm{p}$. We assume this this event occurs throughout the remainder of the proof.

We are now ready to construct ϕ satisfying (8.4). As in the proof of Theorem 8.3, we set $\phi = W^L \rho W^{L-1} \rho W^{L-2} \ldots \rho W^1$ where $W^\ell x = A^\ell x + b^\ell$ and $A^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$, $b^\ell \in \mathbb{R}^{N_\ell}$ are given as follows: $A^1_{1,1} = 0$, $A^1_{1,2} = 1/\delta$ and $A^1_{i,j} = 0$ otherwise, $A^\ell_{1,1} = 1$ and $A^\ell_{i,j} = 0$ otherwise, for $\ell > 1$, and finally $b^\ell = 0$ for every ℓ. Clearly

$$
W^1(x^{k,\delta}) = \begin{cases} e_1 \in \mathbb{R}^{N_1} & \text{if } k \text{ is even} \\ \mathbf{0} \in \mathbb{R}^{N_1} & \text{if } k \text{ is odd} \end{cases},
$$

and therefore we have as before that $\phi(x) = f_a(x)$ for all $x \in \mathcal{T} \cup \mathcal{V}$, thereby establishing (8.4).

Our next task will be to show that if $\hat{\phi} \in \mathcal{NN}_{\mathbf{N},L}$ then there exist uncountably many $v \in \mathbb{R}^{N_0}$ such

$$
|\hat{\phi}(v) - f_a(v)| \geq 1/2, \quad \|v - x\|_\infty < \epsilon \qquad \text{for some } x \in \mathcal{T}.
$$

To this end, note that (22.18) implies that there exist natural numbers $k_1 < k_2 < \cdots < k_{3N_{\text{prod}}}$ such that $x^{k_i,\delta} \in \mathcal{T}$, for all $i \in \{1, \ldots, 3N_{\text{prod}}\}$, as well as $x_1^{k_i,\delta} = z_1^i > z_1^{i+1} = x_1^{k_{i+1},\delta}$ and moreover $f_a(x^{k_i,\delta}) = f_a(z^i) \neq f_a(z^{i+1}) = f_a(x^{k_{i+1},\delta})$, for all $i \in \{1, \ldots, 3N_{\text{prod}} - 1\}$.

Now, consider $\kappa' \in [1/4, 3/4]$ for which $v^k := (a(k+1-\kappa')^{-1}, 0, \ldots, 0)$ satisfy $\|v^k - x^{k,\delta}\|_\infty < \epsilon$, for all $k \leq r$. The existence of such v^k is guaranteed by the assumption that $\delta \in (0, \epsilon)$. Then clearly $\|v^{k_i} - x^{k_i,\delta}\|_\infty < \epsilon$, for all $i \leq N_{\text{prod}}$. Moreover, for every such κ', since $N_{\text{prod}} = (N_1 + 1) \cdots (N_{L-1} + 1)$ we can argue exactly as in Claim III in the proof of Theorem 8.3 to see that there exists a $k' \in \{k_1, \ldots, k_{3N_{\text{prod}}}\}$ (depending on κ') such that $|\hat{\phi}(v^{k'}) - f_a(v^{k'})| \geq 1/2$. As there are uncountably many such κ', this gives us uncountably many $v \in \mathbb{R}^{N_0}$ satisfying (8.5).

Finally, we must show the existence of ψ, which we do with the help of Lemma 22.4. To this end, we set $K = \lceil C(r/\mathrm{p})^2 \rceil$, define $\{\alpha_j\}_{j=1}^{2K}$ by $\alpha_{2k-1} = x_1^{k,\delta} + \varepsilon((r \vee s)/\mathrm{p})$, $\alpha_{2k} = x_1^{k,\delta} - \varepsilon((r \vee s)/\mathrm{p})$, for $k = 1, \ldots, K$, and claim that $\alpha_1 > \alpha_2 > \cdots > \alpha_{2K}$.

To show this, it suffices to prove that $x_1^{j+1,\delta} + \varepsilon((r \vee s)/\mathrm{p}) < x_1^{j,\delta} - \varepsilon((r \vee s)/\mathrm{p})$ for each $j = 1, \ldots, K-1$. To see this, we use (22.2). Indeed, from this equation we must have $x_1^{j,\delta} - x_1^{j+1,\delta} \geq (2(K+1-\kappa)(K-\kappa))^{-1}$. On the other hand, once again employing the result that $C^2 t^2 \geq (4\lceil t \rceil + 3)(4\lceil t \rceil + 4)$, for all $t \geq 1$, (which is a consequence of (22.11)) we obtain

$$
2\varepsilon((r \vee s)/\mathrm{p}) \leq 2C^{-2}(C(r/\mathrm{p})^2)^{-2} \leq 2[(4K+3)(4K+4)]^{-1} \leq [2(K+1-\kappa)(K-\kappa)]^{-1}.
$$

Now, let ψ be the network provided by Lemma 22.4 with this sequence $\{\alpha_j\}_{j=1}^{2K}$. To see that $\psi(x) = f_a(x)$ for all $x \in \mathcal{B}_{\varepsilon((r\vee s)/\mathrm{p})}^{\infty}(\mathcal{T} \cup \mathcal{V})$, note that, for every $x \in \mathcal{B}_{\varepsilon((r\vee s)/\mathrm{p})}^{\infty}(\mathcal{T} \cup \mathcal{V})$, there exists either an $x^{k,\delta} \in \mathcal{T} \cup \mathcal{V}$ such that $\|x^{k,\delta} - x\|_{\infty} \leq \varepsilon((r \vee s)/\mathrm{p})$. Then, as $\max\{\ell \in \mathbb{N} \,|\, x^{\ell,\delta} \in \mathcal{T}\} \leq \lceil C(r/\mathrm{p})^2 \rceil$ and $\max\{\ell \in \mathbb{N} \,|\, x^{\ell,\delta} \in \mathcal{V}\} \leq \lceil C(r/\mathrm{p})^2 \rceil$ occur, we have $k \leq K$, and so

$$\alpha_{2k} = x_1^{k,\delta} - \varepsilon((r \vee s)/\mathrm{p}) \leq x_1 \leq x_1^{k,\delta} + \varepsilon((r \vee s)/\mathrm{p}) = \alpha_{2k-1}.$$

Now, as ψ is constant on $[\alpha_{2k}, \alpha_{2k-1}]$ and equal to $f_a(x_1^{k,\delta})$ by (22.9), the definition of f_a and the definition of $x^{k,\delta}$, and f_a is constant on the same interval and equal to $f_a(x_1^{k,\delta})$ by (22.18) and the definition of $\mathcal{S}_{\varepsilon((r\vee s)/\mathrm{p})}^f$, we deduce that $\psi(x) = f_a(x)$, as desired. As this holds for every $x \in \mathcal{B}_{\varepsilon((r\vee s)/\mathrm{p})}^{\infty}(\mathcal{T} \cup \mathcal{V})$, the proof of the theorem is complete. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

APPENDIX A. EXACT SOLUTIONS TO SIMPLE CONVEX PROBLEMS

In this section we derive the exact solution to the introductory Lasso example (4.2) and prove the lemmas given in §11 on the solutions of the convex problems used in Theorems 3.4, 5.1, and 6.1. We use standard tools from convex analysis, most extensively the subdifferential $\partial f(x) := \{v \in \mathbb{R}^N \,|\, f(y) - f(x) \geq \langle y - x, v \rangle, \forall y \in D_f\}$, for $x \in D_f$, where $f : D_F \to (-\infty, +\infty]$ is a lower semi-continuous convex function on a closed convex domain $D_f \subset \mathbb{R}^N$. For a more detailed account of convex analysis see [60].

A.1. **Solution to** (4.2).

Lemma A.1. *Let $m = 3, N = 2$ and $\lambda \in (0, 1/\sqrt{3}\,]$, as well as*

$$A = \begin{pmatrix} \frac{1}{\sqrt{2}} - a & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} - a & -\frac{1}{\sqrt{2}} \\ 2a & 0 \end{pmatrix} \in \mathbb{R}^{3\times 2}, \quad y = \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \end{pmatrix}^T \in \mathbb{R}^3.$$

where $a > 0$. Then if $w^0 \in \mathbb{R}$ and $w^1 \in \mathbb{R}^2$ are such that

$$(w^0, w^1) \in \operatorname*{arg\,min}_{(x^0,x^1)\in\mathbb{R}\times\mathbb{R}^2} \frac{1}{2m}\|ADx^1 - x^0\mathbf{1}_m - y\|_2^2 + \lambda\|x^1\|_1$$

where D is the unique diagonal matrix such that every column of AD has norm $\sqrt{m}$, we have $w^0 = 0$, $Dw^1 = (1 - \sqrt{3}\lambda)e_2$.

Proof. Note that

$$(w^0, w^1) \in \operatorname*{arg\,min}_{(x^0,x^1)\in\mathbb{R}\times\mathbb{R}^2} \frac{1}{2m}\|ADx^1 - x^0\mathbf{1}_m - y\|_2^2 + \lambda\|x^1\|_1$$

if and only if

$$(w^0, \hat{w}^1) \in \operatorname*{arg\,min}_{(x^0,x^1)\in\mathbb{R}\times\mathbb{R}^2} \frac{1}{2m}\|Ax^1 - x^0\mathbf{1}_m - y\|_2^2 + \lambda\|D^{-1}x^1\|_1,$$

where $\hat{w}^1 = Dw^1$. Now, for all $(x^0, x^1) \in \mathbb{R} \times \mathbb{R}^2$, we have

$$\|Ax^1 - x^0\mathbf{1}_m - y\|_2^2 = \left[\left(\frac{1}{\sqrt{2}} - a\right)x_1^1 + \frac{x_2^1}{\sqrt{2}} - x^0 - \frac{1}{\sqrt{2}}\right]^2 + \left[\left(\frac{1}{\sqrt{2}} + a\right)x_1^1 + \frac{x_2^1}{\sqrt{2}} + x^0 - \frac{1}{\sqrt{2}}\right]^2$$
$$+ (2ax_1^1 - x^0)^2 = (x_1^1 + x_2^1 - 1)^2 + 6a^2(x_1^1)^2 + 3(x^0)^2$$

Note also that by definition of $D = \operatorname{diag}(d_{11}, d_{22})$ we have

$$d_{11} = \sqrt{3}\left[\left(\frac{1}{\sqrt{2}} - a\right)^2 + \left(\frac{1}{\sqrt{2}} + a\right)^2 + 4a^2\right]^{-\frac{1}{2}} = \sqrt{3}(1 + 6a^2)^{-\frac{1}{2}} < \sqrt{3}$$

and $d_{22} = \sqrt{3}$. Hence

$$\frac{1}{2m}\|Ax^1 - x^0\mathbf{1}_m - y\|_2^2 + \lambda\|D^{-1}x^1\|_1 \geq \frac{1}{6}\|Ax^1 - x^0\mathbf{1}_m - y\|_2^2 + \lambda\frac{\|x^1\|_1}{\sqrt{3}}$$

$$= \frac{1}{6}(x_1^1 + x_2^1 - 1)^2 + a^2(x_1^1)^2 + \frac{1}{2}(x^0)^2 + \lambda\frac{\|x^1\|_1}{\sqrt{3}} \geq \frac{1}{6}(x_1^1 + x_2^1 - 1)^2 + \frac{\lambda}{\sqrt{3}}(x_1^1 + x_2^1)$$

$$= \frac{1}{6}\left[\left(x_1^1 + x_2^1 - (1 - \sqrt{3}\lambda)\right)^2 - (1 - \sqrt{3}\lambda)^2 + 1\right] \geq 1 - (1 - \sqrt{3}\lambda)^2$$

These inequalities hold as equalities if and only if $x_1^1 = x^0 = 0$, $x_2^1 \geq 0$, and $x_1^1 + x_2^1 = 1 - \sqrt{3}\lambda$. Therefore, we must have $w^0 = 0$ and $Dw^1 = \hat{w}^1 = (1 - \sqrt{3}\lambda)e_2$, as desired. $\qquad\square$

A.2. **Proofs of results in §11.**

Proof of Lemma 11.1. Since both α and β are assumed to be positive, we have that for any feasible x (that is, x with $x \geq 0$ and $Ax = y^{\mathrm{A}}$),

$$\langle c, x \rangle \geq x_1 + x_2 + 1 \geq \frac{\alpha x_1 + \beta x_2}{\alpha \vee \beta} + 1 = \frac{y_1}{\alpha \vee \beta} + 1. \tag{A.1}$$

Thus $\min\{\langle c, x \rangle \mid x \geq 0, Ax = y^{\mathrm{A}}\} \geq y_1/(\alpha\vee\beta) + 1$. Moreover, all claimed minimisers in the statement of the lemma obey (A.1) as an equality and furthermore such x are feasible for the LP problem. Therefore $\min\{\langle c, x \rangle \mid x \geq 0, Ax = y^{\mathrm{A}}\} = y_1/(\alpha \vee \beta)$, and thus the solutions to $\Xi_{\mathrm{LP}}(y^{\mathrm{A}}, A)$ are precisely the vectors x for which every inequality in (A.1) is obeyed as an equality. More specifically, the following must occur:

- For the first inequality to be an equality we must have $x_3 = 0$, $x_4 = 1$, and $x_5 = \cdots = x_N = 0$.
- For the second inequality to be an equality we must have $x_2 = 0$ in the case $\alpha > \beta$ and $x_1 = 0$ in the case $\alpha < \beta$. In the case $\alpha = \beta$ this is always an equality.

It is easy to verify that the x satisfying these properties as well as the conditions $x \geq 0$ and $Ax = y^{\mathrm{A}}$ are exactly the claimed minimisers in the statement of the lemma. $\qquad\square$

Proof of Lemma 11.2. Let $x^{opt} = \frac{2\alpha y_1 - \lambda}{2\alpha^2}e_1$ if $\alpha \geq \beta$, and $x^{opt} = \frac{2\beta y_1 - \lambda}{2\beta^2}e_2$ if $\beta > \alpha$. Define a dual vector $p = Lx^{opt} - y^{\mathrm{L}} = -\frac{\lambda}{2(\alpha\vee\beta)}e_1 \in \mathbb{R}^m$, and note that then

$$-\frac{2}{\lambda}L^*p = \left(1 \wedge \frac{\alpha}{\beta}\right)e_1 + \left(1 \wedge \frac{\beta}{\alpha}\right)e_2 \in \partial\|\cdot\|_1(x^{opt}).$$

Therefore, for every $x \in \mathbb{R}^N$ we have

$$\frac{1}{2}\|Lx - y^{\mathrm{L}}\|_2^2 + \frac{1}{2}\lambda\|x\|_1 \geq \langle Lx - y^{\mathrm{L}}, p \rangle - \frac{1}{2}\|p\|_2^2 + \frac{\lambda}{2}\|x\|_1$$

$$= \langle Lx^{opt} - y^{\mathrm{L}}, p \rangle - \frac{1}{2}\|p\|_2^2 + \frac{\lambda}{2}\left(\|x\|_1 - \langle x - x^{opt}, -\frac{2}{\lambda}L^*p \rangle\right)$$

$$\geq \frac{1}{2}\|Lx^{opt} - y^{\mathrm{L}}\|_2^2 + \frac{\lambda}{2}\|x^{opt}\|_1.$$

It follows that x is a minimiser if and only if this string of inequalities holds with equality. This is the case if and only if $Lx - y^{\mathrm{L}} = p = Lx^{opt} - y^{\mathrm{L}}$ and $\|x\|_1 - \langle x - x^{opt}, -\frac{2}{\lambda}L^*p \rangle = \|x^{opt}\|_1$, or equivalently $\alpha x_1 + \beta x_2 = \alpha x_1^{opt} + \beta x_2^{opt} = y_1 - \frac{\lambda}{2(\alpha\vee\beta)}$, $\{x_j\}_{j=3}^N = \{x_j^{opt}\}_{j=3}^N$,

$$|x_1| - (x_1 - x_1^{opt})\left(1 \wedge \frac{\alpha}{\beta}\right) = |x_1^{opt}|, \quad \text{and} \quad |x_2| - (x_2 - x_2^{opt})\left(1 \wedge \frac{\beta}{\alpha}\right) = |x_2^{opt}|.$$

It is now straightforward to verify that the minimisers are precisely those as in the statement of the lemma. $\qquad\square$

Proof of Lemma 11.3. From the definition of L and y^{L}, if $\|Lx - y^{\mathrm{L}}\|_2 \leq \delta$ then $|\alpha x_1 + \beta x_2 - y_1| \leq \delta$. Thus

$$(\alpha \vee \beta)\|x\|_1 \geq (\alpha \vee \beta)(|x_1| + |x_2|) \geq \alpha|x_1| + \beta|x_2| \geq \alpha x_1 + \beta x_2 \geq y_1 - \delta. \tag{A.2}$$

Thus $\min\{\|x\|_1 \mid \|Lx - y^{\mathrm{L}}\|_2 \leq \delta\} \geq (y_1 - \delta)/(\alpha \vee \beta)$. Note that all claimed minimisers x in Lemma 11.5 obey (A.2) as an equality and furthermore such x are feasible for the BPDN problem. Therefore $\min\{\|x\|_1 \mid \|Lx - y^{\mathrm{L}}\|_2 \leq \delta\} = (y_1 - \delta)/(\alpha \vee \beta)$. Thus the solutions to $\Xi_{\mathrm{BPDN}}(y^{\mathrm{L}}, L)$ are precisely the vectors x for which every inequality in (A.2) is obeyed as an equality. More specifically, the following must occur:

- For the first inequality to be an equality we require $x_3 = x_4 = \cdots = x_N = 0$.
- For the second inequality to be an equality we require $x_2 = 0$ in the case $\alpha > \beta$ and $x_1 = 0$ in the case $\alpha < \beta$. In the case $\alpha = \beta$ this is always an equality.
- For the third inequality to be an equality we require x_1 and x_2 to be non-negative.
- For the final inequality to be an equality we need $\alpha x_1 + \beta x_2 = y_1 - \delta$.

It is easy to verify that the x satisfying these properties are exactly the claimed minimisers in the statement of the lemma. $\square$

Proof of Lemma 11.4. Suppose that x is such that $\|x\|_1 \leq \tau$. Then

$$\tau - x_3 \geq \sum_{\substack{1 \leq i \leq N \\ i \neq 3}} |x_i| \geq |x_1| + |x_2| \geq \frac{\alpha|x_1| + \beta|x_2|}{\alpha \vee \beta} \geq \frac{|\alpha x_1 + \beta x_2|}{\alpha \vee \beta} \tag{A.3}$$

and in particular since $|\alpha x_1 + \beta x_2| \geq 0$ we conclude that $(\tau - x_3)^2 \geq (\alpha x_1 + \beta x_2)^2/(\alpha \vee \beta)^2$. Thus for such x we have

$$\|Lx - y^{\mathrm{CL}}\|_2^2 = (\alpha x_1 + \beta x_2 - y_1)^2 + (\tau - x_3)^2 + \sum_{i=4}^{m} x_i^2 \geq [\alpha x_1 + \beta x_2 - y_1]^2 + \frac{(\alpha x_1 + \beta x_2)^2}{(\alpha \vee \beta)^2}$$

$$= \left[\frac{(\alpha \vee \beta)^2 + 1}{(\alpha \vee \beta)^2}\right] \left[\alpha x_1 + \beta x_2 - \frac{y_1(\alpha \vee \beta)^2}{(\alpha \vee \beta)^2 + 1}\right]^2 + \frac{y_1^2}{(\alpha \vee \beta)^2 + 1} \geq \frac{y_1^2}{(\alpha \vee \beta)^2 + 1} \tag{A.4}$$

Therefore $\min\{\|Lx - y^{\mathrm{CL}}\|_2 \mid \|x\|_1 \leq \tau\} \geq y_1[(\alpha \vee \beta)^2 + 1]^{-1/2}$. Note that all claimed minimisers x in Lemma 11.4 obey (A.4) as an equality and furthermore such x have $\|x\|_1 \leq \tau$. Therefore $\min\{\|Lx - y^{\mathrm{CL}}\|_2 \mid \|x\|_1 \leq \tau\} \geq y_1[(\alpha \vee \beta)^2 + 1]^{-1/2}$. Thus the solutions to $\Xi_{\mathrm{CL}}(y^{\mathrm{CL}}, L)$ are precisely the vectors x for which every inequality in (A.4) is obeyed as an equality. More specifically, the following must occur:

1. We require each of the equalities in (A.3) to hold as equality. Thus
   a. The first inequality in (A.3) is an equality if and only if $x_3 = \tau - \sum_{\substack{1 \leq i \leq N \\ i \neq 3}} |x_i|$.

   b. The second inequality in (A.3) is an equality if and only if $x_4 = x_5 = \cdots = x_N = 0$.

   c. The third inequality in (A.3) is an equality if and only if $x_2 = 0$ if $\alpha > \beta$ or $x_1 = 0$ if $\alpha < \beta$. If $\alpha = \beta$ then the third inequality in (A.3) is always an equality.

   d. The final inequality is an equality provided x_1 and x_2 both have the same sign.
2. We require $\alpha x_1 + \beta x_2 = y_1(\alpha \vee \beta)^2[(\alpha \vee \beta)^2 + 1]^{-1}$.

It is now simple to check that the claimed minimisers in (11.6) are the only vectors which satisfy these conditions.

$\square$

Proof of Lemma 11.5. Let $x^{opt} = \eta(y_1, \alpha, \beta)$, if $\beta \geq \alpha$, and $x^{opt} = P_{\mathrm{flip}}\eta(y_1, \alpha, \beta)$, if $\alpha > \beta$. In both cases we find $Tx^{opt} - y^{\mathrm{TV}} = -\frac{\delta(m-1)}{\theta}e_1 + \sum_{j=2}^{m} \frac{\delta(\alpha+\beta)}{\theta}e_j$, and so $\|Tx_{opt} - y^{\mathrm{TV}}\|_2 = \delta$, by definition

of θ. Next, define a dual vector according to $p = -\frac{1}{\alpha\vee\beta}e_1 + \frac{\alpha+\beta}{(\alpha\vee\beta)(m-1)}\sum_{j=2}^{m} e_j \in \mathbb{R}^m$ so that $p = \frac{\theta}{(\alpha\vee\beta)\delta(m-1)}(Tx^{opt} - y^{\mathrm{TV}})$ whenever $\delta > 0$. This in particular implies $\langle Tx^{opt} - y^{\mathrm{TV}}, p\rangle = \delta\|p\|_2$ and $\delta p/\|p\|_2 = Tx^{opt} - y^{\mathrm{TV}}$, for all $\delta \in [0, 1]$.

Next, write $D \in \mathbb{R}^{(N-1)\times N}$ for the matrix corresponding to the pairwise differences operator: $Dx = (x_1 - x_2, x_2 - x_3, \ldots, x_{N-1} - x_N)$, so that $\|x\|_{\mathrm{TV}} = \|Dx\|_1$, for $x \in \mathbb{R}^N$. Defining an auxiliary vector $q = \frac{1}{\alpha\vee\beta}\left(\sum_{j=1}^{N-2}\left(\alpha - \frac{(\alpha+\beta)(j\wedge(m-1)-1)}{(m-1)}\right)e_j - \beta e_{N-1}\right) \in \mathbb{R}^{N-1}$, we have

$$1 \geq \frac{\alpha}{\beta}\wedge 1 = q_1 > q_2 > \cdots > q_{m-1} = \cdots = q_{N-2} = q_{N-1} + \frac{\alpha+\beta}{(\alpha\vee\beta)(m-1)} > q_{N-1} = -\frac{\beta}{\alpha\vee\beta} \geq -1. \tag{A.5}$$

Note that $Dx^{opt} = -re_{N-1}$ in the case $\beta \geq \alpha$, and $Dx^{opt} = re_1$ in the case $\alpha > \beta$, where $r := \frac{y_1 - \delta\theta/(m-1)}{\alpha\vee\beta} > 0$ by the assumption in the statement of the lemma. We therefore deduce $q_j \in \partial|\cdot|_1\left((Dx^{opt})_j\right)$, for all $j = 1, \ldots, N-1$. Furthermore,

$$-T^*p = \frac{1}{\alpha\vee\beta}\left(\alpha e_1 - \frac{\alpha+\beta}{m-1}\left(\sum_{j=2}^{m-1} e_j + e_{N-1}\right) + \beta e_N\right) = D^*q$$

and we thus have, for arbitrary $x \in \mathbb{R}^N$ such that $\|Tx - y^{\mathrm{TV}}\|_2 \leq \delta$, the following inequalities

$$\|x\|_{\mathrm{TV}} \overset{C-S}{\geq} \|x\|_{\mathrm{TV}} + \langle Tx - y^{\mathrm{TV}}, p\rangle - \delta\|p\|_2 \tag{A.6}$$

$$= \|x\|_{\mathrm{TV}} - \langle x - x^{opt}, -T^*p\rangle + \underbrace{\langle Tx^{opt} - y^{\mathrm{TV}}, p\rangle - \delta\|p\|_2}_{=0}$$

$$= \sum_{j=1}^{N-1}\left(|(Dx)_j| - (Dx - Dx^{opt})_j q_j\right) \geq \sum_{j=1}^{N-1}|(Dx^{opt})_j| = \|x^{opt}\|_{\mathrm{TV}}. \tag{A.7}$$

A feasible x is therefore a minimiser if and only if the inequality in (A.6) as well as the inequalities in (A.7) (for each j) all hold with equality, that is to say

- By the Cauchy-Schwarz inequality, (A.6) holds with equality iff $Tx - y^{\mathrm{TV}} = \delta p/\|p\|_2 = Tx^{opt} - y^{\mathrm{TV}}$ i.e. if $Tx = Tx^{opt}$. By the definition of T this occurs when each of $\alpha x^1 + \beta x_N = \alpha x_1 + \beta x_N = \alpha x_1^{opt} + \beta x_N^{opt} = y_1 - \frac{\delta(m-1)}{\theta}$, $\{x_j\}_{j=2}^{m-1} = \{x_j^{opt}\}_{j=2}^{m-1}$ and $x_{N-1} = x_{N-1}^{opt}$.
- The inequalities in (A.7) hold with equality iff for each $j = 1, 2, \ldots, N-1$ we have $|(Dx)_j| - (Dx - Dx^{opt})_j q_j = |D(x^{opt})_j|$.

Let x be a minimiser. Suppose for now that $\beta \geq \alpha$. Since $D\{x_j^{opt}\}_{j=2}^{N-2} = 0$ and $|q_j| < 1$, for all $j \in \{2, \ldots, N-2\}$, the requirements for (A.7) to be an equality imply that $D\{x_j\}_{j=2}^{N-2} = 0$ and hence $x_2 = x_3 = x_4 = \cdots = x_{N-1}$. It thus follows that $\{x_j\}_{j=2}^{N-1} = \{x_j^{opt}\}_{j=2}^{N-1}$. Furthermore, the inequalities in (A.7) hold with equality only if

$$|x_1 - x_2| - (x_1 - x_2)q_1 = 0, \text{ and}$$

$$|x_{N-1} - x_N| - (x_{N-1} - x_N + r)q_{N-1} = r.$$

As $q_1 = \frac{\alpha}{\beta}$ and $q_{N-1} = -1$, it follows that the second of these inequalities is satisfied provided $x_N \geq x_{N-1}$, whereas the first is satisfied provided $x_1 = x_2$ in the case $\alpha > \beta$, or $x_1 \geq x_2$ in the case $\alpha = \beta$. As $x_{N-1} = x_2 = \eta(y_1, \alpha, \beta)_1 < \eta(y_1, \alpha, \beta)_N$, and appreciating that x must additionally satisfy $\alpha x_1 + \beta x_N = y_1 - \frac{\delta(m-1)}{\theta}$, we find that x must have the form claimed in the statement of the lemma.

Similarly, if $\alpha > \beta$, we once again $\{x_j\}_{j=2}^{N-1} = \{x_j^{opt}\}_{j=2}^{N-1}$ but now the inequalities in (A.7) hold with equality only if

$$|x_1 - x_2| - (x_1 - x_2 - r)q_1 = r, \text{ and}$$

$$|x_{N-1} - x_N| - (x_{N-1} - x_N)q_{N-1} = 0.$$

Then, as $q_1 = 1$ and $q_{N-1} = -\beta/\alpha$, these are satisfied only if $x_1 > x_2$ and $x_N = x_{N-1}$. Again appreciating $x_{N-1} = x_2 = \eta(y_1, \alpha, \beta)_1$ and $\alpha x_1 + \beta x_N = y_1 - \frac{\delta(m-1)}{\theta}$ establishes $x = x^{opt}$. Conversely, every x of the claimed form satisfies (A.6) and (A.7) as equalities, and is therefore a minimiser.

$\square$

Corollary A.2. *Write* $f(\delta) = \min\{\|x\|_{\mathrm{TV}} \mid \|Tx - y^{\mathrm{TV}}\|_2 \leq \delta\}$. *Then*

$$f(\delta) = \begin{cases} \frac{y_1 - \delta\theta/(m-1)}{\alpha \vee \beta} & \text{if } y_1 > \frac{\delta\theta}{m-1} \\ 0 & \text{if } y_1 \leq \frac{\delta\theta}{m-1} \end{cases}.$$

Proof of Corollary A.2. Assume first that $y_1 > \delta\theta/(m-1)$. Then, using Lemma 11.5, we see that $f(\delta) = \eta(y_1, \alpha, \beta)_N - \eta(y_1, \alpha, \beta)_1 = \frac{y_1 - \delta\theta/(m-1)}{\alpha \vee \beta}$. To prove the result in the case $y_1 \leq \delta\theta/(m-1)$, we note that by the definition of f, f is a non-negative decreasing function, and hence we have $0 \leq f(\delta) \leq f(y_1(m-1)/\theta) \leq \inf\{f(\delta') \mid \delta' < y_1(m-1)/\theta\} = 0$ and thus $f(\delta) = 0$. $\square$

Proof of Lemma 11.6. We can write the unconstrained lasso problem as

$$\underset{x \in \mathbb{R}^N}{\arg\min} \min_{\delta \geq 0}\{\delta^2 + \lambda\|x\|_{\mathrm{TV}} \mid \|Tx - y^{\mathrm{TV}}\|_2 \leq \delta\}. \tag{A.8}$$

Let us analyse $\arg\min_{\delta \geq 0} \min_{x \in \mathbb{R}^N}\{\delta^2 + \lambda\|x\|_{\mathrm{TV}} \mid \|Tx - y^{\mathrm{TV}}\|_2 \leq \delta\}$. Note that this is equivalent to $\arg\min_{\delta \geq 0} \left(\delta^2 + \min_{x \in \mathbb{R}^N}\{\lambda\|x\|_{\mathrm{TV}} \mid \|Tx - y^{\mathrm{TV}}\|_2 \leq \delta\}\right) = \arg\min_{\delta \geq 0}\{\delta^2 + \lambda f(\delta)\}$ where $f(\delta)$ is defined in Corollary A.2. Using Corollary A.2, for $y_1 \leq \delta\theta/(m-1)$ the function $\delta \to \delta^2 + \lambda f(\delta)$ is strictly increasing since $f(\delta) = 0$. Thus we can write $\arg\min_{\delta \geq 0} \min_{x \in \mathbb{R}^N}\{\delta^2 + \lambda\|x\|_{\mathrm{TV}} \mid \|Tx - y^{\mathrm{TV}}\|_2 \leq \delta\}$ as $\underset{\delta \geq 0, y_1 \geq \delta\theta/(m-1)}{\arg\min} \delta^2 + \lambda f(\delta)$. Noting that $f(\delta) = (\alpha \vee \beta)^{-1}(y_1 - \delta\theta/(m-1))$, we get

$$\underset{\delta \geq 0, y_1 \geq \delta\theta/(m-1)}{\arg\min} \delta^2 + \lambda f(\delta) = \underset{\delta \geq 0, y_1 \geq \delta\theta/(m-1)}{\arg\min} \delta^2 - \lambda\delta\theta(m-1)^{-1}(\alpha \vee \beta)^{-1} + \lambda y_1(\alpha \vee \beta)^{-1}$$

which is minimised when $\delta = \hat{\delta}$, where $\hat{\delta} := \lambda\theta(m-1)^{-1}(\alpha \vee \beta)^{-1}/2$ (we have used here that $\hat{\delta}$ is feasible for the problem since $y_1 > \lambda\theta^2[2(m-1)^2(\alpha \vee \beta)]^{-1}$ and so $y_1 > \hat{\delta}\theta/(m-1)$).

We can use this result in (A.8) to get

$$\Xi_{\mathrm{ULTV}}(T, y^{\mathrm{TV}}) = \underset{x \in \mathbb{R}^N}{\arg\min} \left\{\hat{\delta}^2 + \lambda\|x\|_{\mathrm{TV}} \,\middle|\, \|Tx - y^{\mathrm{TV}}\|_2 \leq \hat{\delta}\right\}$$

$$= \underset{x \in \mathbb{R}^N}{\arg\min} \left\{\|x\|_{\mathrm{TV}} \,\middle|\, \|Tx - y^{\mathrm{TV}}\|_2 \leq \frac{\lambda\theta}{2(m-1)(\alpha \vee \beta)}\right\}$$

Noting that $\hat{\delta} \leq \frac{\lambda m}{2(m-1)^{\frac{1}{4}}} \leq 3\lambda \leq 1$, and so Lemma 11.5 can be applied in each of the cases $\alpha < \beta$, $\alpha = \beta$ and $\alpha > \beta$ to reach the desired conclusion.

$\square$

Proof of Lemma 11.8. Suppose first that $y_1 > 0$, $\alpha > 0$, and $\beta \geq 0$. Simple gaussian elimination shows that the set of x for which $x \geq 0$ and $A^{\mathrm{LP,D}}x = y^{\mathrm{L}}$ is exactly

$$\{x \in \mathbb{R}^N \mid \alpha x_1 + \beta x_2 = y_1, x_3 = x_4 = \cdots = x_m = x_{m+1} = 0, x_{m+2}, x_{m+3}, \ldots, x_N \geq 0\}$$

so that $\langle x, c \rangle = \sum_{i=1}^N x_i \geq x_1 \geq (\alpha x_1 + \beta x_2)/\alpha \geq y_1/\alpha$. Therefore as $\langle x, c \rangle_k$ is increasing as a function of $\langle x, c \rangle$ we have that any $x \geq 0$ such that $A^{\mathrm{LP,D}}x = y^{\mathrm{L}}$ must satisfy $\langle x, c \rangle_k \geq \lfloor 10^k y_1/\alpha \rfloor 10^{-k}$ and this equality is attained when $x = y_1 e_1/\alpha$. The proof of (11.15) is complete by noting that $10^{-k}\lfloor 10^k y_1/\alpha \rfloor \leq 10^{-k}\lfloor 10^k M \rfloor \leq M$ if $y_1/\alpha < 10^{-k}(\lfloor 10^k M \rfloor + 1)$ and $10^{-k}\lfloor 10^k y_1/\alpha \rfloor \geq 10^{-k}(\lfloor 10^k M \rfloor + 1) > M$ if $y_1/\alpha \geq 10^{-k}(\lfloor 10^k M \rfloor + 1)$.

Now suppose that $y_1 = 0$. Then $x = 0$ satisfies $A^{\mathrm{LP,D}}x = y^{\mathrm{L}}$ and $x \geq 0$. Moreover, $\langle 0, c \rangle_k = 0 \leq M$. This concludes the proof of the lemma. $\square$

APPENDIX B. STANDARD SPARSE RECOVERY RESULTS

The concept of sparsity has been dominating modern signal and image processing over the last few decades. Sparsity is a crucial element to this paper and we use this appendix to cover some standard concepts and tools that we will make use of.

B.1. **The basics of sparsity.** Recall that we say that a vector $x \in \mathbb{C}^N$ is s-sparse if x has at most s nonzero entries. One of the key mainstays in the theory of sparse recovery is the ℓ^2-robust nullspace property (RNP).

Definition B.1 (Robust Nullspace Property). A matrix $U \in \mathbb{R}^{m \times N}$ satisfies the ℓ^2-robust nullspace property of order s with parameters $\rho \in (0, 1)$ and $\tau > 0$ if

$$\|v_S\|_2 \leq \frac{\rho}{\sqrt{s}} \|v_{S^c}\|_1 + \tau \|Uv\|_2 \tag{B.1}$$

for all sets $S \subset \{1, \ldots, N\}$ of cardinality s and all vectors $v \in \mathbb{R}^n$.

Directly showing that a matrix has the RNP is often difficult. Thus, one usually attempts to instead establish the restricted isometry property (RIP) that implies the RNP, recalled next.

Definition B.2 (RIP). A matrix $U \in \mathbb{C}^{m \times N}$ is said to satisfy the *restricted isometry property (RIP) or order s* if there is a $\delta \in [0, 1)$ such that

$$(1 - \delta)\|x\|_2^2 \leq \|Ux\|_2^2 \leq (1 + \delta)\|x\|_2^2, \tag{B.2}$$

for all s-sparse $x \in \mathbb{R}^N$. The *restricted isometry constant of order s*, denoted by δ_s, is defined to be the smallest such δ so that (B.2) is satisfied. Equivalently, δ_s is given by

$$\delta_s = \sup_{\substack{T \subset \{1,2,\ldots,N\} \\ |T| \leq s}} \|P_T U^* U P_T - I_T\|_2$$

where P_T denotes the projection onto the coordinates indexed by T.

We thus have the following result linking the RIP and RNP.

Theorem B.3 ([61, Theorem 6.13]). *If the restricted isometry constant δ_{2s} of order $2s$ of a matrix $A \in \mathbb{R}^{m \times N}$ obeys $\delta_{2s} < 4/\sqrt{41}$, then A satisfies the ℓ^2-robust nullspace property of order s with parameters*

$$\rho = \frac{\delta_{2s}}{\sqrt{1 - \delta_{2s}^2} - \delta_{2s}/4} \quad and \quad \tau = \frac{\sqrt{1 + \delta_{2s}}}{\sqrt{1 - \delta_{2s}^2} - \delta_{2s}/4}.$$

It is generally difficult to construct deterministic matrices with the RIP. Instead, one often resorts to random matrices, such as provided by the following result taken from [62] and adapted to fit the notation of this paper.

Theorem B.4 ([62, Theorem 3.9]). *For every $K > 0$, there exist constants C_1, C_2 and C_3 depending only on K such that the following occurs. Let $x^1, x^2, \ldots, x^N$ be N-dimensional complex vectors with $N \geq 3$ and $\|x^i\| \leq K$, for all i. Assume that $\sum_{i=1}^{N} x^i \otimes x^i = N I_N$ and that m is a natural number with*

$$N \geq m \geq C_1 s \log(N) \epsilon^{-2} \log[s \log(N) \epsilon^{-2}] \log^2 s$$

where $\epsilon \in (0, 1)$ and s is an integer with $s \geq 3$. Let R be a random subset of $\{1, 2, \ldots, N\}$ chosen according to a Bernoulli model with probability m/N, that is, each entry of R is chosen independently and $i \in R$ with probability m/N. Set $X := \sup_{|T| \leq s} \|I_T - m^{-1} \sum_{i \in R} x_T^i \otimes x_T^i\|_2$, where x_T^i is the vector x^i restricted to a set T. Then

$$\mathbb{P}(X > C_2 \alpha \epsilon) \leq 3 \exp\left(-C_3 \alpha \epsilon m s^{-1}\right) + 2 \exp(-\alpha^2), \quad for\ all\ \alpha > 1. \tag{B.3}$$

We can use Theorem B.4 to show the existence of matrices that satisfy the RIP, as follows:

Theorem B.5. *For every $K > 0$, there exists a constant C_4 depending only on K so that for all natural numbers m, N and s, each at least 3 and satisfying*

$$N \geq m \geq C_4 s \log(N) \log[s \log(N)] \log^2 s, \tag{B.4}$$

as well as any unitary matrix U with $\|U\|_{\max} \leq K/\sqrt{N}$, there exists a set S of size exactly m such that $\sqrt{N/m}\, P_S U$ has the RIP of order s with $\delta_s \leq 1/5$.

Proof. Fix $K > 0$ and let C_1, C_2 and C_3 be the constants from Theorem B.4. We set

$$\epsilon = \frac{1}{10 C_2} \wedge \frac{1}{2}, \quad \alpha = \frac{1}{5 C_2 \epsilon}, \quad C_4 = \left[C_1 \epsilon^{-2} \left(1 + 2 \log \epsilon^{-1} \right) \right] \vee \frac{5 C_2 s}{\epsilon^2 C_3}$$

These choices of parameters ensure that $0 < \epsilon \leq 1/2 < 1$, that $\alpha \geq \frac{10 C_2}{5 C_2} = 2$ and, using B.4, that

$$m \geq C_1 s \log(N) \epsilon^{-2} \log[s \log(N)] \left[1 + 2 \log \epsilon^{-1} \right] \log^2 s \geq C_1 s \log(N) \epsilon^{-2} \log[s \log(N) \epsilon^{-2}] \log^2 s. \tag{B.5}$$

since both s and N are at least 3. Furthermore, our choice of C_4, α and ϵ depend only on K.

Next, we choose the vectors $x^1, x^2, \ldots, x^N$ so that x^i is the ith column of U multiplied by $\sqrt{N}$. Such a choice of x^i gives $\|x^i\|_\infty \leq K$. Moreover, $\sum_{i=1}^N x^i \otimes x^i = N U^* U = I_N$ by the assumption that U is unitary. Therefore, Theorem B.4 applies and we conclude that for a random set R chosen according to the Bernoulli model equation (B.3) holds.

The choice of the parameters made in (B.5) ensures that $C_2 \alpha \epsilon \leq 1/5$. Because $\log(s), \log(N)$ and $\log(s \log(N))$ are each at least 1, condition (B.4) implies that $m \geq C_4 \geq \frac{5 C_2 s}{\epsilon^2 C_3}$. Therefore, again using (B.5), we see that $C_3 \alpha \epsilon m/s \geq \epsilon^{-2}$. Finally, our choice of x^i ensure that X can be written as $X = \|I_T - m^{-1} N P_T U^* P_S U P_T\|_2$. By the definition of the restricted isometry constant this is in fact δ_s for the matrix $\sqrt{N/m}\, P_S U$. Thus (B.3) implies that the random variable δ_s corresponding to the random matrix $\sqrt{N/m}\, P_S U$ satisfies $\mathbb{P}(\delta_s > 1/5) \leq 3 \exp(-\epsilon^{-2}) + 2 \exp(-\alpha^2) \leq 5 \exp(-4)$.

At this point, the proof would be done if the cardinality of S were fixed since $5 \exp(-4) < 1$ and hence there must exist at least one such R with $\delta_s \leq 1/5$. Unfortunately, the cardinality of R is a random variable since S is selected according to the Bernoulli model. We will therefore consider random sets S' chosen as follows: let X be the set of all subsets of $\{1, 2, \ldots, N\}$ with cardinality m. We take S to be an element of X chosen uniformly at random. Using an argument similar to the one presented in [61, p. 468] we will bound the probability that the random variable δ_s corresponding to the random matrix $\sqrt{N/m}\, P_{S'} U$ exceeds $1/5$.

To this end, let E_U^u be the event that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_{S'} U x\|_2^2 > 6/5$ and let E_U^l be the event that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_{S'} U x\|_2^2 < 4/5$. Similarly, for the random sets S chosen according to the Bernoulli model, let E_B^u be the event that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_S U x\|_2^2 > 6/5$ and let E_B^l be the event that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_S U x\|_2^2 < 4/5$. We have shown already that $\mathbb{P}(E_B^l \cup E_B^u) \leq 5 \exp(-4)$.

For a given i, let B_i denote the collection of sets S with $S \subseteq \{1, 2, \ldots, N\}$, $|S| = i$ and such that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_S U x\|_2^2 > 6/5$. Because the Bernoulli model selects elements independently and with equal probability, $\mathbb{P}\left(E_B^u \,\middle|\, |S| = i \right) = |B_i|/\binom{n}{i}$. By an argument originally used by Sperner to prove Sperner's theorem [63, p.3], the shade ∇B_i of B_i defined by

$$\nabla B_i = \{\hat{S} \subseteq \{1, 2, \ldots, N\}|, | |\hat{S}| = i + 1 \text{ and } \exists S \in B_i \text{ with } S \subseteq \hat{S}\}$$

satisfies $|\nabla B_i| \geq (n - i)|B_i|/(i + 1)$. Moreover, if S is in B_i and $\hat{S}$ is such that $|\hat{S}| = i + 1$ and $S \subset \hat{S}$ then $\|\sqrt{\frac{N}{m}} P_{\hat{S}} A x\|_2^2 \geq \|\sqrt{\frac{N}{m}} P_S A x\|^2$ and hence $\hat{S} \in B_{i+1}$. Thus $\nabla B_i \subset B_{i+1}$. Therefore for

$i \leq N - 1$ we have

$$\mathbb{P}\left(E_B^u \,\middle|\, |S| = i\right) = \frac{|B_i|}{\binom{n}{i}} \leq \frac{(i+1)|\nabla B_i|}{\binom{n}{i}(n-i)} = \frac{|\nabla B_i|}{\binom{n}{i+1}} \leq \frac{|B_{i+1}|}{\binom{n}{i+1}} = \mathbb{P}\left(E_B^u \,\middle|\, |S| = (i+1)\right).$$

Let A_i denote the collection of sets S with $S \subseteq \{1, 2, \ldots, N\}$, $|S| = i$ and such that there exists an s-sparse unit vector x such that $\|\sqrt{N/m}\, P_S U x\|_2^2 < 4/5$. A similar argument to the preceding one for B_i this time using the fact (also proven by Sperner) that the shadow ΔA_i of A_i defined by

$$\Delta A_i = \{\hat{S} \subseteq \{1, 2, \ldots, N\}|, \, |\,|\hat{S}| = i - 1 \text{ and } \exists S \in A_i \text{ with } \hat{S} \subseteq S\}$$

satisfies $|\Delta A_i| \geq (i|A_i|)/(n - i + 1)$ gives $\mathbb{P}\left(E_B^l \,\middle|\, |S| = i\right) \leq \mathbb{P}\left(E_B^l \,\middle|\, |S| = (i-1)\right)$. Thus

$$\mathbb{P}(E_B^u) = \sum_{i=0}^{N} \mathbb{P}\left(E_B^u \,\middle|\, |S| = i\right) \mathbb{P}(|S| = i) \geq \sum_{i=m}^{N} \mathbb{P}\left(E_B^u \,\middle|\, |S| = i\right) \mathbb{P}(|S| = i)$$

$$\geq \mathbb{P}\left(E_B^u \,\middle|\, |S| = m\right) \sum_{i=m}^{N} \mathbb{P}(|S| = i) \geq \frac{\mathbb{P}(E_U^u)}{2}. \quad \text{(B.6)}$$

where the final inequality follows because m is the median of the random variable $|S|$. Similarly we obtain

$$\mathbb{P}(E_B^l) \geq \sum_{i=0}^{m} \mathbb{P}\left(E_B^u \,\middle|\, |S| = i\right) \mathbb{P}(|S| = i) \geq \frac{\mathbb{P}(E_U^l)}{2}. \quad \text{(B.7)}$$

Combining (B.6), (B.7) and the already established result $\mathbb{P}(E_B^l \cup E_B^u) \leq 5\exp(-4)$ gives

$$\mathbb{P}(E_U^l \cup E_U^u) \leq \mathbb{P}(E_U^l) + \mathbb{P}(E_U^u) \leq 2\left[\mathbb{P}(E_B^l) + \mathbb{P}(E_B^u)\right] \leq 4\mathbb{P}(E_B^l \cup E_B^u) \leq 20\exp(-4) < 1$$

Hence $\mathbb{P}(\delta_s > 1/5) < 1$ where δ_s is the random RIP constant associated to the random matrix $P_{S'} U$ such that S' a uniformly randomly chosen subset of $\{1, 2, \ldots, N\}$ with $|S'| = m$.

Thus there exists a set S with $|S| = m$ such that the matrix $P_S U$ has $\delta_s \leq 1/5$, as otherwise we would have $\mathbb{P}(\delta_s \leq 1/5) = 0$ and this would be a contradiction. $\qquad\square$

We can apply this result to the discrete cosine matrix to derive the following:

Theorem B.6. *There exists a constant C_5 with the following property: for any natural numbers m,N, and s with $N \geq m \geq 3$ and $s \geq 2$ obeying*

$$N \geq m \geq C_5 s \log(N) \log[s \log(N)] \log^2 s, \quad \text{(B.8)}$$

there exists a matrix $F \in \mathbb{R}^{m \times N}$ with the RNP of order s with parameters $\rho < 1/3$ and $\tau < 2$ that also satisfies $\|F\|_2 \leq \sqrt{N/m}$.

Proof. Let $U \in \mathbb{R}^{N \times N}$ be the matrix corresponding to the second variant of the discrete cosine transform, i.e., its entries are given by

$$U_{jk} = \begin{cases} \sqrt{\frac{2}{N}} \cos\left(\frac{\pi}{2N}(j-1)(2k-1)\right) & \text{for } j, k \in \{1, \ldots, N\}, j \neq 1 \\ \sqrt{\frac{1}{N}} & \text{for } j = 1 \end{cases}$$

Note that U is an orthonormal matrix with $\|U\|_{\max} \leq \sqrt{2/N}$. Take $C_5 = 16C_4$ where C_4 is taken from the statement of Theorem B.5 with $K = \sqrt{2}$. Then condition B.8 implies that

$$N \geq m \geq 2C_4 s \log(N)(2\log(s\log(N)))(2\log(s))^2 \geq 2C_4 s \log(N)(\log(2s\log(N)))(\log(2s))^2$$

since $s \geq 2$. Thus, we can apply Theorem B.5 to obtain a set S so that $F := \sqrt{N/m}\, P_S U$ obeys the RIP of order $2s$ with $\delta_{2s} \leq 1/5$. Hence, by Theorem B.3, the matrix F satisfies the RNP with parameters

$$\rho := \frac{1/5}{\sqrt{1 - (1/5)^2} - (1/5)/4} < \frac{1}{3} \quad \text{and} \quad \tau := \frac{\sqrt{1 + 1/5}}{\sqrt{1 - (1/5)^2} - (1/5)/4} < 2\,.$$

The proof is complete by noting that $\|F\|_2 \leq \sqrt{N/m}\,\|U\|_2 \leq \sqrt{N/m}$.

$\square$

This paper also makes use of the following results:

Theorem B.7 ([61, Theorem 4.25]). *Suppose that $A \in \mathbb{C}^{m \times N}$ satisfies the RNP of order s with parameters $\rho \in (0,1)$ and $\tau > 0$. Then, for all $x, z \in \mathbb{C}^N$, we have*

$$\|x - z\|_2 \leq \frac{(1+\rho)^2}{1-\rho}\frac{1}{\sqrt{s}}\left(\|z\|_1 - \|x\|_1 + 2\sigma_s(x)_1\right) + \frac{(3+\rho)\tau}{1-\rho}\|Ax - Az\|_2, \qquad \text{(B.9)}$$

where $\sigma_s(x)_1 = \min\{\|x - y\|_1 \mid s\text{-sparse } y \in \mathbb{R}^N\}$.

Lemma B.8. *Let $b_y, \varepsilon_y > 0$ and suppose $A \in \mathbb{R}^{m \times N}$ satisfies the RNP of order s with parameters $\rho \in (0,1)$ and $\tau > 0$. Suppose we are given $y \in \mathbb{R}^m$ such that $\|y\|_2 \leq b_y\sqrt{N/m}$ and $\|Ax - y\|_2 \leq \varepsilon$, for some s-sparse $x \in \mathbb{R}^N$. Let A' and y' be such that*

$$\|y - y'\|_2 \leq \varepsilon_y, \qquad \|A - A'\|_2 \leq \frac{\varepsilon_y}{\tau\left(\varepsilon + b_y\sqrt{N/m}\right)}.$$

Then

$$\|x\|_2 \leq \tau\left(\varepsilon + b_y\sqrt{N/m}\right) \quad \text{and} \quad \|A'x - y'\|_2 \leq \varepsilon + 2\varepsilon_y.$$

Proof of Lemma B.8. Let $S = \text{supp}(x)$. Now, by applying (B.1) to x we have

$$\|x\|_2 \leq \tau\|Ax\|_2 \leq \tau\left(\|Ax - y\|_2 + \|y\|_2\right) \leq \tau\left(\varepsilon + b_y\sqrt{N/m}\right),$$

and thus

$$\begin{aligned}
\|A'x - y'\|_2 &\leq \|A - A'\|_2\|x\|_2 + \|Ax - y\|_2 + \|y - y'\|_2 \\
&\leq \|A - A'\|_2 \cdot \tau\left(\varepsilon + b_y\sqrt{N/m}\right) + \varepsilon + \varepsilon_y \leq \varepsilon + 2\varepsilon_y.
\end{aligned}$$

$\square$

Lemma B.9 ([64, Lemma 8.5]). *Suppose $A \in \mathbb{R}^{m \times N}$ has the RNP of order s with constants $\rho \in (0,1)$ and $\tau > 0$. If $A' \in \mathbb{R}^{m \times N}$ satisfies $\|A' - A\|_2 \leq \epsilon$ where ϵ is a non-negative real number with $\epsilon \leq \frac{1-\rho}{\tau(\sqrt{s}+1)}$ then A' satisfies the RNP of order s with constants ρ' and τ' satisfying*

$$\rho' = \frac{\rho + \tau\epsilon\sqrt{s}}{1 - \tau\epsilon}, \quad \tau' = \frac{\tau}{1 - \tau\epsilon}$$

B.2. **Standard matrices used in the theory of sparsity.** We recall here two types of matrices that are frequently used in compressive sensing and sparse regularisation applications:

Hadamard matrices. In this paper we will only consider Hadamard matrices of size $2^n \times 2^n$, for $n \in \mathbb{N}$. We define a 'naturally ordered' Hadamard matrix H_n with entries ± 1 of dimension $2^n \times 2^n$ by the recurrence relation

$$H_j = H_1 \otimes H_{j-1}, \quad H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad H_0 = 1, \quad j \in \mathbb{N}.$$

where $\otimes$ denotes the Kronecker product. H_n has orthogonal rows, columns and the property that $H_n^* H_n = 2^n I_{2^n}$ where $I_{2^n} \in \mathbb{R}^{2^n \times 2^n}$ is the identity matrix. We also consider Hadamard matrices in the 'sequency ordering': that is, the rows of H_n are ordered so that the number of sign changes in a given row is increasing. We shall make the distinction clear where important.

Hadamard-to-Haar matrices. Let $H_n \in \mathbb{R}^{2^n \times 2^n}$ be the Hadamard matrix in the sequency ordering and let $W_n \in \mathbb{R}^{2^n \times 2^n}$ be the 1D discrete Haar wavelet transform matrix. Then, by [65, Lemma 1 & Lemma 2] we have $H_n W_n^{-1} = \begin{pmatrix} 1 \end{pmatrix} \oplus \bigoplus_{i=0}^{n-1} X^i$ where the matrix $X^i \in \mathbb{R}^{2^i \times 2^i}$ is unitary and satisfies $|(X^i)_{jk}| = 1/\sqrt{2^i}$, for all $i = 0, \ldots, n-1$ and $j, k = 1, \ldots, 2^i$. Now, a row-subsampled Hadamard-to-Haar matrix is any matrix of the form

$$A = \begin{pmatrix} 1 \end{pmatrix} \oplus \bigoplus_{i=0}^{n-1} c_i P_{S_i} X^i, \tag{B.10}$$

where the S_i are subsets of $\{1, \ldots, 2^{i-1}\}, P_{S_i} : \mathbb{R}^{2^{i-1}} \to \mathbb{R}^{|S_i|}$ are the corresponding projection operators selecting the coordinates in S_i and $c_i = \sqrt{\frac{2^i}{|S_i|}}$ for nonempty S_i and $c_i = 0$ if $S_i = \varnothing$. Matrices formed by taking the product of a Hadamard transform with an inverse wavelet transform, like the Hadamard-To-Haar matrices defined above, have proven to be very effective in compressive sensing, particularly on imaging applications [66].

APPENDIX C. SEPARATION ORACLES AND THE ELLIPSOID ALGORITHM

In this section we define the concepts needed in the statements of Theorem 20.2 and Theorem 20.3, namely those of the weak optimisation problem, weak separation oracle, encoding functions and polynomially separable classes. We start with the weak optimisation problem, which we define as a minimisation problem (rather than a maximisation problem as in [37]) for convenience.

Definition C.1 ([37, Def. 2.1.10], Weak optimisation problem). Let $\mathcal{K} \subset \mathbb{R}^n$ be a compact convex set, and suppose that $R > 0$ is a rational such that $\mathcal{K} \subset \overline{\mathcal{B}}_R(0)$. Furthermore, let $c \in \mathbb{Q}^n$ and $\zeta \in \mathbb{Q}$. The weak optimisation problem $(\mathcal{K}, R, c, \zeta)$ is the task to either

(a) find a $z^* \in \mathbb{Q}^n$ such that $z^* \in S(\mathcal{K}, \zeta)$ and $\langle c, z^* \rangle \leq \langle c, z \rangle + \zeta$, for all $z \in S(\mathcal{K}, -\zeta)$, or
(b) assert that $S(\mathcal{K}, -\zeta) = \varnothing$.

The following is the definition of a weak separation oracle, which we present as a synthesis of [37, Assump. 1.2.1], [37, Def. 2.1.13], and the discussion on pages 54 and 55 of [37] on the description of compact convex sets by means of a separation oracle.

Definition C.2 (Weak separation oracle). Let $\mathcal{K} \subset \mathbb{R}^n$ be a compact convex set. We say that a procedure $\mathrm{SEP}_{\mathcal{K}}$ is a weak separation oracle for $\mathcal{K}$ if, given a vector $w \in \mathbb{Q}^n$ and a rational $\xi > 0$,

(a) $\mathrm{SEP}_{\mathcal{K}}$ either outputs a $d \in \mathbb{Q}^n$ with $\|d\|_\infty = 1$ and such that $\langle d, z \rangle \leq \langle d, w \rangle + \xi$, for all $z \in S(\mathcal{K}, -\xi)$, or
(b) $\mathrm{SEP}_{\mathcal{K}}$ asserts that $w \in S(\mathcal{K}, \xi)$.

Moreover, in the Turing case, we insist that there exist a polynomial $P_{\mathcal{K}} : \mathbb{R} \to \mathbb{R}$ such that, whenever $\mathrm{SEP}_{\mathcal{K}}$ outputs a vector d as in item (i), we have $\mathrm{Len}(d) \leq P_{\mathcal{K}}(\mathrm{Len}(w) + \mathrm{Len}(\xi))$.

Note that by the separating hyperplane theorem [29] applied to the convex sets K and $\{w\}$, at least one of (a) and (b) in Definition C.2 are satisfied whenever K is non-empty. Moreover, (a) is trivially satisfied for any $d \in \mathbb{Q}^n$ with $\|d\|_\infty = 1$ if K is empty.

Remark C.3. When considering the BSS instead of the Turing model, all quantities in Definitions C.1 and C.2 specified to be rationals are allowed to be (not necessarily rational) real numbers.

Next, we introduce Turing encoding functions which represents the encoding of the various sets $\mathcal{K}$ in a form that can be presented to a Turing machine as input.

Definition C.4 (Turing encoding function). Let $\mathcal{A}^*$ denote the set of finite-length strings in a finite alphabet $\mathcal{A}$. A Turing encoding function for $\mathscr{K}$ is an injective function $\mathrm{Data}^{\mathrm{Tur}} : \mathscr{K} \to \mathcal{A}^*$.

Note that the explicit form of $\mathrm{Data}^{\mathrm{Tur}}$ depends on the particular class $\mathscr{K}$ under consideration. The condition on $\mathrm{Data}^{\mathrm{Tur}}$ in Definition C.4 simply states that the sets are encoded uniquely. For example, for basis pursuit denoising where our compact convex sets will be of the form $\{z \in \mathbb{R}^N \mid \|A'z - y'\|_2 \leq \delta', \|z\|_2 \leq R'\}$, where $\delta', R' \in \mathbb{D}$, $y' \in \mathbb{D}^m$, and $A' \in \mathbb{D}^{m \times N}$, one possible encoding in the alphabet $\mathcal{A} = \{0, \ 1, -, \ ., \ ;\}$ is

$$m \ ; \ N \ ; \ \delta' \ ; \ R' \ ; \ y'_1 \ ; \ y'_2 \ \cdots \ ; \ y'_m \ ; \ A'_{1,1} \ ; \ A'_{1,2} \ \cdots \ ; \ A'_{m,N} \quad \in \mathcal{A}^*$$

where all the dyadic rationals are written out in their binary representation.

We are now ready to define polynomially separable classes in the Turing model.

Definition C.5 (Polynomially separable class – Turing case)**.** Suppose $\mathscr{K}$ is a circumscribed class equipped with a Turing encoding function $\mathrm{Data}^{\mathrm{Tur}} : \mathscr{K} \to \mathcal{A}^*$. We say that $\mathscr{K}$ is *Turing-polynomially separable with respect to* $\mathrm{Data}^{\mathrm{Tur}}$ if there exist a Turing machine that takes in $\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R) \in \mathcal{A}^*$, a $w \in \mathbb{Q}^n$, and a rational $\xi > 0$ as its input and acts as a weak separation oracle for $\mathcal{K}$, i.e., it either

  (a) outputs a $d \in \mathbb{Q}^n$ with $\|d\|_\infty = 1$ and such that $\langle d, z \rangle \leq \langle d, w \rangle + \xi$, for all $z \in \mathcal{K}$, or
  (b) asserts that $w \in S(\mathcal{K}, \xi)$,

such that the runtime of the Turing machine is bounded by a polynomial of $\mathrm{Len}\big(\mathrm{Data}^{\mathrm{Tur}}(\mathcal{K}, n, R)\big)$, $\mathrm{Len}(R)$, $\mathrm{Len}(w)$, $\mathrm{Len}(\xi)$, and n.

Finally, we present analogues of the concepts above for the BSS model of computation, following the ideas in [57, Sec. 1.1, Sec. 1.2].

Definition C.6 (BSS encoding function)**.** We define $\mathcal{V} = \bigcup_{k=1}^\infty \mathbb{R}^k$, i.e., the set of real vectors of arbitrary length. A BSS encoding function for $\mathscr{K}$ is a function $\mathrm{Data}^{\mathrm{BSS}} : \mathscr{K} \to \mathcal{V}$ so that, for all $(\mathcal{K}_1, n, R), (\mathcal{K}_2, n, R) \in \mathscr{K}$, $\mathcal{K}_1 \neq \mathcal{K}_2$ implies $\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}_1, n, R) \neq \mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}_2, n, R)$.

In analogy to the Turing encoding function, $\mathrm{Data}^{\mathrm{BSS}}$ serves to encode $\mathcal{K}$ as a vector of reals, which can be accepted as input by a BSS machine. For basis pursuit denoising, the convex set $\{z \in \mathbb{R}^N \mid \|A'z - y'\|_2 \leq \delta', \|z\|_2 \leq R\}$, where $y' \in \mathbb{R}^m$ and $A' \in \mathbb{R}^{m \times N}$, can be encoded as

$$\big(m, N, \delta', R', y'_1, \cdots, y'_m, A'_{1,1}, A'_{1,2}, \cdots, A'_{m,N}\big) \in \mathbb{R}^{4+m+mN} \subset \mathcal{V}.$$

Definition C.7 (Polynomially separable class – BSS case)**.** Suppose $\mathscr{K}$ is a circumscribed class equipped with a BSS encoding function $\mathrm{Data}^{\mathrm{BSS}} : \mathscr{K} \to \mathcal{V}$. We say that $\mathscr{K}$ is *BSS-polynomially separable with respect to* $\mathrm{Data}^{\mathrm{BSS}}$ if there exists a BSS machine that takes in $\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}, n, R) \in \mathcal{V}$, a $w \in \mathbb{R}^n$, and a real $\xi > 0$ as its input and acts as a weak separation oracle for $\mathcal{K}$, i.e., it either

  (a) outputs a $d \in \mathbb{R}^n$ with $\|d\|_\infty = 1$ and such that $\langle d, z \rangle \leq \langle d, w \rangle + \xi$, for all $z \in \mathcal{K}$, or
  (b) asserts that $w \in S(\mathcal{K}, \xi)$,

such that the runtime of the BSS machine is bounded by a polynomial of $\dim(\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}, n, R))$, i.e., the dimension of the real vector $\mathrm{Data}^{\mathrm{BSS}}(\mathcal{K}, n, R)$.

Note that, unlike in the Turing case, we now have no concept of the length of the encoding of a convex set, but instead the separation oracle must be executable in runtime which is polynomial only in the dimension of the data vector encoding the convex set.

REFERENCES

[1] S. Smale, "Mathematical problems for the next century", *Mathematical Intelligencer*, vol. 20, pp. 7–15, 1998.

[2] T. C. Hales, "A proof of the Kepler conjecture", *Ann. of Math. (2)*, vol. 162, no. 3, pp. 1065–1185, 2005.

[3] T. C. Hales and et al., "A formal proof of the kepler conjecture", *Forum of Mathematics, Pi*, vol. 5, e2, 2017.

[4] C. Fefferman and L. Seco, "On the energy of a large atom", *Bull. Amer. Math. Soc. (N.S.)*, vol. 23, no. 2, pp. 525–530, Oct. 1990.

[5] C. Fefferman and L. Seco, "Eigenvalues and eigenfunctions of ordinary differential operators", *Adv. Math.*, vol. 95, no. 2, pp. 145–305, 1992.

[6] C. Fefferman and L. Seco, "Aperiodicity of the Hamiltonian flow in the Thomas-Fermi potential", *Revista Matemática Iberoamericana*, vol. 9, no. 3, pp. 409–551, 1993.

[7] C. Fefferman and L. Seco, "The eigenvalue sum for a one-dimensional potential", *Adv. Math.*, vol. 108, no. 2, pp. 263–335, 1994.

[8] ——, "On the Dirac and Schwinger corrections to the ground-state energy of an atom", *Adv. Math.*, vol. 107, no. 1, pp. 1–185, 1994.

[9] ——, "The density in a three-dimensional radial potential", *Adv. Math.*, vol. 111, no. 1, pp. 88–161, 1995.

[10] C. Fefferman and L. Seco, "Interval arithmetic in quantum mechanics", in *Applications of interval computations*, Springer, 1996, pp. 145–167.

[11] C. Fefferman and L. Seco, "The eigenvalue sum for a three-dimensional radial potential", *Adv. Math.*, vol. 119, no. 1, pp. 26–116, 1996.

[12] C. Fefferman and L. Seco, "The density in a one-dimensional potential", *Adv. Math*, vol. 107, May 1997.

[13] A. M. Turing, "I.-Computing machinery and intelligence", *Mind*, vol. LIX, no. 236, pp. 433–460, 1950.

[14] E. J. Candès, "An introduction to compressive sensing", *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, 2008.

[15] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information", *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[16] A. Cohen, W. Dahmen, and R. DeVore, "Compressed sensing and best k-term approximation", *J. Amer. Math. Soc.*, vol. 22, no. 1, pp. 211–231, 2009.

[17] A. Chambolle, "An algorithm for total variation minimization and applications", *Journal of Mathematical Imaging and Vision*, vol. 20, no. 1, pp. 89–97, 2004.

[18] A. Chambolle and P.-L. Lions, "Image recovery via total variation minimization and related problems", *Numerische Mathematik*, vol. 76, no. 2, pp. 167–188, 1997.

[19] T. F. Chan, G. H. Golub, and P. Mulet, "A nonlinear primal-dual method for total variation-based image restoration", *SIAM J. Sci. Comput.*, vol. 20, no. 6, pp. 1964–1977, May 1999.

[20] D. L. Donoho, "Compressed sensing", *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[21] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit", *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, Jan. 2001.

[22] J. M. Fadili and G. Peyre, "Total variation projection with first order schemes", *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 657–669, Mar. 2011.

[23] G. Peyré, S. Bougleux, and L. D. Cohen, "Non-local regularization of inverse problems", in *Proc. of ECCV'08*, D. A. Forsyth, P. H. S. Torr, and A. Zisserman, Eds., ser. Lecture Notes in Computer Science, vol. 5304, Springer, 2008, pp. 57–68.

[24] R. Tibshirani, "Regression shrinkage and selection via the lasso", *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1996.

[25]   T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[26]   L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms", *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[27]   J.-F. Cai, B. Dong, S. Osher, and Z. Shen, "Image restoration: Total variation, wavelet frames, and beyond", *Journal of the American Mathematical Society*, vol. 25, no. 4, pp. 1033–1089, 2012.

[28]   Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[29]   S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[30]   P. Bürgisser and F. Cucker, *Condition : the geometry of numerical algorithms*, ser. Grundlehren der mathematischen Wissenschaften. Berlin, Heidelberg, New York: Springer, 2013.

[31]   Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*, ser. SIAM studies in applied mathematics. Philadelphia: Society for Industrial and Applied Mathematics, 1994.

[32]   Y. Nesterov, *Introductory lectures on convex optimization : a basic course*, ser. Applied optimization. Boston, Dordrecht, London: Kluwer Academic Publ., 2004.

[33]   A. Ben-Tal and A. S. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2001.

[34]   L. G. Khachiyan, "Polynomial algorithms in linear programming", *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, vol. 20, no. 1, pp. 51–68, 1980.

[35]   P. Gács and L. Lovász, "Khachiyan's algorithm for linear programming", in *Mathematical Programming at Oberwolfach*, Springer, 1981, pp. 61–68.

[36]   E. L. Lawler, "The great mathematical sputnik of 1979", *The Mathematical Intelligencer*, vol. 2, no. 4, pp. 191–198, 1980.

[37]   M. Grötschel, L. Lovász, and A. Schrijver, *Geometric algorithms and combinatorial optimization*, eng. Berlin [u.a.]: Springer, 1988.

[38]   N. Karmarkar, "A new polynomial-time algorithm for linear programming", in *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, ACM, 1984, pp. 302–311.

[39]   J. Renegar, "A polynomial-time algorithm, based on newton's method, for linear programming", *Mathematical Programming*, vol. 40, no. 1-3, pp. 59–93, 1988.

[40]   L. Lovász, "Discrete and continuous: Two sides of the same?", in *Visions in Mathematics: GAFA 2000 Special Volume, Part I*, N. Alon, J. Bourgain, A. Connes, M. Gromov, and V. Milman, Eds., Basel: Birkhäuser Basel, 2010, pp. 359–382.

[41]   L. Lovasz, *An Algorithmic Theory of Numbers, Graphs and Convexity*, ser. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1987.

[42]   L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and Real Computation*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1998.

[43]   L. Valiant, *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. New York, NY, USA: Basic Books, Inc., 2013.

[44]   A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem", *Proc. London Math. Soc.*, vol. S2-42, no. 1, p. 230, 1936.

[45]   J. Kepler, *Strena seu de nive sexangula*. 1611.

[46]   J. Ben-Artzi, M. J. Colbrook, A. C. Hansen, O. Nevanlinna, and M. Seidel, "On the Solvability Complexity Index hierarchy and towers of algorithms", *Preprint*, 2018.

[47] B. Adcock, A. C. Hansen, C. Poon, and B. Roman, "Breaking the coherence barrier: A new theory for compressed sensing", *Forum of Mathematics, Sigma*, vol. 5, pp. 1–84, Jan. 2017.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[49] J. Ben-Artzi, A. C. Hansen, O. Nevanlinna, and M. Seidel, "Can everything be computed? - On the Solvability Complexity Index and Towers of Algorithms", *arXiv:1508.03280v1*, 2015.

[50] A. C. Hansen, "On the solvability complexity index, the n-pseudospectrum and approximations of spectra of operators", *J. Amer. Math. Soc.*, vol. 24, no. 1, pp. 81–124, 2011.

[51] J. Ben-Artzi, A. C. Hansen, O. Nevanlinna, and M. Seidel, "New barriers in complexity theory: On the solvability complexity index and the towers of algorithms", *Comptes Rendus Mathematique*, vol. 353, no. 10, pp. 931–936, 2015.

[52] J. Renegar, "Linear programming, complexity theory and elementary functional analysis", *Mathematical Programming*, vol. 70, no. 1, pp. 279–351, 1995.

[53] ——, "Incorporating condition measures into the complexity theory of linear programming", *SIAM Journal on Optimization*, vol. 5, no. 3, pp. 506–524, 1995.

[54] P. Bürgisser and F. Cucker, "On a problem posed by Steve Smale", *Ann. of Math. (2)*, vol. 174, no. 3, pp. 1785–1836, 2011.

[55] K. Ko, *Complexity Theory of Real Functions*. 1991.

[56] S. Arora and B. Barak, *Computational Complexity - A Modern Approach*. Princeton University Press, 2009.

[57] A. Nemirovski, "Polynomial time methods in convex programming", in *The Mathematics of Numerical Analysis*, M. S. J. Renegar and S. Smale, Eds., vol. 32, AMS-SIAM Summer Seminar on Applied Mathematics, 1995, pp. 543–589.

[58] J.-M. Muller, *Elementary Functions: Algorithms and Implementation*. Birkhauser, 2005.

[59] Н. С. Закревская and А. П. Ковалевский, "Однопараметрические вероятностные модели статистик текста", Сиб. журн. индустр. матем., vol. 4, no. 2, pp. 142–153, 2001.

[60] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, 1970.

[61] S. Foucart and H. Rauhut, *A mathematical introduction to compressive sensing*. Springer, 2013.

[62] M. Rudelson and R. Vershynin, "On sparse reconstruction from fourier and gaussian measurements", *Communications on Pure and Applied Mathematics*, vol. 61, no. 8, pp. 1025–1045, 2008.

[63] K. Engel, *Sperner Theory*, ser. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1997.

[64] B. Adcock and A. Hansen, *Compressive Imaging: Structure, Sampling, Learning*. Cambridge University Press, 2021.

[65] L. Thesing and A. C. Hansen, "Linear reconstructions and the analysis of the stable sampling rate", *Sampling Theory in Signal and Image Processing*, 2018.

[66] V. Studer, J. Bobin, M. Chahid, H. S. Mousavi, E. Candes, and M. Dahan, "Compressive fluorescence microscopy for biological and hyperspectral imaging", *Proceedings of the National Academy of Sciences*, vol. 109, no. 26, E1679–E1687, 2012.

DAMTP, University of Cambridge
Email address: ab2053@cam.ac.uk

DAMTP, University of Cambridge
Email address: ach70@cam.ac.uk

D-ITET, ETH Zürich
Email address: vlacicv@mins.ee.ethz.ch