Cryptographic approaches to security and optimization in machine learning

Kevin Shi

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2020

Abstract

Cryptographic approaches to security and optimization in machine learning

Kevin Shi

Modern machine learning techniques have achieved surprisingly good standard test accuracy, yet classical machine learning theory has been unable to explain the underlying reason behind this success. The phenomenon of adversarial examples further complicates our understanding of what it means to have good generalization ability. Classifiers that generalize well to the test set are easily fooled by imperceptible image modifications, which can often be computed without knowledge of the classifier itself. The adversarial error of a classifier measures the error under which each test data point can be modified by an algorithm before it is given as input to the classifier. Followup work has showed that a tradeoff exists between optimizing for standard generalization error versus for adversarial error. This calls into question whether standard generalization error is the correct metric to measure.

We try to understand the generalization capability of modern machine learning techniques through the lens of adversarial examples. To reconcile the apparent tradeoff between the two competing notions of error, we create new security definitions and classifier constructions which allow us to prove an upper bound on the adversarial error that decreases as standard test error decreases. We introduce a cryptographic proof technique by defining a security assumption in a simpler attack setting and proving a security reduction from a restricted black-box attack problem to this security assumption. We then investigate the double descent curve in the interpolation regime, where test error can continue to decrease even after training error has reached 0, to give a natural explanation for the observed tradeoff between adversarial error and standard generalization error.

The second part of our work investigates further this notion of a black-box model by looking at the separation between being able to evaluate a function and being able to actually understand it. This is formalized through the notion of function obfuscation in cryptography. Given some concrete implementation of a function, the implementation is considered obfuscated if a user cannot produce the function output on a test input without querying the implementation itself. This means that a user cannot actually learn or understand the function even though all of the implementation details are presented in the clear. As expected this is a very strong requirement that does not exist for all functions one might be interested in. In our work we make progress on providing obfuscation schemes for simple, explicit function classes.

The last part of our work investigates non-statistical biases and algorithms for nonconvex optimization problems. We show that the continuous-time limit of stochastic gradient descent does not converge directly to the local optimum, but rather has a bias term which grows with the step size. We also construct novel, non-statistical algorithms for two parametric learning problems by employing lattice basis reduction techniques from cryptography.

Table of Contents

List of Figures

Chapter 1: Introduction

Deep learning has achieved breakthrough success in a wide variety of domains such as image recognition, natural language processing, and reinforcement learning for game playing. However, despite many such examples of empirical success, we have a very limited understanding of how deep learning achieves these results. State of the art deep neural networks with many more parameters than training data points have the capacity to fit random noise perfectly, which renders classical statistical learning theory results on generalization inapplicable to this setting. Understanding the success of deep learning requires rethinking generalization theory [1].

The recent phenomenon of adversarial examples [2] calls into question whether standard generalization error is even the correct metric to optimize for. State of the art neural networks make extremely brittle predictions on test data points, meaning the ℓ_p distance from a test data point to the decision boundary towards an incorrect class can be extremely small. In the domain of image recognition, this means that images can be modified in a way such that they are visually indistinguishable from the original image by humans, yet are incorrectly classified by standard models. This is problematic from both an implementation security perspective as well as a theoretical perspective, because such classifiers do not seem to fit our intuition of what it means for a model to be well-generalizing.

Current research on defenses against adversarial attacks is highly empirical, and researchers proposing new defense techniques are asked to validate against an ever-increasing library of attack algorithms. This makes it difficult to argue for the security of a defense against attacks that have yet to be tested or discovered. To make matters worse, researchers have noticed that trying to secure models against adversarial examples has the effect of decreasing standard generalization accuracy [3], causing an apparent contradiction between two competing definitions of accuracy. This is again problematic from both a practitioner's perspective, because high accuracy with respect to both measures is desirable, and from a theoretical perspective, because it seems unclear whether deep learning is actually doing the right thing. Our thesis aims to address these concerns and others by introducing new ideas from cryptography.

1.1 Thesis statement

We apply several ideas from cryptography, including proof techniques, security definitions, and algorithms, to address problems faced in modern machine learning. We show how these techniques can reconcile problems in adversarial machine learning, rigorously define models of information access to classifiers, and achieve improved algorithmic results for certain learning problems. We split our contributions into three chapters:

- We propose studying adversarial examples from the perspective of both defenses and explanations as a pathway to developing a new understanding of generalization. To that end, we develop new definitions, constructions, and proofs to achieve security guarantees in a restricted but still practical attack setting which at the same time is not at odds with standard generalization accuracy. We then try to explain the apparent observed tradeoff between robustness and accuracy as the fault of ill-behaved optimization algorithms in the presence of many weak features.

- We link together the two separate notions of white-box adversarial security and black-box adversarial security through the study of cryptographic obfuscation. We construct a provable obfuscation scheme for a function class with some resemblance to the majority voting of an ensemble classifier scheme.

- We investigate ideas outside of standard statistical optimization techniques to better understand the algorithmic aspects of nonconvex optimization problems in machine learning. We show quantitative non-statistical biases of stochastic gradient descent away from the global minimum by studying solutions continuous-time stochastic differential equations. We also propose the application of cryptanalysis algorithms such as LLL Lattice Basis reduction for solving nonconvex parameter recovery problems. We illustrate how this algorithm can be used to achieve significantly better sample complexity for two variants of classical machine learning problems.

1.2 Overview

Modern cryptographic protocols and constructions have many different complicated components, yet we can understand their security in simpler terms. This is because cryptography has solved this problem of empirical validation by reducing the overall security problem to a smaller and simpler-to-state problem that can be verified independently of the original security problem. Even though standard cryptographic assumptions such as the existence of one-way functions have never been mathematically proven, they have stood the test of time as one-way function candidates continue to resist empirical attacks. Cryptography makes progress by building more complicated constructions on top of these foundational assumptions. We want to apply this key principle to the field of adversarial machine learning.

However progress in cryptography often proceeds in the opposite direction, because we start with the final construction that we want to show security for. The challenge is finding a cryptographic assumption which is both believable and from which the final construction security can be reduced to. It is easy to define an assumption that is too strong to empirically verify, but one which makes the security proof trivial. Conversely, a security assumption that is too weak may not even admit a reduction from the final security problem. Defining the correct security assumption is a balancing act between these two extremes.

The second chapter of this thesis investigate new definitions, constructions, and proof techniques to achieve this in the realm of adversarial machine learning. We show that our new definitions allow for provable security guarantees in a restricted but still practical attack setting. Our security reduction yields an new adversarial attack setting which is applicable to standard classifiers and yields more reliable empirical validation. Our new security definitions are well-behaved with standard generalization accuracy in the sense that higher standard accuracy yields a tighter upper bound on adversarial error. This chapter is based off work in [4].

Our construction works in a black-box setting, which practitioners often think of in the machine learning as a service model, where users interact with a classifier hosted on an external server. A user is only allowed to remotely query this server and cannot see the underlying code running on the server which produces the classification output. However this model requires a completely trusted third-party server, and many algorithms which are cryptographically secure in a black-box setting are not secure when the source code is visible. Many services nowadays are run on cloud platforms such as Amazon Web Services, leading to a centralization of trust that a single company will not give out unauthorized access to the underlying code running on the platform. Cryptographers would prefer to replace this trust with a mathematical guarantee that the company hosting the server cannot glean any information from the code being run even if they acted maliciously.

The third chapter explores the field of cryptographic obfuscation which addresses this problem. The goal of obfuscation is to encrypt a function such that anyone can obtain input-output behavior from the encrypted function, but no other information can be extracted. This formalizes what we mean by a black-box, but unfortunately positive results in black-box obfuscation have been limited. Historically, most research in this area has focused on trying to achieve obfuscation for very general function classes, which necessitated strong security assumptions that lacked empirical footing. We take the opposite approach of just trying to achieve obfuscation for a limited function class similar to an ensemble classifier voting scheme and basing the security off more standard and tested cryptographic assumptions. This work was previously published as [5].

The last chapter takes the opposite perspective of the attacker. We investigate alternative approaches to the standard statistical loss optimization view of machine learning. We propose that cryptanalysis can be viewed as a machine learning problem in the sense that the goal of cryptanalysis is to learn the input-output mapping of some target encryption function. We show how a standard cryptanalysis technique, the LLL Lattice Basis Reduction algorithm, can be modified to solve parameter recovery problems in machine learning. This chapter is based off previously published work [6] and [7] which explore two variants of classical machine learning problems.

Chapter 2: Adversarial machine learning

Current machine learning models are vulnerable at test time to adversarial examples, which are data points that have been imperceptibly modified from legitimate data points but are misclassified with high confidence. This phenomenon was first described by [2], who constructed a simple attack that resembled gradient descent on the feature space. This fast gradient sign method computed the gradient of the loss function with respect to the feature space, took the sign of the gradient values, and then added it to the feature values with a small constant factor. Followup work constructed more efficient attacks by iteratively applying this gradient method [8][9] or by solving a direct constrained optimization problem [10].

These attacks all required access to the explicit loss function and parameter settings of the trained classifier, and so black-box models which only revealed the final class label output of the model seemed like a potential method to hide the gradients. Unfortunately, a major show-stopper with black-box models is the phenomenon of transferability [11], where an adversarial perturbation computed for an independently trained model has a high chance of being a successful attack against a separate black-box oracle model. This independently trained model is called a substitute model. Even if the adversary is only given black-box oracle access to predicted labels, existing machine learning models are vulnerable to transfer learning attacks executed by training substitute models [12]. The transfer success rate is the probability that an adversarial example computed for the substitute model is also misclassified by the black-box oracle.

Direct query-based attacks such as zeroth order optimization [13] and boundary attack [14] have also emerged as alternative black-box attacks without training substitute models. These attacks initialize with any misclassified data point on the other side of the decision boundary and iteratively perform rejection sampling to find a misclassified point closer to the decision boundary. This technique requires at least 10^4 adaptive queries to the classifier, which means the choice of the next query point depends on the result obtained for the previous query points. In contrast, transfer-based attacks from training substitute models can succeed using a much smaller number of between 0 to 10 epochs of adaptive queries, where multiple queries can be presented simultaneously in each epoch.

Researchers have tried many avenues of constructing defenses to prevent these attacks. Previous work has attempted to train models to be explicitly robust to attacks by incorporating robustness into the optimization problem [15][16], by input transformations and discretization to reduce model linearity [17], or by injecting randomness at inference time [18]. However defenses based on robust training have been subsequently broken by changing the space of allowable perturbations [19], and other defenses have been broken by more sophisticated attacks [20].

Recent explanations suggest that the existence of adversarial examples is actually inevitable in high-dimensional spaces. [21] [22][23] suggest that these examples exist for any linear classifier with nonzero error rate under additive Gaussian noise. This vulnerability is a simple geometrical fact when the dimension d is large: because most of the mass of a Gaussian distribution is concentrated near the shell, the distance to the closest misclassified example is a factor $d^{1/2}$ closer than the distance to the shell. [24] argue that adversarial perturbations can actually be robust features for generalization, and thus their adversarial nature is just a misalignment with our natural human notions of robustness.

In light of the evidence for the inevitability of adversarial perturbations, one goal we can still hope to achieve is a computational separation between declaring their existence and finding one. We propose a solution which uses hidden random bits that behave like a cryptographic key, meaning that any instantiation of the random bits works with high probability, but an attacker should not be able to attack the overall classifier without knowing the random bits. The space of all possible random bits in our construction will be exponential in the number of classes, so guessing the random bits is intractable.

In order to hide the randomness in a single classifier, we use a black-box ensemble scheme in which the adversary learns only the output of the overall ensemble without learning the output of any individual classifiers. Previous ensemble techniques for increasing adversarial robustness only subsample or augment the training data within each class [25], whereas our ensemble samples random splits of the labels themselves within the overall multiclass classification setup. This means that the underlying classification problem is unknown to the adversary, and we argue that this randomness decreases the transfer success rate. In addition, our ensemble construction is allowed to abstain from making a prediction, which behaves functionally like a built-in adversarial example detector and amplifies the robustness gain within each individual classifier.

Because the scope of attacks an adversary can mount is so large, we restrict our adversary to a constant number of epochs of adaptive queries. This still captures practical attacks such as transfer-based attacks that train substitute models from a constant number of epochs, but does not capture iterative attacks making tens of thousands of adaptive queries. In the case of just a single epoch of adaptive queries, we prove that the adversarial test error converges to twice the standard test error as the number of classes increases. The proof is based on a new security assumption which is in principle simpler to empirically verify than the entire construction, and we provide evidence for it on CIFAR-10 against projected gradient descent [8] and momentum iterative gradient method [9] attacks. We also provide empirical evidence of the effectiveness of this defense against 10 epochs of adaptive queries on the MNIST and CIFAR-10 data sets using a standard substitute model attack benchmark by [26].

2.1 Preliminaries

Let $\mathcal{X} \subset \mathbb{R}^d$ be the feature space, and let $\mathcal{Y} = \{1, 2, \ldots, N\}$ be the set of classes. The learning problem is to construct a multiclass classifier $F : \mathcal{X} \to \mathcal{Y} \cup \{\omega\}$ that is allowed to abstain from making a prediction by returning the symbol ω. We assume all classifier training is conducted using a fixed training algorithm for binary classification ML which is public knowledge. ML takes as input a set of binary-labeled data points $\{(x_i, z_i)\}_{i=1}^n$, where each $x_i \in \mathcal{X}$ and $z_i \in \{\pm 1\}$, and outputs a binary classifier $f : \mathcal{X} \to \{\pm 1\}$. The multiclass training data $\{(x_i, y_i\})$ is public knowledge, and the binary classifiers are trained over this data set by defining a mapping ϕ :

$\{1, \ldots, N\} \to \{\pm 1\}$ that takes each data point (x_i, y_i) to $(x_i, \phi(y_i))$. Furthermore, we assume that $\mathsf{ML}(\{(x_i, z_i)\})_{i=1}^n = -\mathsf{ML}(\{(x_i, -z_i)\})_{i=1}^n$, which just means that if the labels -1 and 1 were reversed in the training data, then the trained classifier would be identical except for outputting the opposite sign . Lastly, we fix some space $\mathcal{P} \subset \mathcal{X}$ to be the set of allowable adversarial perturbations; a commonly used perturbation space is $\{\rho \in \mathcal{X} \mid \|\rho\|_\infty < c\}$, which for example constrains each pixel in an image to be modified by a small vaule.

2.1.a Threat model

We consider the setting of a server hosting a fixed classifier $F : \mathcal{X} \to \{1, \ldots, N, \omega\}$ and users who interact with the server by presenting a query $q \in \mathcal{X}$ to the server and receiving the output label $F(q)$. We call F a black-box classifier, because the user does not see any of the intermediate computation values of $F(q)$. Two types of users access the server: honest users who present queries drawn from a natural data distribution, and adversarial users who present adversarial examples designed to intentionally cause a misclassification. The desired property is to serve the honest users the true label while simultaneously preventing the adversarial users from causing a misclassification; the latter is accomplished by either continuing to return the true label on adversarial examples or by returning the abstain label ω.

In order for this distinction to be well-defined, we need to separate natural misclassified examples from adversarial examples. We achieve this by fixing in advance a data point x which is correctly classified by $F(x)$ and requiring the adversary to compute a perturbation $\rho \in \mathcal{P}$ for this specific x such that $F(x + \rho) \notin \{F(x), \omega\}$. We think of x as a parameter of the attack, for example the natural image of the face of an attacker who wishes to masquerade as someone else. The classifier F is secure for x if, with high probability over the construction of F, the adversary cannot find a ρ satisfying this.

We formalize this attack problem by the notion of a *security challenge*. The adversary is given all the information about F except for any internal randomness used to initialize F. The adversary is then given the *challenge point* (x, y) with $F(x) = y$ being the correct classification, and the adversary successfully solves the security challenge if he finds a ρ such that $F(x+\rho) \notin \{\omega, F(x)\}$ with non-negligible probability. The solution to the security challenge is a successful attack.

The separation between existence of a solution and feasibility of finding it is given by resource constraints on the adversary, most commonly in the form of runtime. We say that a security challenge is *computationally secure* if there does not exist an algorithm for finding a solution within these resource constraints. In addition to runtime, we also consider the constraint of how many times the adversary is allowed to interact with the classifier.

We make a distinction between these *query points* (denoted by q) and the *challenge point* (denoted by x), both of which are feature vectors in $\mathcal{X}$. Query points are arbitrarily chosen by the adversary for the purpose of learning more about the black-box F, and there is no notion of correctness for $F(q)$. The ability to obtain labels for arbitrary query points is the key factor that enables the adversary to mount more powerful black-box attacks; without query access, the attacker is limited to relatively simple transfer-based attacks from models trained on standard datasets. We leverage this distinction to obtain a provable security guarantee by using cryptographic proof techniques.

2.1.b Security proofs in cryptography

Instead of directly trying to prove the security of F, we define a simpler system f that is easier to empirically test and reason about. We then prove a reduction from the security challenge of F to the security challenge of f, which shows that F is at least as hard to attack as f. We define a *security assumption* that characterizes the hardness of attacking f. This security assumption is not mathematically proven to be true, but nonetheless defining the right assumption makes the reduction is useful, because this assumption can be easier to empirically study. If the security assumption is true, then F is secure. The security assumption we define is the hardness of attacking a new type of randomized classifier without any query access to it.

2.1.c Random binary classifiers

In a multiclass classification problem with labels $1, \ldots, N$, suppose we have a binary classifier $f : \mathcal{X} \to \{\pm 1\}$ for two particular classes y and t, where class y is mapped to $+1$ and class t is mapped to -1. An adversary is given a data point (x, y) with $f(x) = +1$, and the adversary wishes to attack this binary classifier by computing a perturbation ρ such that $f(x + \rho) = -1$. If f were a standard binary classifier trained on the y versus t classification problem, then this would be a straightforward transfer attack scenario. However, instead f is trained with all remaining $N - 2$ classes also having been randomly remapped to ± 1 with equal probability. In other words, for each class $k \notin \{y, t\}$, we sample a Rademacher random variable $z_k \sim \{\pm 1\}$ and assign every data point of original label k to the new binary label z_k. This random assignment does not change the original y-vs-t classification task when all data points are only of original class y or t. The resulting f corresponding to training with the random binary labels $\{\pm 1\}^{N-2}$ is a *random binary classifier*:

Definition 1 (Random binary classifier)**.** Let $\mathcal{D}$ be a distribution over $\{\pm 1\}^N$. The *random binary classifier* over $\mathcal{D}$ is the distribution of f over $z \sim \mathcal{D}$ where each training data point x_i is relabeled to ± 1 by z_{y_i}:

$$f_z := \mathsf{ML}\left(\left\{(x_i, z_{y_i})\right\}_{i=1}^{n}\right). \quad \square$$

The security challenge for the random binary classifier is to compute a perturbation that changes its output with high probability over the sampling of z.

Definition 2 (Security challenge for random binary classifier)**.** Let $f_z := \mathsf{ML}\{(x_i, z_{y_i})\}_{i=1}^{n}$. Let $z \sim \{\pm 1\}^N$ be a Rademacher random vector, and let $\mathcal{D}_{yt}$ be the distribution of z conditioned on $z_y = +1, z_t = -1$. The *security challenge* for a challenge data point (x, y), failure rate $\delta > 0$, and target label $t \neq y$ is to compute a perturbation $\rho \in \mathcal{P}$ which changes the output of $f_z(x)$ with failure rate no greater than δ:

$$\mathop{\mathbf{Pr}}_{z \sim \mathcal{D}_{yt}} [f_z(x+\rho) \neq f_z(x)] > 1 - \delta.$$

In particular, the adversary has no ability to obtain labels for query points from the random binary classifier. □

Note that the adversary has knowledge of two of the bits of z, corresponding to the original label y and some target label $t \neq y$. Our security assumption is that for any $\rho \in \mathcal{P}$, there is enough randomness in the remaining $N-2$ data classes such that the failure rate is non-negligible.

Assumption 1 (Security assumption)**.** Given an instance of the security challenge for a random binary classifier with parameters defined as in Definition 2, for any $\rho \in \mathcal{P}$, for all $c > 0$, there exists a constant $N_0 > 0$ such that

$$\mathop{\mathbf{Pr}}_{z \sim \mathcal{D}_{yt}} \left[f_z(x+\rho) \neq f(x)\right] \leq 1 - 1/N^c$$

whenever $N \geq N_0$. □

Note that this implicitly assumes $\mathcal{P}$ does not contain any non-adversarial perturbations, such as those of the form $x' - x$ where x' is a legitimate image of class t. This assumption also does not place any computational constraints on the adversary yet; the security comes from the randomness in $z \sim \mathcal{D}_{yt}$, which is sampled after ρ is already fixed. In Section 2.4.a, we experimentally justify this assumption by estimating the transfer success probability for all pairs of classes (y,t) in the CIFAR-10 dataset using the standard ℓ_∞-ball for $\mathcal{P}$ and two different state-of-the-art transfer attacks.

We give two reasons why this assumption is the right one to make. Firstly, the scope of attacks to analyze is greatly reduced when the attacker has no access to the classifier. The adversary can essentially only mount transfer learning attacks by training models on the public dataset. Secondly, we only require the probability of success of the adversary to be bounded below 1 by a constant, and the overall security of the ensemble can be boosted from this bound.

2.1.d Main construction

Recall that our goal is to construct a multiclass classifier $F : \mathcal{X} \longrightarrow \{1, 2, \dots, N, \omega\}$ which is allowed to abstain from making a prediction (as represented by the output ω), and an adversarial perturbation ρ is only considered a successful attack if $F(x + \rho) \notin \{F(x), \omega\}$.

Our ensemble construction is the error-correcting code approach for multiclass-to-binary reduction [27], except with completely random codes for security purposes.

Construction 1 (Random ensemble classifier)**.** Given a multiclass classification problem with labels $\mathcal{Y} = \{1, \dots, N\}$, a codelength M, and a threshold parameter $r \in (0, 1/3)$:

- Sample random matrix $Z \in \{\pm 1\}^{N \times M}$, where each $Z_{ij} \sim \{\pm 1\}$ independently and with equal probability
- For $j = 1, \dots, M$, construct the binary classifier $f_j = \mathsf{ML}\left(\{(x_i, Z_{y_i j})\}_{i=1}^{n}\right)$

Given a query data point x, compute output $F(x)$ by:

- Compute the predicted codeword vector $C(x) := (f_1(x), \dots, f_M(x))$
- Compute $(d^*, y^*) = \min_y \|Z_y - C(x)\|_H$, where y^* is the index and d^* is the Hamming distance to Z_{y^*}
- If $d^* < Mr$, then output y^*, else output ω □

In this construction, the codeword $Z_y \in \{\pm 1\}^M$ acts as the identity of class y, and thus the classification of a data point x is the class codeword which is closest to its predicted codeword $C(x)$. We should think of the free parameters as $M = \Omega(\text{poly}(N))$ and $r = O(1/N)$. M needs to be sufficiently large in order for the random ensemble classifier to be accurate on natural examples. The parameter r should be greater than the standard test error of a trained classifier, or otherwise the ensemble will abstain on too many legitimate test samples. However r must be small enough for security purposes, which we will quantify in our main theorem.

We give some intuition for why this construction has desirable security properties. In order for an adversary to change the overall output of some test point (x, y), he needs to change the output of sufficiently many binary classifiers f_j so that $C(x+\rho)$ is close to some codeword $Z_t, t \neq y$. But the Hamming distance between Z_y and Z_t is $M/2$ on expectation, and $x, x+\rho$ must be within distance Mr to Z_y, Z_t respectively. Since each f_i is constructed independently at random, the overall probability of success is exponentially decreasing in the probability of successfully changing the output of an individual classifier.

We proceed to define the security challenge for this construction. We will use the shorthand notation $Z \sim \{\pm 1\}^{N \times M}$ to denote the distribution of $Z \in \{\pm 1\}^{N \times M}$ where each entry is independently sampled from $\{\pm 1\}$ with equal probability.

Definition 3 (Security challenge for random ensemble)**.** Let $F_Z(\cdot)$ be the ensemble classifier constructed with random hidden code matrix Z as defined in Construction 1. The *security challenge* for a challenge data point (x, y) and accuracy $\varepsilon \in (0, 1)$ is a two-round protocol:

1. Provide Q nonadaptive queries to $F_Z(\cdot)$ and receive answer labels, denoted by $\{(q_k, a_k)\}_{k=1}^{Q}$. The queries cannot depend on the hidden random code Z, but can otherwise depend on the public information such as the training data and the oracle ML.

2. Return a perturbation $\rho \in \mathcal{P}$ by some function of the query answers $\rho = \phi(\{a_k\}_{k=1}^{Q})$ such that ρ satisfies

$$\Pr_{Z \sim \{\pm 1\}^{N \times M}} \left[F_Z(x+\rho) \notin \{F_Z(x), \omega\} \right] > \varepsilon,$$

An algorithm for solving the security challenge is determined by its query set $\{q_k\}_{k=1}^{Q}$ and the function ϕ for computing the final perturbation from the query answers. □

For example, one possible attack captured by this definition is training a substitute model with a one epoch of data augmentation obtained from querying the classifier, as described by [12]. The adversary starts with a pre-labeled dataset of arbitrary size, usually the public training data set, and trains an initial substitute model. The adversary then refines this initial model by using Jacobian data augmentation to add new synthetic data points to the training data. In each epoch of data augmentation, the adversary obtains labels for these synthetic points using the black-box classifier.

The synthetic data points are the queries $q_1, \ldots, q_Q$, and thus our proof guarantees security against a single epoch of data augmentation. The actual implementation of this attack in [26] uses 10 substitute training epochs, and our proof does not apply directly to this implementation, because the second round of queries can depend on the answers in the first round. Nonetheless, we show empirically in Section 2.4.b that our construction is still secure against the benchmark of 10 data augmentation epochs.

2.2 Security results

The main theoretical result is a reduction from solving the random classifier challenge to solving the random ensemble challenge. In our reduction, we make the simplifying assumption that the space of allowable perturbations $\mathcal{P}$ is the same in both security challenges. This allows us to get away with not explicitly defining which perturbations are adversarial and which are legitimate, because a perturbation which makes $x + \rho$ a legitimate image of the class t would solve both security challenges simultaneously. We also assume without loss of generality that r is chosen such that $Mr \in \mathbb{Z}$, because Hamming distance is an integer.

Theorem 4. *Suppose there exists an algorithm $\mathcal{A}$ that can solve the security challenge for the random ensemble with any threshold $r \in (0, 1/2)$ such that $Mr \in \mathbb{Z}$ using Q queries and with accuracy $\varepsilon \in (0, 1)$. Then there is an algorithm that can compute a perturbation ρ which solves the security challenge for a random binary classifier with failure rate*

$$\delta < 2\left(r + \sqrt{\frac{\log(1/\varepsilon)}{2M}}\right).$$

The algorithm succeeds in computing this perturbation with probability (over Z) at least

$$1 - 4NQ\sqrt{\frac{1-r}{2\pi Mr}}2^{-M(1-H_2(r))},$$

where $H_2(r) = -r\log_2 r - (1-r)\log_2(1-r)$ is the negative entropy function and can be bounded away from 1 when r is bounded away from $1/2$.

The theorem shows that if such an algorithm $\mathcal{A}$ exists, $r = O(1/N)$, and $M = \Omega(\text{poly}(N))$, then the failure rate decreases as $O(1/N^c)$ for some constant c, which contradicts the security assumption (Assumption 1). Conversely, if the security assumption is true, then an adversary cannot solve the security challenge for the random ensemble with $O(\text{poly}(N))$ nonadaptive queries to the ensemble classifier. When $r < \frac{\delta}{2}$ and the security assumption is true, the theorem gives the following upper bound on the adversarial test error:

$$\varepsilon < \exp\left(-2M\left(\frac{\delta}{2} - r\right)^2\right).$$

Recall that the parameter r needs to be greater than the standard test error of a random binary classifier for good standard test accuracy of the ensemble, but less than $\frac{\delta}{2}$ for good adversarial accuracy. The more accurate each random binary classifier is, the smaller we can set the value of r to be, which in turn gives a smaller upper bound on the adversarial test error ε. This shows that our definition of adversarial test error is compatible with standard test error.

We give a brief proof sketch here, deferring the full proof to Section 2.3. Given a single random classifier f_z, we can simulate the entire ensemble classifier F_Z by constructing the remaining $M-1$ random classifiers using the public data set and ML. However, we cannot apply $\mathcal{A}$ to F_Z directly, because in Definition 2 there is no query access to f_z. Thus we first show in Lemma 2.3.1 that we can simulate the output of the entire ensemble using only $M-1$ classifiers with high probability.

Applying the algorithm $\mathcal{A}$ the ensemble of $M-1$ classifiers produces an attack perturbation ρ. Since this simulates the ensemble of M classifiers with high probability, then this attack perturbation also applies to the entire ensemble of M classifiers. Now we want to compute the probability of the output of each individual classifier in the ensemble being changed, but the Q queries could potentially leak information about some column Z^j. We use Lemma 2.3.1 for each column j to show that this is not the case; i.e. that the query answers are completely determined by the remaining $M-1$ columns with high probability and thus independent of column j itself. Then we show in Lemma 2.3.3 that an overall success probability of ε gives an upper bound on δ for each individual classifier.

2.3 Proofs

Lemma 2.3.1. *Fix any query point q and threshold $r < 1/2$ such that $Mr \in \mathbb{Z}$. Given a random ensemble function $F_Z : \mathcal{X} \to \{1, \ldots, N\}$ with M independently and identically generated random classifiers and threshold $r < 1/3$, fix some $j \in \{1, \ldots, M\}$ and let $F_{Z^{-j}}$ denote the modified ensemble which ignores the jth random classifier and takes the vote over only the remaining $M-1$ classifiers. Then*

$$\Pr_{Z^{-j} \sim \{\pm 1\}^{N \times (M-1)}} \left[F_Z(q) \neq F_{Z^{-j}}(q) \right] \leq 4N \sqrt{\frac{1-r}{2\pi M r}} 2^{-M(1-H_2(r))},$$

where the probability is taken only over the matrix Z^{-j} and is independent of the column Z^j. $H_2(r) = -r \log_2 r - (1-r)\log_2(1-r)$ can be bounded away from 1 when r is bounded away from $1/2$.

The lemma shows that for any j, with high probability over Z^{-j} the query answer $F_Z(q)$ is independent of Z^j, so that no information is revealed by the queries about column j. In the following proofs we will use the shorthand $f_j := f_{Z^j}$, i.e. the random classifier constructed from the jth column of Z.

Proof. The only way the additional classification output of $f_j(q)$ can influence the decision of the entire ensemble of $F_{Z^{-j}}(q)$ is if the predicted codeword of length $M-1$ is on the decision boundary between some class i and the abstaining space corresponding to ω. In the boolean hypercube

$\{\pm 1\}^{M-1}$, the number of points that are at a distance of exactly k to any fixed point is $\binom{M-1}{k}$. Because we want our probability bound to hold true regardless of the value of f_j, we have to consider the possibility of $f_j(q)$ influencing the points on either side of the decision boundary. To account for this, we multiply the number by 2. Then over all N classes, the number of possible points on the decision boundary is at most $2N\binom{M-1}{Mr}$ by a union bound.

$$\frac{2N}{2^{M-1}}\binom{M-1}{Mr} = \frac{4N(1-r)}{2^M}\binom{M}{Mr}. \tag{2.1}$$

We now apply the binomial coefficient upper bound from [28], reproduced below:

Lemma 2.3.2. *Suppose λn is an integer, where $0 < \lambda < 1$. Then*

$$\binom{n}{\lambda n} \leq \frac{1}{\sqrt{2\pi n\lambda(1-\lambda)}} 2^{nH_2(\lambda)},$$

where $H_2(\lambda) = -\lambda \log_2 \lambda - (1-\lambda)\log_2(1-\lambda)$ is the negative entropy function.

This gives the result

$$\binom{M}{Mr} \leq \frac{1}{\sqrt{2\pi Mr(1-r)}} 2^{MH_2(r)},$$

where $H_2(r) = -r\log_2 r - (1-r)\log_2(1-r)$ is the negative entropy function. Thus the probability in (2.1) can be bounded by

$$\frac{4N(1-r)}{2^M}\frac{1}{\sqrt{2\pi Mr(1-r)}}2^{MH_2(r)} \leq 4N\sqrt{\frac{1-r}{2\pi Mr}}2^{-M(1-H_2(r))}.$$

□

Since $H_2(r)$ is bounded away from 1 when r is bounded away from $1/2$, this gives an exponentially decaying probability bound in M.

The next lemma is a concentration result that holds when no information is revealed by the queries about any individual column.

Lemma 2.3.3. *Suppose that the event $f_j(x+\rho) \neq f_j(x)$ is independent and identical for each column j. Fix a data point (x, y). Given a perturbation ρ which solves the security challenge for the random ensemble with target probability $\varepsilon > 0$, then for every random classifier in the ensemble, ρ solves the security challenge for it with failure rate $\delta < 2(r + \sqrt{\log(1/\varepsilon)/2M})$*

Proof. Recall that the adversary is said to have solved the security challenge for the random ensemble if the vector of code bits $C_Z(x+\rho) := (f_1(x+\rho), \dots, f_M(x+\rho))$ has Hamming distance less than Mr to any other codeword Z_i, where $i \neq y$. Since each entry of the code matrix is sampled independently, we can consider the probability of this event bit-by-bit.

Let $\mathcal{E}_{tj}$ be the event where $f_j(x+\rho) = Z_{tj}$. Let $\mathcal{E}_t$ be the probability of the event where $\|C_Z(x+\rho') - Z_t\|_1 \leq Mr$, meaning the codeword for class t is the closest. By the independence assumption, we have $\mathbf{Pr}[\mathcal{E}_t] = \mathbf{Pr}[X > M(1-r)]$ where $X \sim \text{Binom}(M, \mathbf{Pr}[\mathcal{E}_{tj}])$, or equivalently,

$$\mathbf{Pr}[\mathcal{E}_t] = \mathbf{Pr}\left[X < Mr \,|\, X \sim \text{Binom}(M, 1 - \mathbf{Pr}[\mathcal{E}_{tj}])\right]. \tag{2.2}$$

The probability of changing $F(x)$ from y to any other class can be bounded by applying the union bound to all $t \neq y$. We obtain

$$\mathbf{Pr}[F_Z(x+\rho) \neq F_Z(x)] \leq (N-1)\,\mathbf{Pr}[\mathcal{E}_t],$$

and by the assumption of the lemma we know the left-hand side probability is $\delta > 0$. Thus we just need to compute $\mathbf{Pr}[\mathcal{E}_{ij}]$ and apply a tail inequality for the binomial distribution.

Fix one underlying code bit j and some other class $t \neq y$. Each bit Z_{tj} differs from the corresponding bit of C_{yj} with probability $1/2$ under the random code sampling scheme. Without loss of generality, we'll let $Z_{yj} = +1$. We analyze the probability of the event $f_j(x+\rho) = Z_{tj}$ by conditioning on Z_{tj}, obtaining

$$\begin{aligned}\mathbf{Pr}\left[\mathcal{E}_{tj}\right] &= \mathbf{Pr}\left[Z_{tj} = -1\right]\mathbf{Pr}\left[f_j(x+\rho) = -1 | Z_{tj} = -1, Z_{yj} = 1\right] \\ &\quad + \mathbf{Pr}\left[Z_{tj} = +1\right]\mathbf{Pr}\left[f_j(x+\rho) = +1 | Z_{tj} = +1, Z_{yj} = +1\right].\end{aligned}$$

We note that the term $\mathbf{Pr}[f_j(x+\rho) = -1 | Z_{tj} = -1, Z_{yj} = +1]$ is exactly the the probability $1-\delta$ in Definition 2. Then $\Pr[\mathcal{E}_{tj}]$ can be bounded by

$$\mathbf{Pr}[\mathcal{E}_{tj}] \leq \frac{1}{2}(1-\delta) + \frac{1}{2}(1) = 1 - \frac{\delta}{2}.$$

Then the probability in (2.2) can be bounded by using Hoeffding's inequality, which states that given $X \sim \text{Binom}(M, p)$, for any $\alpha > 0$,

$$\mathbf{Pr}\left[X \leq (p-\alpha)M\right] \leq \exp\left(-2M\alpha^2\right).$$

We let $X = \sum_j \mathcal{E}_{tj}$, so $p < 1 - \frac{\delta}{2}$ and $\alpha < p + r - 1 = r - \frac{\delta}{2}$. Applying Hoeffding's inequality with these parameters yields

$$\mathbf{Pr}[\mathcal{E}_t \leq Mr] \leq \exp\left(-2M\left(r - \frac{\delta}{2}\right)^2\right).$$

$\mathbf{Pr}[\mathcal{E}_t \leq Mr]$ is the probability of the perturbation ρ solving the security challenge for the random ensemble, so by the assumption in the lemma, this is at least ε. Thus we obtain

$$\varepsilon \leq \exp\left(-2M\left(r - \frac{\delta}{2}\right)^2\right).$$

We solve for δ as a function of ε to obtain the failure probability of solving the security challenge for an individual classifier:

$$\begin{aligned} \log(1/\varepsilon) &\geq 2M\left(\frac{\delta}{2} - r\right)^2 \\ \delta &\leq 2\left(r + \sqrt{\frac{\log(1/\varepsilon)}{2M}}\right). \end{aligned}$$

□

Proof of Theorem 4. We are given an instance of the security challenge for a random binary classifier (Definition 2). Let $f_{\overline{z}}$ be the random binary classifier, where $\overline{z} \sim \{\pm 1\}^N$ is uniformly sampled. We can simulate an entire random ensemble by constructing $M - 1$ additional random classifiers in the same way that $f_{\overline{z}}$ is sampled, so that $f_1 = f_{\overline{z}}$ and $f_2, \ldots, f_M$ are freshly sampled. Let Z^{-j} denote the matrix Z without the jth column, so that $F_{Z^{-j}} : \mathcal{X} \to \{1, \ldots, N\}$ denotes the output of the random ensemble ignoring f_j.

By the definition of the security challenge, the adversary cannot query f_1; however since $F_{Z^{-1}}$ is simulated by the adversary, he can make queries to $F_{Z^{-1}}$ and run $\mathcal{A}$ to produce a perturbation ρ attacking $F_{Z^{-1}}$. But if $F_{Z^{-1}}(q_i) = F_Z(q_i)$ for each query q_i, then $\mathcal{A}$ would have produced the same perturbation ρ attacking F_Z.

By Lemma 2.3.1 and a union bound over the number of queries, the hypothetical query answers $a_1, \ldots, a_Q$ to the entire ensemble F_Z depend only on $F_{Z^{-1}}$ with probability at least

$$1 - \Pr_{Z^{-1}}\left[\exists i\, F_{Z^{-1}}(q_i) \neq F_Z(q_i)\right] \geq 1 - 4NQ\sqrt{\frac{1-r}{2\pi M r}}2^{-M(1-H_2(r))}. \tag{2.3}$$

Now in order to apply Lemma 2.3.3 to bound ε as a function of δ, we want to show for each j that the event $f_j(x+\rho) \neq f_j(x)$ is independent of the query answers $a_1, \ldots, a_Q$. This can be done by applying Lemma 2.3.1 again to each column j to show that with high probability, the query answers only depend on the random sampling of Z^{-j}. Since $\rho = \phi(\{a_k\}_{k=1}^{Q})$ is a function of the query answers, then this means that the adversary's chosen ρ also only depends on Z^{-j}. We obtain

$$\begin{aligned} \mathop{\mathbf{Pr}}_{Z^j}\left[f_j(x+\rho) \neq f_j(x) \,|\, a_1, \ldots, a_Q\right] &= \mathop{\mathbf{Pr}}_{Z^j}\left[f_j(x+\rho) \neq f_j(x) \,|\, Z^{-j}\right] \\ &= \mathop{\mathbf{Pr}}_{Z^j}\left[f_j(x+\rho) \neq f_j(x)\right], \end{aligned}$$

and we see that this probability has no dependence on the actual column j since Z^j is independent and identical for each j. We incur a factor M in the probability of failure by applying a union bound of the failure probability in (2.3) over all $j = 1, \ldots, M$. Thus the event $f_j(x+\rho) \neq f_j(x)$ is independent and identical for each column j with probability at least

$$1 - 4NQ\sqrt{\frac{M(1-r)}{2\pi r}}2^{-M(1-H_2(r))}.$$

Then by Lemma 2.3.3, the probability of ρ changing the output of $f_{\overline{z}}$ is at least

$$1 - 2\left(r + \sqrt{\frac{\log(1/\varepsilon)}{2M}}\right).$$

□

2.4 Empirical results

We provide empirical analysis on the security assumption (Assumption 1) and the adversarial test accuracy against black-box substitute model training attacks for the MNIST [29] and CIFAR-

10 [30] datasets. We use code from the CleverHans adversarial examples library [26] and from the MadryLab CIFAR10 adversarial examples challenge [31] for the base classifier architecture, training, and attacks. The only modification to the base classifier architecture was to change the output layer from dimension 10 to dimension 2 for a binary output; no further architecture tuning was performed to optimize natural accuracy.

2.4.a Analysis of random binary classifiers

First, we empirically estimate the transfer success rate for all pairs of classes. We train a sample size of 40 random binary classifiers and then compute an adversarial perturbation for each test data point and each target class. The perturbation is computed by using a pre-trained standard model for the respective dataset with all N output dimensions. We then compute whether each random binary classifier makes a different prediction on the original test data point versus the perturbed test data point. Finally, for each pair (y, t), we empirically estimate the probability of the output of $f_z(\cdot)$ being changed conditioned on $z_y \neq z_t$ and plot this. The goal of this analysis is to show that this probability is bounded below 1 by a constant.

We use the Projected Gradient Descent and the Momentum Iterated Gradient Descent transfer attacks on the cross-entropy loss with an ℓ_∞ norm bound of $\varepsilon = 8$. The pre-trained substitute is a w28-10 wide residual network [32], and the random binary classifiers are the same ResNet architecture but with two output dimensions instead of ten. We visualize the average-case success probability in an $N \times N$ grid where the (y, t) coordinate shows the attack success probability over original data points of class y and target label t. The color of each cell represents the probability using the Viridis color palette shown in Figure 2.1.

Figure 2.1: Viridis color palette, uniformly scaled from 0 to 1

Figure 2.2 shows the empirical success probabilities of the attack over the CIFAR-10 data set for all pairs of classes.

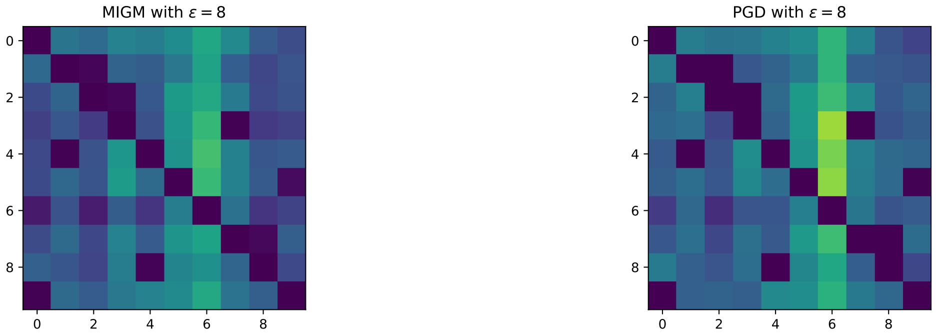

Figure 2.2: Success probabilities for targeted attacks on CIFAR-10 random binary classifiers

In the image, the cell $(4, 6)$ appears to have the highest probability, and the entire column $y = 6$ (frog) appears to have particularly high average success rate as a target class. For our security definition, we are interested in worst-case attack success rates, so we plot the distribution over each test data point for the (y, t) pairs $(4, 6)$ and $(5, 3)$. Figure 2.3 and Figure 2.4 show the individual success rates for MIGM and PGD, respectively.

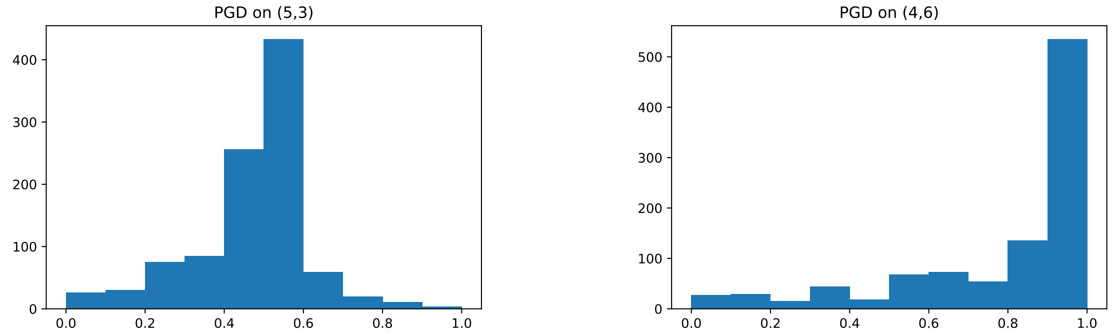

Figure 2.3: Distribution of success probabilities for individual CIFAR10 test data points under PGD attack

We see that among the (y, t) pairs where $t \neq 6$, the security definition needed for our main theorem is satisfied with high probability over the test examples. However, many of the examples are vulnerable to a targeted attack with target class $t = 6$. This suggests that the Frog class is especially distinct from the other 9 classes, such that even when it is randomly included in a binary partition, the neural network still builds a kind of frog detector separate from the other randomly

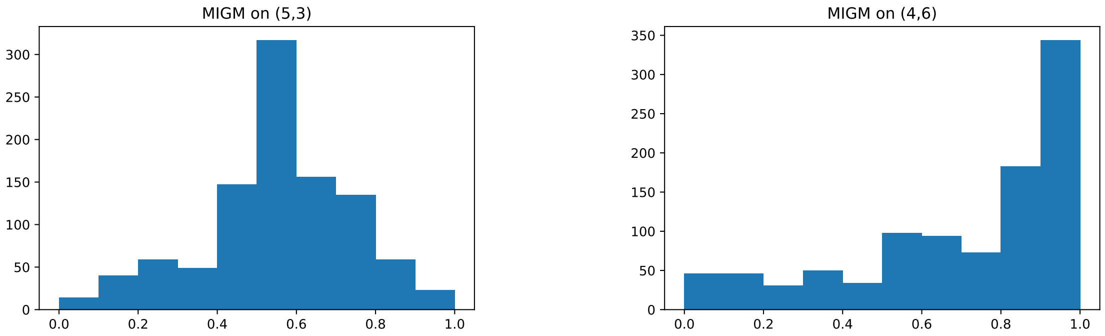

Figure 2.4: Distribution of success probabilities for individual CIFAR10 test data points under MIGM attack

included classes.

2.4.b Analysis of black-box adversarial accuracy

Next, we empirically analyze the robustness of our random ensemble construction to black-box transfer learning attacks. Instead of performing a transfer attack from a standard model, these attacks train a specific substitute model by querying the black-box classifier directly. We use the CleverHans attack library [26] to benchmark this. The attack algorithm trains a two-layer fully connected substitute model iteratively augmenting its training data set via queries to the random ensemble scheme and then uses the Fast Gradient Sign Method on the substitute model.

Because the attack library is not designed for querying classifier which abstains, we perform substitute model training with a non-abstaining random ensemble (i.e. $r = 1/2$). We consider the threshold r at the end when analyzing the final true and adversarial test accuracies. In order to incorporate the abstain label, we use the following definitions of accuracy for our experiments. The true test accuracy requires the classifier to make the correct, non-abstaining prediction. However when computing adversarial accuracy, we also consider it a success if the classifier outputs ω.

Definition 5 (True and adversarial test accuracy)**.** Given a multiclass classifier $F : \mathcal{X} \to \{1, \cdots, N, \omega\}$ which is allowed to abstain from making a prediction (as represented by the output ω), the relevant accuracy benchmarks are

$$\text{True accuracy} := \mathop{\mathbb{E}}_{(x,y)} \left[\mathbb{1}[F(x) = y]\right]$$

$$\text{Adversarial accuracy} := \mathop{\mathbb{E}}_{(\widehat{x},y)} \left[\mathbb{1}[F(\widehat{x}) \in \{y, \omega\}]\right],$$

where x is the original data point and $\widehat{x}$ is an adversarial perturbation of x. □

All random binary classifiers used in these experiments are the same architecture as the random binary classifiers in Section 2.4.a. Figure 2.5 shows that the ensemble enjoys good adversarial accuracy in the low-r regime.

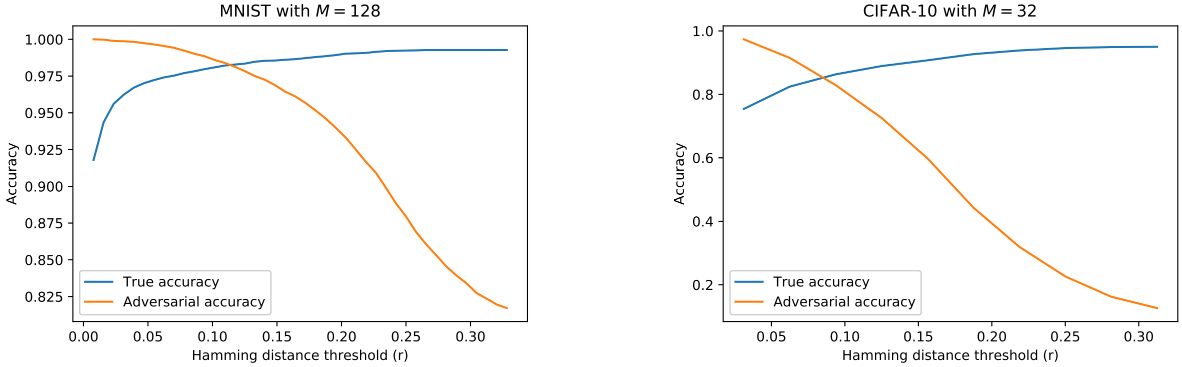

Figure 2.5: Accuracy versus Hamming distance ratio (r)

2.5 Analysis of adversarial features

While these black-box defenses are a promising line of research for constructing secure machine learning models, they do not give much insight as to why adversarial examples actually exist. The idea of non-robust features [24] is proposed as a mechanism to explain the tradeoff between adversarial error and standard test error. The non-robust features used by adversarial attacks are well-generalizing features that happen to be human-unrecognizable because of a misalignment between the ℓ_p metric and the metric in the intrinsic data space. The authors argue for the existence of non-robust features by two experiments. Firstly, removing non-robust features decreases standard test performance. Secondly, after mislabeling the original training data $\{x_i, y_i\}$ to $\{x_i + \Delta(x_i, y_{t_i}), y_{t_i}\}$, where $y_{t_i} \neq y_i$ is a different target class and $\Delta(x_i, y_{t_i})$ is an adversarial perturbation towards y_{t_i}, a trained classifier still achieves nontrivial test accuracy.

In this section, we look to understand this phenomenon in the interpolation regime in the second stage of the double descent curve [33]. While conventional machine learning suggests that the optimal test accuracy is achieved before training accuracy reaches 100%, the double descent curve suggests that test accuracy increases as we continue to train after 100% train accuracy, possibly even converging to a higher value. An intuitive explanation of this phenomenon, given by [34], is that overfitting incorrectly labeled training points more sharply decreases the size of the sample space which is incorrectly classified.

We give a simple model illustrating this phenomenon using purely random noise; that is, we show that augmenting noisy features with purely random features can in fact increase test accuracy in the interpolating regime. This is stronger than the result by [35] for linear regression, which showed only that random features were benign with regards to test risk. We show empirically that random features can increase classification performance for linear models and argue that random features increase the number of support vectors in an SVM, which can be a desirable behavior in the case of weak features. These results suggest that adversarial examples aren't necessarily a result of non-robust features, but rather of random noise being beneficial for helping standard optimization algorithms avoid overfitting to weak features. In other words, adversarial examples are a result of both suboptimal optimization algorithms and insufficient sample sizes.

2.5.a Random feature model

We consider a mixture of two Gaussians as a classification problem with label $y \in \{\pm 1\}$. Let $\mu \in \mathbb{R}_+^d$ be a mean vector with nonnegative entries. Let $A \in \mathbb{R}^{n \times d}$ be a matrix of independent noisy features, where each sample is drawn from $\mathcal{N}(y\mu, I_d)$. Let $B \in \mathbb{R}^{n \times k}$ be the matrix of independent random features, where each sample is drawn from $\mathcal{N}(0, \gamma^2 I_k)$. Let $X = [A|B]$ be the matrix of n data points, with $n/2$ generated from each class.

Since the covariance matrix is the identity, the Bayes optimal classifier for this is to estimate the empirical mean $\widehat{\mu}_A = \frac{1}{2n} A^t y$ over just the noisy features. Given a new test point (x, y) where $x = (a, b)$ is sampled as $a \sim \mathcal{N}(y\mu, I_d), b \sim \mathcal{N}(0, \gamma I_k)$, the classification output is sign$(\langle a, \widehat{\mu} \rangle) =$ sign$(\langle a, A^T y \rangle)$.

We illustrate the double descent phenomenon with an empirical example using the parameter settings $n = 2000, d = 500, \mu = 0.05, \sigma = 1.0, \gamma = 0.1$, and $k = 24000$. Figure 2.6 shows that test accuracy continues to increase while adversarial accuracy decreases as we add in additional random features even after training accuracy has reached 1.0.

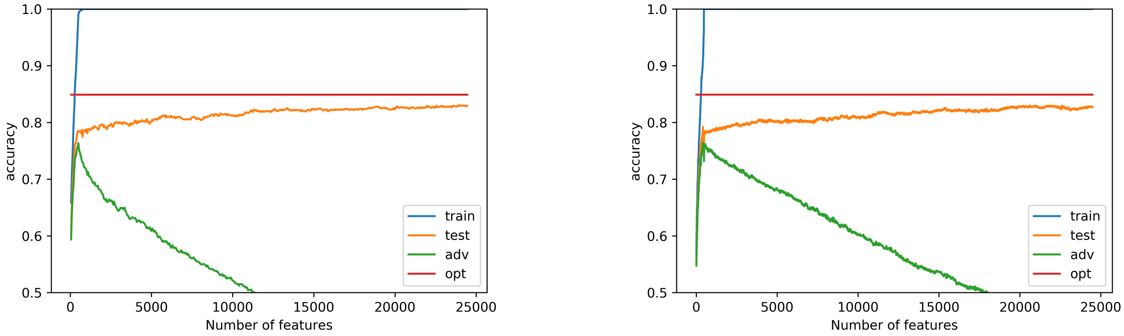

Figure 2.6: Test accuracy with random features. Left: SVM. Right: Logistic Regression

To show that these random features do indeed cause convergence to the correct model, we look at the signs of the feature weights for the d weak features. Every single weak feature is positively correlated with the label y, so each feature weight should be positive. Figure 2.7 shows that many feature weights are negative when using only weak features, but as random features are added to the model, the learned feature weights converge to the correct sign.

This suggests that the presence of random features decreases overfitting on the weak features. In the classical bias-variance regime, minimizing training error causes overfitting on the weights on the weak features in order interpolate the training data. In the interpolation regime, however, allowing the classifier to interpolate the training data using random, meaningless features makes the weights on the weak features more accurate. Because the random features are an order of magnitude smaller than the weak features, overfitting on the random features has diminished impact on

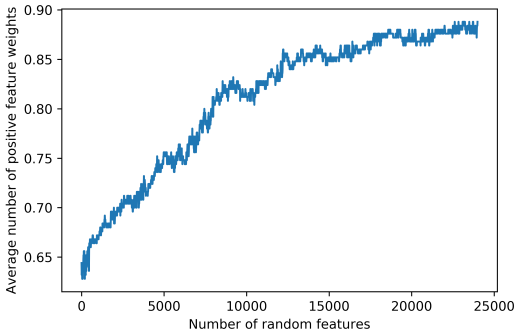

Figure 2.7: Average sign of weights for weak features.

classification accuracy over a test sample. We next analyze the effect of adding random features to an existing support vector machine solution in order to quantify how this may work.

2.5.b Support vector analysis

We consider the SVM model as an illustrative case which shows the effect of random features. The hinge loss has many of the properties of the ReLU activation function in terms of determining which data points are actively contributing to the weight vector, so the results can potentially be applied to explaining the activation of hidden units in a ReLU layer. Classical error bounds for support vector machines increase as the number of support vectors increases [36], but in the case of weak features, we show that the opposite effect occurs.

The loss function for the SVM objective is

$$L = \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2}\|w\|^2.$$

The first order conditions give the solution

$$\nabla L = \sum_{i=1}^{n} \mathbb{1}\left[y_i w^T x_i \leq 1\right](-y_i x_i) + \lambda w = 0$$
$$w = \frac{1}{\lambda}\sum_{i=1}^{n} \mathbb{1}\left[y_i w^T x_i \leq 1\right] y_i x_i.$$

The indicator function $\mathbb{1}[y_i w^T x_i \leq 1]$ indicates whether the data point (x_i, y_i) is a support vector. From now on we'll use the notation $\mathbb{1}_i := \mathbb{1}[y_i w^T x_i \leq 1]$. Note that the entire summation $\frac{1}{n}\sum_{i=1}^{n} y_i x_i$ is an estimate of the true mean vector μ, whereas the SVM solution is an estimate of the true mean vector using only support vectors as samples.

We show that random features cause the algorithm to use additional support vectors to estimate the weight vector. Let $\overline{w}$ be a solution to the original SVM objective with d weak features only, and consider the effect of adding k additional random features. We initialize the feature weights $\overline{w}_{d+1}, \ldots, \overline{w}_{d+k} = 0$ and analyze a two-stage optimization process of fitting the weight vector for the random features and then for the weak features. For each random feature r, we obtain the new weight

$$w_r = \frac{1}{\lambda}\sum_{s=1}^{n} \mathbb{1}_s y_s x_{sr}.$$

Thus for a data point (x_i, y_i), the new indicator function argument $y_i w^T x_i$ is distributed as

$$\begin{aligned} y_i w^T x_i &= y_i \overline{w}^T x_i + y_i \frac{1}{\lambda}\sum_{r=1}^{k}\sum_{s=1}^{n} \mathbb{1}_s y_s x_{sr} x_{ir} \\ &= y_i \overline{w}^T x_i + \frac{1}{\lambda}\sum_{r=1}^{k}\left[\mathbb{1}_i x_{ir}^2 + \sum_{s\neq i} \mathbb{1}_s y_i y_s x_{ir} x_{sr}\right]. \end{aligned}$$

We consider the case where (x_i, y_i) is not a support vector, in which case $\mathbb{1}_i = 0$. The additive term is equal in distribution to $\frac{\sqrt{km}}{\lambda} z_1 z_2$, where $m = \sum_{s \neq i} \mathbb{1}_s$ and $z_1, z_2 \sim \mathcal{N}(0, \gamma^2)$. Then the probability of (x_i, y_i) becoming a support vector is equal to

$$\mathbf{Pr}\left[z_1 z_2 > \frac{\lambda}{\sqrt{km}} (1 - y_i \overline{w}^T x_i) \right].$$

We apply the Paley-Zygmund inequality to obtain an anti-concentration bound on this probability:

$$\mathbf{Pr}\left[Z > \theta \, \mathbf{E}[Z] \right] \geq (1 - \theta)^2 \frac{\mathbf{E}[Z]^2}{\mathbf{E}[Z^2]}.$$

Letting $Z = |z_1 z_2|$ be the half-normal distribution and using the fact that $z_1 z_2$ is symmetric about the origin, we obtain

$$\begin{aligned}
\mathbf{Pr}\left[z_1 z_2 > \frac{\lambda}{\sqrt{km}} (1 - y_i \overline{w}^T x_i) \right] &= \frac{1}{2} \mathbf{Pr}\left[|z_1 z_2| > \frac{\lambda}{\sqrt{km}} (1 - y_i \overline{w}^T x_i) \right] \\
&= \frac{1}{2} \mathbf{Pr}\left[|z_1 z_2| > \frac{\pi}{2\gamma^2} \frac{\lambda}{\sqrt{km}} (1 - y_i \overline{w}^T x_i) \, \mathbf{E}\left[|z_1 z_2| \right] \right] \\
&\geq \frac{1}{2} \left(1 - \frac{\lambda \pi}{2\gamma^2 \sqrt{km}} (1 - y_i \overline{w}^T x_i) \right)^2 \left(\frac{2}{\pi} \right)^2 \\
&\geq \frac{1}{5} \left(1 - \frac{\lambda \pi}{2\gamma^2 \sqrt{km}} (1 - y_i \overline{w}^T x_i) \right)^2.
\end{aligned}$$

Thus as the number of random features k increases, at least $1/5$ of non-support vectors become support vectors and help in estimating feature weights for the weak features. If the original number of support vectors was small, then this explains why the addition of purely random features can increase standard test accuracy. Figure 2.8 confirms empirically that number of support vectors increases with the number of random features in the model.

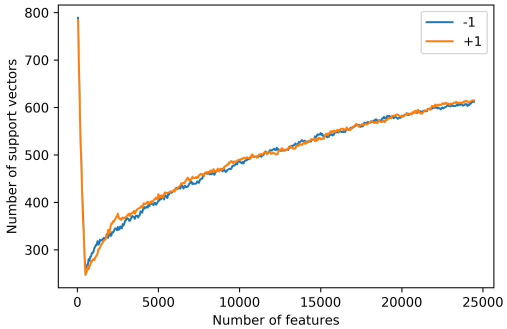

Figure 2.8: Number of support vectors. Total $n = 2000$

This example suggests that non-robust features are not necessarily meaningful or positively correlated with the class labels in any way. Rather, standard optimization algorithms used to train machine learning models can behave poorly in the presence of weak features, leading to a worse model estimate than the Bayes optimal estimator. Adding purely random features can help regularize this behavior.

2.6 Conclusion

This chapter makes some progress towards resolving the apparent misalignment between adversarial error and standard test error. We gave a novel construction with a provable guarantee that the adversarial test error converges to a constant multiple of the standard test error when the adversary is limited to black-box substitute model attacks. The security of our scheme can be quantitatively estimated this particular class of attacks by measuring our security assumption. Empirical results show that the security assumption holds for most pairs of classes, and the ensemble shows good accuracy against a benchmark substitute model training attack library.

Our security reduction makes it easy to analyze the security of new classifier architectures designed specifically for random binary classifiers. Optimizing the architecture of the individual classifiers used in the ensemble against our security assumption is an interesting direction of future work. One important item to note is that the random ensemble construction is not compatible with standard techniques of robust training. Robust training tends to decrease the standard test error of the classifier, which means that a larger threshold r needs to be used to account for natural errors in the individual random classifiers. However, a larger r value leads to weaker security in the ensemble.

We also gave an illustrative example of how purely random features can explain both vulnerability to adversarial perturbations and the double descent curve phenomenon in the case of linear models. It would be interesting to understand how random features generalize to the individual layers in a ReLU network. The random features result also suggests the following technique for defending against adversarial attacks. We embed each training image into a much larger unrelated image and train the classifier over this entire image using the original label. At test time, however, the unrelated image areas are dropped out, so the classifier is only applied to the original image area. The appended unrelated image behaves like random noise which the classifier can use to interpolate; however because it is dropped out at test time, the adversary cannot leverage these weights to attack the classifier.

Chapter 3: Cryptographic obfuscation

3.1 Introduction

The discipline of cryptography is fundamentally about the separation of seemingly intertwined information and abilities: how do we separate the ability the compute a function from the ability to invert a function? How do we separate the ability to encrypt from the ability to decrypt? How do we separate partial knowledge of a key through a side-channel attack from the ability to compromise a cryptographic scheme? The study of cryptographic obfuscation is born from the question: how do we separate the ability to run code from the ability to read code? Since the seminal work of [37] that placed this question firmly on a rigorous theoretical foundation, it has been clear that this kind of separation would be powerful, both inside and outside the typical reach of the discipline of cryptography.

If we can hide secrets inside functioning software, we can protect cryptographic keys, and many of cryptography's disparate and hard won achievements follow as a consequence. We can also protect intellectual property, and the inner workings of critical code like software patches, which in their unprotected form might leak information that could be used to attack remaining vulnerable machines. But as with any cryptographic primitive, the suitability of program obfuscation for any particular task depends on three main axes by which we must evaluate proposed constructions: (1) efficiency, (2) the underlying computational and architectural assumptions, and (3) the derived security guarantees.

Two possibiilities for (3), defined in [37], are the notion of virtual black box obfusction (VBB) and the notion of indistinguishability obfuscation (IO). Virtual black box obfuscation is a very powerful and intuitive notion, which requires that anything that can be done by an attacker in possession of the obfuscated code can also be done by a simulator who can only run the software in a “black box," with no access to intermediary values or other properties of the computation between input ingestion and output production. This notion would be suitable for virtually[1] all possible applications of obfuscation, but it is shown in [37] that it is impossible to achieve for general functionalities. The notion of IO requires something weaker, merely that an attacker in possession of two different obfuscations of the *same* functionality cannot tell them apart. In other words, we only enforce indistinguishability for program descriptions that may differ internally but whose external input/output behavior is *identical*.

At the time of its introduction by [37], IO was neither shown to be impossible, nor shown to be particularly useful. Progress instead was made for VBB obfuscation of very basic functionalities, such as point obfuscation [38, 39] and hyperplane membership [40], which lie below the reach of the impossibility result for VBB. But following the unprecedented construction of cryptographic multilinear maps in [41], two breakthroughs occurred in quick succession. A first candidate construction for indistinguishability obfuscation of general functions was proposed in [42], and the flexible technique of “punctured programming" was developed for deriving meaningful cryptographic results from the IO security guarantee [43].

Since then, the cryptographic research community has been riding out wave of positive and negative results: increasingly powerful constructions employing idealized models on multilinear maps or new, complex assumptions [42, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54], attacks on the underlying multilinear maps [55, 56, 57, 58, 59, 60], and a steady stream of works deriving applications and consequences from various forms of obfuscation(e.g., [61, 62, 63], and many more).

Our work is focused on the goal of obfuscating a modest but well-motivated functionality, one that does not require the use of multilinear maps, and hence does not inherit the risks of their still volatile security assumptions or the inefficiency that currently comes with using such a general-purpose tool. We consider the problem of *pattern matching with wildcards:* suppose there is an input binary string S of length n, and a pattern specification P also of length n, where for each bit

[1] Pun intended

P either dictates a particular bit value, or has a wildcard $*$, indicating that either value is allowed. For example, with $n = 5$, a pattern P would look like: $00 * 11$, and there would be two “matching" input strings S in this case, 00011 and 00111. The function we will obfuscate is the final “yes" or “no" outcome: for each P, we define the associated function $f_P(S)$ that outputs 1 when S matches P and outputs 0 otherwise.

This kind of functionality might appear, for instance, in a context like software patching. If a pattern P represents a problematic type of user input, say, that needs to be filtered out, we can obfuscate this function f_P to reject bad inputs without unnecessarily revealing P in full and helping attackers learn how to design such bad inputs. If the input length n is reasonably long and the number of matches to the pattern is not too dense in the space of inputs, we can hope that an attacker who queries a polynomial number of input strings will never manage to find a “bad" input that matches the pattern. We find these situations (where the adversary does not have enough information to identify the function being obfuscated) to be the most compelling subset of the standard VBB obfuscation security guarantee (as opposed to the subset involving simulating an adversary that already knows the function being obfuscated). Accordingly, we demonstrate that our construction satisfies a distributional security notion from [64, 65, 66]: if the pattern P is chosen from a suitable random distribution (and the number of wildcards $w \leq 0.75n$), then a PPT attacker will not be able to distinguish our obfuscation of f_P from an obfuscation of a function that always outputs 0.

Our construction uses only the basic tools of group operations and polynomial interpolation, and so is quite efficient. Our security analysis will be in the generic group model, for a regular cyclic group, with no multilinearity required. It remains an interesting open problem to obtain a security analysis in the standard model, using standard assumptions like DDH, for instance. [67] showed that the easier problem of bounded Hamming distance decoding is at least as hard as the DDH problem. While the result is not applicable to the obfuscation construction, the intermediary problem of finding nontrivial representations of the identity element first described by [68] is potentially applicable.

The functionality of pattern matching with wildcards has been previously obfuscated in [64, 65]. These constructions rely on multiplicative encoding schemes that enable multiplication of the encoded values and also zero-testing, i.e. checking whether an encoded value is zero. Unlike multilinear maps, these encoding schemes do not need to have additive properties. This functionality has been realized either through the use of general multilinear maps [64] or through lattice-based encodings relying on a new instance dependent assumption called entropic LWE [65]. A recent work by Wichs and Zirdelis [66] provides an obfuscation construction for a more general high entropy class, called compute-and-compare functions, from LWE. This class includes our pattern matching with wildcards. We view our construction as a simple and highly efficient alternative to such an LWE-based construction, and this is in line with the long tradition of analogous functionalities being achieved in the discrete-logarithm and LWE regimes.

To keep our scheme as intuitive and as efficient as possible, we start from additive basics. Let's first consider a pattern P with no wildcards. In this case, our function f_P is just a point function, since there is only one input string that matches the fully prescriptive pattern. Here we can work over Z_p and choose uniformly random values $a_1, \ldots, a_{n-1} \in Z_p$ and set $a_n = -(a_1 + \cdots + a_{n-1})$. We can choose additional random values $r_1, \ldots, r_n \in Z_p$. Now our obfuscated program can be comprised of $2n$ elements of Z_p, which we will label as $x_{i,b}$ where $i \in [n]$ and $b \in [0, 1]$. For each input bit position i, if the pattern value P is b, we set $x_{i,b} := a_i$ and $x_{i,1-b} = r_i$. To evaluate the obfuscated program on an input string, the evaluator simply selects the value corresponding to each input bit, and takes the sum modulo p. If it is 0, the output is 1. Otherwise the output is 0. Given these $2n$ values, if an attacker wants to find the pattern P, they are essentially trying to solve the subset sum problem (this is a slight variant since we have this kind of pair structure on the elements, but still the security intuition is the same).

Now if we want to introduce wildcards, it is clear we cannot simply give out a_i for both values for input bit i, since this will be noticed. The next thing we might try is to choose a random polynomial F of degree n over Z_p whose constant term is 0. Now we can set $x_{i,b} = F(2i + b)$ for positions that match the pattern, including both values of b in a wildcard position i. Our desired functionality can now be evaluated through polynomial interpolation. However, we quickly start to run into attacks based on list-decoding or regular decoding of Reed-Solomon codes, which can enable an attacker to recover the polynomial F once there are enough valid evaluations due to the wild cards.

A key observation at this point is that these decoding-style attacks rely upon non-linear functions of the given values, while the honest evaluation of the intended program needs only linear operations. This allows us to place the values $x_{i,b}$ in the exponent of a group $G = \langle g \rangle$ where discrete-log is difficult, and give out $g^{x_{i,b}}$ instead. This stops the decoding attacks without preventing honest evaluations. In the generic group model, the attacker is essentially limited to linear functions of the given exponents, so we can indeed formalize this intuition and obtain a security proof.

The hardness of noisy polynomial interpolation in the exponent was previously analyzed by [67], who gave a generic group argument concerning the problem of interpolating a polynomial with a slightly different error distribution. Our work follows a similar idea, but the specific wildcard structure we employ for our application creates some subtle differences, so we give a full argument here for completeness. We also provide a more rigorous exposition of the generic group proof argument.

It is an interesting problem to prove security for such a scheme without resorting to a generic group analysis. It seems that we should need a computational assumption like subset sum to assert that even though the group operations allow a discovery of the hidden structure, it is too sparse inside a combinatorially large space of possible input evaluations to be efficiently found. It also seems that we should need a computational assumption like DDH to explain exactly how the group blocks non-linear attacks. However, assumptions like DDH allow us to hide structure that is already non-linear, but requires us to preserve any structure that is linear, since linear structure on any small number of group elements can be discovered by brute force by an attacker. We could try to formulate some new assumption that is a strengthening of the subset sum assumption to the kind of intertwined linear structures that arise from polynomial evaluation, but this doesn't yet seem to yield insight beyond asserting security of the scheme itself. We would ideally like to see a hybrid argument that combined simple subset-sum like steps with simple DDH-like steps, but designing such a reduction remains an intriguing challenge. Given that LWE-based approaches in the standard model are known, this represents a new test case on the boundary of the analogies we know between DDH-hard groups and the LWE setting. We expect that further study of this disconnect in proof technology between the LWE setting and the DDH setting may yield general insights into the inherent relationships (or lack thereof) between these different mathematical underpinnings.

3.2 Preliminaries

3.2.a The generic group model

We will prove the security of our construction against *generic adversaries*, which interact with group elements via the generic group model as defined in [69]. In this model, an adversary can only interact with the group via oracle calls to its group operation and zero test functionality. Group elements are represented by “handles," which are uniformly random strings long enough that the small probability of collision between handles representing different group elements can be ignored. A generic group operation oracle takes as input two group handles and returns a new handle representing the group element that is the result of the group operation on the two inputs (and is consistent with all handles previously used). Note that such an oracle can be efficiently simulated using a lookup table.

We use $\mathcal{G}$ to denote such a generic group operation oracle that answers adversary calls. $\mathcal{A}^{\mathcal{G}}$ will denote an adversary given access to this oracle and $\mathcal{O}^{\mathcal{G}}$ will denote the set of handles generated by $\mathcal{G}$ corresponding to the group elements in the construction $\mathcal{O}$.

3.2.b Distributional virtual black-box obfuscation in the generic group model

We will use a definition of *distributional virtual black-box (VBB) obfuscation in the generic group model* which is essentially the definition of [64], except using the generic group model instead of the random graded encoding model:

Definition 6 (Distributional VBB Obfuscator)**.** Let $\mathcal{C} = \{C_n\}_{n \in \mathbb{N}}$ be a family of polynomial-size circuits, where $\mathcal{C}_n$ is a set of boolean circuits operating on inputs of length n, and let $\mathcal{O}$ be a ppt algorithm which takes as input an input length $n \in \mathbb{N}$ and a circuit $C \in \mathcal{C}$ and outputs a boolean circuit $\mathcal{O}(C)$ (not necessarily in $\mathcal{C}$). Let $\mathcal{D} = \{\mathcal{D}_n\}_{n \in \mathbb{N}}$ be an ensemble of distribution families $\mathcal{D}_n$ where each $D \in \mathcal{D}_n$ is a distribution over $\mathcal{C}_n$.

$\mathcal{O}$ is a *distributional VBB obfuscator* for the distribution class $\mathcal{D}$ over the circuit family $\mathcal{C}$ if it has the following properties:

1. Functionality-Preserving: For every $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, and $\vec{x} \in \{0,1\}^n$, with all but $negl(n)$ probability over the coins of $\mathcal{O}$:

$$(\mathcal{O}(C, 1^n)(\vec{x}) = C(\vec{x})$$

2. Polynomial Slowdown: For every $n \in \mathbb{N}$ and $C \in \mathcal{C}_n$, the evaluation of $\mathcal{O}(C, 1^n)$ can be performed in time $poly(|C|, n)$.

3. Distributional Virtual Black-Box in Generic Group Model: For every polynomial (in n) time generic adversary $\mathcal{A}$, there exists a polynomial time simulator $\mathcal{S}$, such that for every $n \in \mathbb{N}$, every distribution $D \in \mathcal{D}_n$ (a distribution over $\mathcal{C}_n$, and every predicate $P : \mathcal{C}_n \to \{0,1\}$:

$$|\,\mathbf{Pr}_{C \leftarrow \mathcal{D}_n, \mathcal{G}, \mathcal{O}^{\mathcal{G}}, \mathcal{A}}[\mathcal{A}^{\mathcal{G}}(\mathcal{O}^{\mathcal{G}}(C, 1^n)) = P(C)] - \mathbf{Pr}_{C \leftarrow \mathcal{D}_n, \mathcal{S}}[\mathcal{S}^{C}(1^{|C|}, 1^n) = P(C)]| = negl(n)$$

Remark 7. As in [64], we remark that a stronger notion of functionality-preserving exists in the literature, where the obfuscated program must agree with $C(\vec{x})$ on all inputs $\vec{x}$ *simultaneously*. We use the relaxed requirement that for every input (individually), the obfuscated circuit is correct except for negligible probability. We also note that our construction can be modified to achieve the stronger property by using a group of sufficiently large size (2^{2n}) and the union bound over each of the 2^n inputs.

Schwartz-Zippel Lemma

A key step in our hybrid proof of security relies on the Schwartz-Zippel Lemma, which we will reproduce here:

Lemma 3.2.1. *Let $\mathbb{Z}_p$ be a finite field of size p and let $P \in \mathbb{Z}_p[x_1, \ldots, x_n]$ be a non-zero polynomial of degree $\leq d$. Let $r_1, ..., r_n$ be selected at random independently and uniformly from $\mathbb{Z}_p$. Then:* $\mathbf{Pr}[P(r_1, ..., r_n) = 0] \leq \frac{d}{p}$.

3.3 Obfuscating pattern matching with wildcards

The class of functions for pattern matching with wildcards is parametrized by $(n, \vec{y}, \mathcal{W})$, where $\mathcal{W} \subset [n]$ is an index set and $f_{\vec{y}} : \{0,1\}^{n-|\mathcal{W}|} \longrightarrow \{0,1\}$ is a point function over $n - |\mathcal{W}|$ input variables that outputs 1 on the single input $\vec{y} \in \{0,1\}^{n-|\mathcal{W}|}$. The function $\Pi_{\mathcal{W}^c} : \{0,1\}^n \longrightarrow \{0,1\}^{n-|\mathcal{W}|}$ projects a boolean vector of length n onto only the entries not in the index set $\mathcal{W}$. $f_{\vec{y},\mathcal{W}}$, the function for pattern $\vec{y}$ with wildcard slots $\mathcal{W}$, is defined to be $f_{\vec{y},\mathcal{W}}(x) := f_{\vec{y}}\left(\Pi_{\mathcal{W}^c}(x)\right)$. Our obfuscation scheme for the class of functions for pattern matching with wildcards is as follows:

Setup(n): sample $a_1, \cdots, a_{n-1} \sim \mathbb{Z}_p$ uniformly at random and construct the fixed polynomial $F(x) := a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$. Let G be a group with generator g of prime order $p > 2^n$.

Construction$(n, \vec{y}, \mathcal{W})$: the obfuscator outputs $2n$ elements arranged in a $2 \times n$ table of n columns corresponding to the n input variables with two entries each corresponding to the two possible boolean values of each input. For each slot h_{ij} where $(i, j) \in \{0,1\}^n \times \{0,1\}$, if either $i \in \mathcal{W}$ or $y_i = j$, then the obfuscator releases the element $h_{ij} = g^{F(2i+j)}$. Otherwise, the obfuscator releases h_{ij} as a uniformly random element of G.

Evaluation$(\vec{x})$: to evaluate $f_{\vec{y},\mathcal{W}}(\vec{x})$, for each $i = 1, \cdots, n$, compute:

$$C_i := \prod_{j \neq i} \frac{-2j - x_j}{2i - x_i - x_j + 2j}$$

choose the elements h_{ix_i}, and compute:

$$T := \prod_{i=0}^{n-1} (h_{ix_i})^{C_i}$$

Output 1 if $T = g^0$ and 0 otherwise.

Functionality-Preserving: The fact that this obfuscation scheme is functionality-preserving follows from the fact that, if $\vec{x}$ is an accepting input of f ($f(\vec{x}) = 1$), then the chosen handles form n proper evaluations of the polynomial $F(x)$ on distinct elements. Further, the C_i scalars used in evaluation are Lagrange coefficients, making the evaluation a polynomial interpolation that returns $F(0) = 0$ in this case, causing $T = g^0$ and the evaluation to output 1 (with probability 1).

$$\begin{aligned}\prod_{i=0}^{n-1} (h_{ix_i})^{C_i} &= \prod_{i=0}^{n-1} g^{C_i F(2i+x_i)} \\ &= g^{\sum_{i=0}^{n-1} C_i F(2i+x_i)} \\ &= g^{F(0)} \\ &= g^0\end{aligned}$$

On the other hand, if even one input bit was not accepting (so $f(\vec{x}) = 0$), then at least one of the h_{ix_i}'s used in interpolation would be a uniformly random group element (not $g^{F(2i+j)}$). Thus, the evaluation product would be a product that includes a uniformly random group element raised to some power, which would result in $T = g^0$ with negligible probability $\frac{1}{p}$.

Polynomial Slowdown: Given a the set of $2n$ group elements, assuming group operations can be performed in $poly(n)$ time, the computation of C_i and T described in the **Evaluation** procedure can be performed in polynomial time.

Distributional Virtual Black-Box: We give a proof of our construction's distributional VBB security in the generic group model in Section 3.4 in Theorem 14.

3.4 Distributional VBB security in the generic group model

This section will prove Theorem 14, which establishes the distributional virtual black box security of our construction in the generic group model over the class of uniform distributions for point functions with wildcards. Our framework for reasoning in the generic group setting draws from [69].

The security proof shows that the obfuscation scheme constructed for a specific pattern matching with wildcards function $f_{\vec{y},\mathcal{W}}$ is indistinguishable from an obfuscation where all h_{ij} are random group elements. The analysis involves analyzing the performance of an adversary interacting with three different implementations of the generic group oracle. The first of these oracles operates according to an honest instantiation of the construction, while the third oracle operates according to an ideal instantiation where each group element is drawn from a uniformly random distribution.

Given that a low probability failure event does not occur, any algorithm's behavior when interacting with either of these oracles should be identical. The actual calculation of the probability of such a failure event is conceptually simple and done by many previous works for different noise distributions. On the other hand, properly formalizing the notion of "identical behavior" is where most of our new technical machinery is introduced.

In order to rigorously reason about the space of handles in a generic proof, we define an equivalent security game where an adversary calls two oracles simultaneously, one of whose behavior is already completely known. The purpose of incorporating a known oracle into the security game is to rigorously define when the unknown oracle deviates from expected behavior, and thus, when the adversary has distinguishing power. We make use of basic tools from category theory in order to describe the space of handles that the adversary has access to and when these handle sets become distinguishable.

In a generic group proof, there are many closely related but technically distinct kinds of objects that are often conflated. There are the underlying group elements, which can be associated with their exponents in $\mathbb{Z}_p$ relative to the common base. There are the handles that the group oracle associates to these elements. There are formal polynomials which may track known or unknown relationships between group elements. There are subsets of handles which the adversary has previously seen, and other handles whose distribution remains independent of the adversary's view so far. In order to make our proof as rigorous and precise as possible, we will keep explicit track of all of these various objects, and the maps between them.

We define an equivalent security game where an adversary calls two oracles simultaneously, one of whose behavior is already completely known. The purpose of incorporating a known oracle into the security game is to rigorously define when the unknown oracle deviates from expected behavior, and thus, when the adversary has distinguishing power. Given that a low probability failure event does not occur, any algorithm's behavior when interacting with either of these oracles should be identical. The actual calculation of the probability of such a failure event is conceptually simple and done by many previous works for different noise distributions. On the other hand, in order to properly describe the notion of "identical behavior" we introduce some basic technical machinery from category theory.

We establish some notation before proceeding. Let bold letters denote symbolic variables and non-bold letters denote the sampled random values for the corresponding variable. Let $f \in \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{x}]$ be a fixed polynomial of degree $n-1$ in $\mathbf{x}$ which is linear in each $\mathbf{a}_i$ individually. Let $\mathcal{H}_S$ and $\mathcal{H}_M$ be two identical copies of the same space of strings corresponding to handles in the generic group model.

Since our proof takes place in the generic group model, and our obfuscated program consists of a set of group elements, we will use the notation $\mathcal{G}_S, \mathcal{G}_M, \mathcal{G}_E$ to denote three different ways that an adversary can be supplied with handles representing an obfuscated program and how requests to the generic group operation oracle are answered. $\mathcal{G}_S$ will implement faithful interaction with the true construction in the generic group model. $\mathcal{G}_M$ implements a hybrid setting that we will show is indistinguishable from $\mathcal{G}_S$ to the adversary. Finally, $\mathcal{G}_E$ implements a setting that can be simulated without knowledge of the function drawn from the distribution (and is indistinguishable from $\mathcal{G}_M$).

The high level structure of our proof is pretty typical for a generic group argument. The group oracle $\mathcal{G}_M$ will behave similarly to $\mathcal{G}_S$, but instead of sampling random exponents according to the proscribed polynomial structure, it will work with formal polynomials representing this structure, hence ignoring any spurious relationship arises from a particular choice at the sampling stage. Arguing that $\mathcal{G}_S$ and $\mathcal{G}_M$ are indistinguishable is where we use the Schwartz-Zippel Lemma. An adversary will only receive a different distribution of handles if it manages to find a spurious relationship while interacting with $\mathcal{G}_S$, which must mean that the sampling happened to choose a root of a non-trivial, low degree formal polynomial. The Schwartz-Zippel Lemma allows us to conclude that this will occur with only negligible probability over the sampling employed by $\mathcal{G}_S$.

To argue that $\mathcal{G}_M$ and $\mathcal{G}_E$ are indistinguishable, we will need to argue that the adversary cannot (except with negligible probability), detect the remaining formal polynomial structure in $\mathcal{G}_M$, since doing so requires referencing many correctly structured elements and avoiding the random elements completely. As long as the wildcards are not too dense, this is an intractable combinatorial problem for the adversary.

Definition 8 ($\mathcal{G}_S$: Oracle *Start*)**.**

First, sample the following uniformly at random:

- $\mathcal{W} = \{i_1, \cdots, i_w\} \subset [n]$
- $y_i \in \{0, 1\}$ for each $i \notin \mathcal{W}$
- $a_1, \cdots, a_n \in \mathbb{Z}_p$
- Random embedding $\Phi_S : G \hookrightarrow \mathcal{H}_S$

For the initial set of handles representing the $2n$ group elements in the obfuscation of $f_{\vec{y},\mathcal{W}}$, for each entry $(i, j) \in [n] \times \{0, 1\}$:

- If $i \in \mathcal{W}$ or $y_i = j$ (input bit is part of an accepting string), output $\Phi_S\left(g^{F(a_1,\cdots,a_n,2i+j)}\right)$
- Otherwise sample a uniformly random exponent ρ_{ij} and output $\Phi_S(g^{\rho_{ij}})$

Given a group operation query on (h_1, h_2):

- Find $g_1 = \Phi_S^{-1}(h_1)$ and $g_2 = \Phi_S^{-1}(h_2)$. If either does not exist, ignore the query.
- Return $\Phi_S(g_1 \cdot g_2)$

Note that $\mathcal{G}_S$ faithfully instantiates our construction described in Section 3.3 in the generic group model. We will now describe an alternative oracle implementation that uses symbolic variables instead of group elements to produce the generic group functionality:

Definition 9 ($\mathcal{G}_M$: Oracle *Middle*)**.**

First, sample the following uniformly at random:

- $\mathcal{W} = \{i_1, \cdots, i_w\} \subset [n]$
- $y_i \in \{0, 1\}$ for each $i \notin \mathcal{W}$
- Random embedding $\Phi_M : \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] \hookrightarrow \mathcal{H}_M$.

Let $\sigma : \{0,1\}^n \times \{0,1\} \to [n-w]$ be an arbitrary ordering of the $(n-w)$ coordinate pairs (i, j) where $i \notin \mathcal{W}$ and $j \neq y_i$, and which is not defined on the other coordinate pairs.

For the initial set of handles representing the $2n$ group elements in the obfuscation of $f_{\vec{y},\mathcal{W}}$, for each entry $(i, j) \in [n] \times \{0, 1\}$:

- If $i \in \mathcal{W}$ or $y_i = j$ (i.e. the input bit is part of an accepting string), output $\Phi_M(F(\mathbf{a}_1, \cdots, \mathbf{a}_n, 2i+j))$
- Otherwise output the label $\Phi_M(\mathbf{b}_{\sigma(ij)})$

Given a group operation query on (h_1, h_2):

- Find $p_1 = \Phi_M^{-1}(h_1)$ and $p_2 = \Phi_M^{-1}(h_2)$. If either does not exist, ignore the query.
- Return $\Phi_M(p_1 + p_2)$

The two oracles are related by the existence of the following *evaluation map in the exponent*:

$$\begin{aligned} \phi : \mathbb{Z}[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] &\longrightarrow G \\ F(\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}) &\longmapsto g^{F(a_1, \cdots, a_n, b_1, \cdots, b_{n-w})} \end{aligned}$$

where $b_k = \rho_{\sigma^{-1}(k)}$ are the values of the random exponents sampled by Oracle S for the non-accepting slots. Only the existence of this evaluation map is necessary for the proof, so its dependence on unknown random values is not an issue.

In particular ϕ is a surjective group homomorphism of $\big(\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}], +\big)$ into $(G, \times)$, since it is a composition of an evaluation map with an exponential map, which are both surjective group homomorphisms.

The idea behind defining such an evaluation map is to define the failure event as a substructure of a larger structure which may then be used to formalize when the behavior is identical. In particular, we will see that the failure event corresponds to the kernel of this evaluation map that we just defined.

3.4.a Simultaneous oracle game

Rather than proving that the difference in any adversary's output probabilities when interacting with ($\mathcal{G}_S$ vs. $\mathcal{G}_M$) or ($\mathcal{G}_M$ vs. $\mathcal{G}_E$) is small directly, we will define another security game and exhibit a reduction to the desired statements. In this new security game, the adversary simultaneously queries two oracles for operations on group elements: one oracle $\mathcal{G}_M$ is known and serves as a convenience for formalizing the generic group oracle, and the second $\mathcal{G}_*$ is the unknown that the adversary wishes to identify. We define the game with oracles $(\mathcal{G}_S, \mathcal{G}_M)$ below and note that the game and reduction for oracles $(\mathcal{G}_M, \mathcal{G}_E)$ is symmetric.

Definition 10 (Simultaneous Oracle Game)**.** An adversary is given access to a pair of oracles $(\mathcal{G}_M, \mathcal{G}_*)$, where $\mathcal{G}_*$ is $\mathcal{G}_M$ with probability $1/2$ and $\mathcal{G}_S$ with probability $1/2$. In each round, the adversary asks the same query to both oracles. The adversary wins the game if he guesses correctly the identity of $\mathcal{G}_*$.

To make precise the notion of an adversary playing both oracles simultaneously and asking the same queries, the adversary maintains two sets $\mathcal{H}_S^t$ and $\mathcal{H}_M^t$ which are the sets of handles returned by the oracles after t query rounds. The adversary then maintains a function $\Psi : \mathcal{H}_M^t \to \mathcal{H}_S^t$. Initially, the adversary sets $\Psi(h_{ij}^b) = h_{ij}^a$ for each initial slot location $(i, j) \in \{1, n\} \times \{0, 1\}$, where h^a_{ij} is the handle corresponding to the slot (i, j) in oracles S and h^b_{ij} the handle in oracles M. After each query $h^m = \mathcal{G}_M(h^b_1, h^b_2)$ and $h^s = \mathcal{G}_S(\Psi(h^b_1), \Psi(h^b_2))$ the adversary updates the function with the definition $\Psi(h^s) = h^m$.

Lemma 3.4.1. *Suppose there exists an algorithm $\mathcal{A}$ such that*

$$\left| Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] \right| \geq \delta$$

Then an adversary can win the simultaneous oracle game with probability at least $\frac{1}{2} + \frac{\delta}{2}$ for any pair of oracles $(\mathcal{G}_M, \mathcal{G}_* = \mathcal{G}_M/\mathcal{G}_S)$.

Proof. Let $p = \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$ and $q = \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]$. The adversary can estimate these parameters to within a bounded polynomial of the true parameter by simulating each oracle and $\mathcal{A}$'s behavior on each.

Without loss of generality, we can assume that $p \geq q$. Otherwise, we can define p, q to be the inverse quantities $\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 0], \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0]$ respectively.

The adversary will guess $\mathcal{G}_* = \mathcal{G}_M$ if $\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = 1$ and $\mathcal{G}_* = \mathcal{G}_S$ if $\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = 0$. The probability of success is given by

$$\begin{aligned}
\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = \mathcal{G}_*] &= \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_M]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] \\
&\quad + \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_S]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0] \\
&= \frac{1}{2} + \frac{1}{2}(p - q) \\
&\geq \frac{1}{2} + \frac{\delta}{2}
\end{aligned}$$

□

3.4.b Indistinguishability between *Start* and *Middle*

The following gives a criteria for overall indistinguishability of the output handle distributions.

Definition 11. The pair (h^s, h^m) of answers returned by $(\mathcal{G}_S, \mathcal{G}_M)$ after query number t is called identical if it satisfies one of the following:

1. $h^s \notin \mathcal{H}_S^t$ and $h^m \notin \mathcal{H}_M^t$

2. The oracles return handles $h^s \in \mathcal{H}_S, h^m \in \mathcal{H}_M$ respectively such that $\Psi(h^m) = h^s$

Note that in case (1), h^s and h^m are both freshly sampled uniformly random strings and their distributions are equal.

Lemma 3.4.2. *In the simultaneous oracle game with $\mathcal{G}_* = \mathcal{G}_S$, suppose for every query (h_1^m, h_2^m) to oracle M and corresponding query $(\Psi(h_1^m), \Psi(h_2^m))$ to oracle S, the answers returned are identical. Then for any algorithm $\mathcal{A}$, we have*

$$\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] = \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$$

Proof of Lemma 3.4.2. If we had swapped the oracles $\mathcal{G}_S$ and $\mathcal{G}_M$ and the adversary had used Ψ^{-1} instead of Ψ, the answer distributions would have been identical and $\mathcal{A}$ would have to produce the same output distribution. □

Remark 12. Note that this argument does not depend on the particular implementations of $\mathcal{G}_S, \mathcal{G}_M$, and therefore the lemma also holds for the pair of oracles $\mathcal{G}_M, \mathcal{G}_E$ (to be defined later in Definition 13).

Thus it suffices to show that

Lemma 3.4.3. *Suppose an adversary makes an arbitrary sequence of queries and receives answers*

$$\{h_t^s = \mathcal{G}_S(\Psi(h_{t1}^m), \Psi(h_{t2}^m))\}_{t=1}^Q$$
$$\{h_t^m = \mathcal{G}_M(h_{t1}^m, h_{t2}^m)\}_{t=1}^Q$$

Then with overall probability at least $1 - \dfrac{(Q+2n)^2}{p}$, for every t, h_t^s and h_t^m are identical as defined in Definition 11.

Proof. Initially each set of $2n$ handles given by each oracle are uniformly random strings and hence indistinguishable. The proof is by induction under the following hypothesis:

Suppose the adversary has made t queries so far and has $\mathcal{H}_S^t, \mathcal{H}_M^t$ satisfying the following:

1. For each query made so far, the answer distributions have been identical.

2. For every $h^s \in \mathcal{H}_S^t$, there exists a unique $f \in \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n]$ such that $\Phi_S \circ \phi(f) = \Phi_M^{-1}(f)$

We can state this inductive hypothesis this in the following commutative diagram:

$$\begin{tikzcd}
\mathbb{Z}_p[\vec{\mathbf{a}}, \vec{\mathbf{b}}] \arrow[r, "\Phi_M{,}\simeq"] \arrow[d, two heads, "\phi"'] & \mathrm{Im}(\Phi_M) & \mathcal{H}_M^t \arrow[l, hook', "i_M"'] \arrow[d, "\Psi{,}="] \\
G \arrow[r, "\Phi_S{,}\simeq"'] & \mathrm{Im}(\Phi_S) & \mathcal{H}_S^t \arrow[l, hook, "i_S"] \arrow[llu, "\exists!"]
\end{tikzcd}$$

Here $\mathrm{Im}(\Phi_M), \mathrm{Im}(\Phi_S)$ are the relevant handles in the handle spaces. Commutativity of the lower triangle under the unique lift means that for all $h^s \in \mathcal{H}_S^t, \exists! f \in \mathbb{Z}_p[\vec{x}]$ such that $i_S(h^s) = \Phi_S \circ \phi(f)$. Note that the upper triangle trivially commutes because the unique lift is defined by the composition $\Phi_M \circ i_M \circ \Psi^{-1}$. To ease the notation a little, we'll omit the inclusion maps from here on when it is obvious the handle is in $\mathcal{H}_*^t$.

Now assuming the inductive hypothesis, suppose the $(t+1)$th query is the group operation of $h_1, h_2 \in \mathcal{H}_M^t$ and $\Psi(h_1), \Psi(h_2) \in \mathcal{H}_S^t$. Oracle M will output the handle

$$h^m = \Phi_M\left(\Phi_M^{-1}(h_1) + \Phi_M^{-1}(h_2)\right) =: h_1 \cdot h_2,$$

and Oracle S will output the handle

$$h^s = \Phi_S\left(\Phi_S^{-1}(\Psi(h_1)) \times \Phi_S^{-1}(\Psi(h_2))\right) =: \Psi(h_1) \cdot \Psi(h_2).$$

The $(\cdot)$ notation on handles is justified by the fact that $\mathrm{Im}(\Phi_M) \subset \mathcal{H}_M$ is trivially isomomorphic as a group to $\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n]$, where its group operation is obtained by pulling back by Φ_M, and likewise for $\mathrm{Im}(\Phi_S) \subset \mathcal{H}_S$.

We have the following two cases:

1. $h^m \in \mathcal{H}_M^t$ (i.e. this handle was seen previously). Then

$$\begin{aligned}\Psi(h_1) \cdot \Psi(h_2) &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_1) \cdot (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_2) \\ &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_1 \cdot h_2) \\ &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h^m) \\ &= \Psi(h^m)\end{aligned}$$

   where we use commutativity of the diagram on each factor handle, the homomorphism property of the maps, the definition of oracle M's output, and commutativity of the diagram on the output handle (which we can do since the handle was previously defined).

   Thus the handles in the output pair have the same distribution, and since no new handles are created, the inductive hypothesis trivially remains satisfied.

2. $h^m \notin \mathcal{H}_M^t$ (i.e. this is a new handle).

   (a) If $h^s \notin \mathcal{H}_S^t$ is also a new handle, then the unique lift simply extends to map h^s to $\Phi_M^{-1}(h^m)$, and both $\mathcal{H}_M^t$ and $\mathcal{H}_S^t$ are augmented by one element. The handles in the output pair are new and uniformly distributed, and the inductive hypothesis is satisfied.

   (b) If $h^s \in \mathcal{H}_S^t$, then by the inductive hypothesis, h^s lifts to some $f_s \in \mathbb{Z}_p[\vec{\mathbf{x}}]$ which maps to some $\tilde{h}^b = \Psi^{-1}(h^s)$. However we also have $f_m = \Phi_M^{-1}(h^m) \neq f_s$, since $h^m \notin \mathcal{H}_M^t$. Thus both f_s and f_m are lifts of h^s which make the diagram commute, so after this query the inductive hypothesis is no longer satisfied for the next query.

   This event only happens if $f_s - f_m \in \ker \phi$ and $f_s - f_m$ is nontrivial. Thus the proof is complete as long as we show this event happens with low probability.

Now consider the following sequential variant of the game. The adversary plays the game using the real Oracle M and his own simulation of Oracle S obtained by outputting a uniformly random string when $\mathcal{G}_M$ does and using the Ψ map when $\mathcal{G}_M$ outputs an existing string. He then plays the exact same sequence to the real Oracle S and compares these answers to the ones produced by the real Oracle M. As long as the bad event does not occur, the sequence of queries asked in this sequential game is identical to the sequence of queries asked playing the real pair of oracles.

The occurrence of the bad event is decided by the initial random sampling of $a_1, \cdots, a_n \in \mathbb{Z}_p$, and thus the bad event either occurs in both the sequential and parallel variants or in neither. So it suffices to just bound the probability of the bad event occurring at any time in the sequential game.

For each pair (f_s, f_m), $f_s - f_m$ is a degree-1 polynomial in n variables over $\mathbb{Z}_p$. Thus the bad event happens with probability at most $\frac{1}{p}$ by Lemma 3.2.1, the Schwartz-Zippel lemma. Thus by a union bound, after Q queries of either type, there are at most $(Q + 2n)^2$ pairs of symbolic polynomials, so with probability at most $\frac{(Q+2n)^2}{p}$ the two distributions of handles are distinguishable.

□

We remark that everything in the proof only relied on diagram arguments and did not care about the actual structure of the underlying objects, except for analyzing when $f_s - f_m \in \ker \phi$ occurred. Thus in the proceeding reductions between other oracles, all this automatically follows provided we can define an appropriate evaluation map ϕ, and we only need to analyze the kernel of the corresponding evaluation map.

Lemma 3.4.4. *For an adversary $\mathcal{A}$ in the generic group model which makes Q queries to the generic group oracle,*

$$|\, \mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}(C, 1^n)) = P(C)] - \mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C, 1^n)) = P(C)]| \le \frac{(Q+2n)^2}{2^n},$$

where the probability is taken over the distribution $C \leftarrow \mathcal{D}_n, \mathcal{G}_S, \mathcal{O}, \mathcal{A}$

Proof. From Lemma 3.4.2 we have that:

$$\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] = \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$$

as long as all queries to the generic group oracles are *identical* as defined in Definition 11.

Lemma 3.4.3 tells us that the probabilities of all queries not being identical during the simultaneous oracle game between $(\mathcal{G}_S, \mathcal{G}_M)$ is at most $\dfrac{(Q+2n)^2}{p}$, where Q is the number of the adversary's queries to the generic group oracle and $p > 2^n$ is the order of the group.

Therefore, the difference $\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]$ is at most $\dfrac{(Q+2n)^2}{2^n}$, and so an adversary's advantage in the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_S)$ and $(\mathcal{G}_M, \mathcal{G}_M)$ is:

$$\begin{aligned}\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = \mathcal{G}_*] &= \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_M]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] \\ &\quad + \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_S]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0] \\ &= \frac{1}{2} + \frac{1}{2}(\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]) \\ &\leq \frac{1}{2} + \frac{(Q+2n)^2}{2 \cdot 2^n}\end{aligned}$$

This, plugged into the reduction from Lemma 3.4.1, tells us that for all adversaries:

$$\left| Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] \right| \leq \frac{(Q+2n)^2}{2^n}$$

□

3.4.c Game between *Middle* and *End*

Definition 13 ($\mathcal{G}_E$: Oracle *End*)**.**

First, sample the following uniformly at random:

- Random embedding $\Phi_E : \mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}] \hookrightarrow \mathcal{H}_E$.

For the initial set of handles representing the $2n$ group elements in the obfuscation of $f_{\vec{y},\mathcal{W}}$, for each entry $(i, j) \in [n] \times \{0, 1\}$:

- Output $\Phi_E(\mathbf{c}_{2i+j})$

Given a group operation query on (h_1, h_2):

- Find $p_1 = \Phi_E^{-1}(h_1)$ and $p_2 = \Phi_E^{-1}(h_2)$. If either does not exist, ignore the query.

- Return $\Phi_E(p_1 + p_2)$

Oracle M and Oracle E are related by the following *evaluation map* which is defined on the generators of $\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}]$ and extended by linearity.

$$\begin{aligned}
\phi : \mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}] &\longrightarrow \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] \\
\mathbf{c}_k &\longmapsto \mathbf{b}_{\sigma(\lfloor k/2 \rfloor, k \bmod 2)} \text{ if } \sigma \text{ is defined here} \\
\mathbf{c}_k &\longmapsto F(\mathbf{a}_1, \cdots, \mathbf{a}_n, k) \text{ otherwise}
\end{aligned}$$

In other words the monomial c_k is mapped to the same symbolic polynomial that Oracle *Middle* assigned to the slot $\big(\lfloor k/2 \rfloor, k \mod 2\big)$, which is either a symbolic variable **b** or a symbolic polynomial $F(\mathbf{a}_1, \cdots, \mathbf{a}_n, k)$. Since the $\mathbf{c}_k$'s generate the entire additive group $\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}]$, this extends to a group homomorphism of $\big(\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}], +\big)$ into $\big(\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}], +\big)$.

Lemma 3.4.5. *Suppose an adversary makes an arbitrary sequence of queries and receives answers*

$$\begin{aligned}
&\big\{h_t^m = \mathcal{G}_S(\Psi(h_{t1}^e), \Psi(h_{t2}^e))\big\}_{t=1}^{Q} \\
&\big\{h_t^e = \mathcal{G}_M(h_{t1}^e, h_{t2}^e)\big\}_{t=1}^{Q}
\end{aligned}$$

If $w/n \leq 3/4$*, then with overall probability at least* $1 - \frac{2}{2^{0.0613n}}$ *for every t,* h_t^s *and* h_t^m *are identical as defined in Definition 11.*

The proof of this lemma starts with the same setup as the proof of 3.4.3. The adversary maintains a function $\Psi : \mathcal{H}_E \to \mathcal{H}_M$ and two sets of handles $\mathcal{H}_E^t, \mathcal{H}_M^t$.

Proof. Inductively, after t queries, assume the following commutative diagram is true:

$$\begin{tikzcd}
\mathbb{Z}_p[\mathbf{c}_1, \cdots, \cdots, \mathbf{c}_{2n}] \arrow[r, "{\Phi_E,\simeq}"] \arrow[d, two heads, "\phi"'] & \mathrm{Im}(\Phi_E) & \mathcal{H}_E^t \arrow[l, hook', "i_E"'] \arrow[d, "{\Psi,=}"] \\
\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] \arrow[r, "{\Phi_M,\simeq}"'] & \mathrm{Im}(\Phi_M) & \mathcal{H}_M^t \arrow[l, hook', "i_M"] \arrow[ull, "\exists!"]
\end{tikzcd}$$

The same diagram chase from the proof of (3.4.3) tells us that the next pair of query answers (h^e, h^m) only fails to satisfy the inductive hypothesis if h^m lifts to $f_m \in \mathbb{Z}_p[\vec{\mathbf{c}}]$ by the inductive hypothesis, but$f_m \neq \Phi_E^{-1}(h^e) =: f_e$, so $f_m - f_e \in \ker\phi$ and $f_m - f_e$ is nontrivial. Necessary but not sufficient conditions for $f_m - f_e$ to be in the kernel of ϕ are:

1. $f_m - f_e$ must have a zero coefficient in front of any $\mathbf{c}_k$ that is defined under the σ map, since each free variable $\mathbf{b}_j$ has a unique preimage.

2. $f_m - f_e$ must have at least $n-1$ nonzero coefficients

As with the proof of (3.4.3), we analyze the sequential variant where the adversary plays a sequence of queries to $\mathcal{G}_E$ and then plays the exact same sequence of queries to $\mathcal{G}_M$. After Q queries the adversary has at most $Q + 2n$ symbolic polynomials in $\mathbb{Z}_p[\vec{\mathbf{c}}]$. For each pair of polynomials f_m, f_e in this set, the variables $\mathbf{c}_k$ are mapped by the initial random sampling of the wildcard slots by Oracle M.

Now suppose the adversary fixes a polynomial containing $n-1$ nonzero coefficients of the $\mathbf{c}_k$'s such that m columns in the original table of $2n$ entries have nonzero coefficients for both entries in the column. This means that the oracle must necessarily choose those m columns to be wildcard slots, since otherwise one of the two entries in the column will not be in the kernel of the ϕ map.

This means that the probability over the initialization of the oracle that these m columns are all chosen to be wildcard slots is $\frac{\binom{n-m}{w-m}}{\binom{n}{w}}$. The remaining $n - 1 - 2m$ columns each must either match the entry chosen by the adversary or be a wildcard slot. There are $(n - 1 - 2m) - (w - m) = (n - 1 - w) - m$ slots that cannot be wildcard slots and thus have at most probability $1/2$ each of matching the entry chosen by the adversary. Thus the probability that this polynomial is in the kernel of ϕ is

$$\frac{\binom{n-m}{w-m}}{\binom{n}{w}} \left(\frac{1}{2}\right)^{n-1-w-m} \tag{3.1}$$

An upper bound for this can be computed by maximizing the expression with respect to the adversary's choice of m. If we increment m by 1, the first factor is multiplied by $\frac{w-m}{n-m}$ while the second factor is multiplied by 2. Note that $\frac{w-m}{n-m}$ is monotonically decreasing in m; thus, this quantity is maximized when m is the largest possible integer such that $\frac{w-m}{n-m} > 1/2$ is still true. Note that when $w < n/2$, then the optimal choice is $m = 0$. Assuming $w > n/2$ and solving for this inequality we obtain that $m = 2w - n$. Now the problem also has a physical constraint that $m \leq n/2$ since the adversary can choose at most $n/2$ slots. Thus there are three parameter regimes based on α:

1. $\alpha \leq n/2$: the optimal choice is $m = 0$

2. $n/2 \leq \alpha \leq 3n/4$: the optimal choice is $m = 2w - n$

3. $n > 3n/4$: the optimal choice is $m = n/2$

In case 1, the probability is then clearly bounded by $(1/2)^{n-1-w}$.

In case 2, making the substitution $m = 2w - n$ and $w = \alpha n$ where $\alpha \in [0, 1)$ in the expression (3.1), we obtain

$$\frac{\binom{2(1-\alpha)n}{(1-\alpha)n}}{\binom{n}{\alpha n}} 2^{(3\alpha-2)n} = \frac{[2(1-\alpha)n]!}{[(1-\alpha)n]![(1-\alpha)n]!} \frac{[\alpha n]![(1-\alpha)n]!}{n!} 2^{(3\alpha-2)n}$$

$$= \frac{[2(1-\alpha)n]![\alpha n]!}{[(1-\alpha)n]!n!} 2^{(3\alpha-2)n}$$

Recall that for all integers k the following is true by Sterling's formula:

$$\sqrt{2\pi}\sqrt{k}\left(\frac{k}{e}\right)^k \leq k! \leq e\sqrt{k}\left(\frac{k}{e}\right)^k$$

We can absorb the factors of $\sqrt{2\pi}$ and e in front into a small constant term less than 2. Note that since each factorial is a constant multiple of n, then the $\sqrt{k}$ term also yields a constant term, so we only need to compute the $(k/e)^k$ terms. This gives

$$\frac{[2(1-\alpha)n/e]^{2(1-\alpha)n}[\alpha n/e]^{\alpha n}}{[(1-\alpha)n/e]^{(1-\alpha)n}[n/e]^{n}}2^{(3\alpha-2)n} = \left(\frac{[2(1-\alpha)n/e]^{2(1-\alpha)}[\alpha n/e]^{\alpha}}{[(1-\alpha)n/e]^{(1-\alpha)}[n/e]^{1}}2^{(3\alpha-2)}\right)^{n}$$

We just need to show that the base is a constant bounded away from 1. Collecting terms in this, we obtain

$$\begin{aligned}
&2^{2(1-\alpha)}[1-\alpha]^{2(1-\alpha)}[n/e]^{2(1-\alpha)}\alpha^{\alpha}[n/e]^{\alpha}[1-\alpha]^{-(1-\alpha)}[n/e]^{-(1-\alpha)}[n/e]^{-1}2^{3\alpha-2}\\
&= [n/e]^{2(1-\alpha)n-(1-\alpha)+\alpha-1}[1-\alpha]^{2(1-\alpha)-(1-\alpha)}\alpha^{\alpha}2^{2(1-\alpha)+3\alpha-2}\\
&= (1-\alpha)^{1-\alpha}\alpha^{\alpha}2^{\alpha}
\end{aligned}$$

Taking $\log_2$ we obtain $(1-\alpha)\log_2(1-\alpha)+\alpha\log_2\alpha+\alpha \leq -0.0613$ when $\alpha \leq 3/4$, so the probability of success is bounded by $\frac{2}{2^{0.0613n}}$.

Finally in case 3, substituting $m = n/2$ in the expression (3.1) gives

$$\begin{aligned}
\frac{\binom{n/2}{(\alpha-1/2)n}}{\binom{n}{\alpha n}}2^{(\alpha-1/2)n} &= \frac{[n/2]!}{[(1-\alpha)n]![(\alpha-1/2)n]!}\frac{[\alpha n]![(1-\alpha)n]!}{n!}2^{(\alpha-1/2)n}\\
&= \frac{[n/2]![\alpha n]!}{n![(\alpha-1/2)n]!}2^{(\alpha-1/2)n}
\end{aligned}$$

Applying the Sterling approximation, we obtain

$$\frac{[n/e]^{n/2}2^{-n/2}[\alpha n/e]^{\alpha n}}{[n/e]^{n}[(\alpha-1/2)n/e]^{(\alpha-1/2)n}}2^{(\alpha-1/2)n} = \left(\frac{[n/e]^{1/2}2^{-1/2}[\alpha n/e]^{\alpha}}{[n/e]^{1}[(\alpha-1/2)n/e]^{(\alpha-1/2)}}2^{(\alpha-1/2)}\right)^{n}$$

The base of the exponent is

$$[n/e]^{1/2+\alpha-1-(\alpha-1/2)}[\alpha]^{\alpha}[\alpha-1/2]^{(1/2-\alpha)}2^{\alpha-1} = \alpha^{\alpha}(\alpha-1/2)^{1/2-\alpha}2^{\alpha-1}$$

Again taking $\log_2$ we obtain the condition $(1/2-\alpha)\log_2(\alpha-1/2)+\alpha\log\alpha+\alpha-1 < 0$, which is satisfied when $\alpha < 0.774$. This does not give much of an improvement over the previous constraint of $\alpha \leq 3/4$, so we state our final result just in that regime.

Apply a union bound of this probability over all $(Q+2n)^2$ pairs of symbolic polynomials to get the statement in the theorem.

□

Lemma 3.4.6. *For an adversary $\mathcal{A}$ in the generic group model which makes Q queries to the generic group oracle,*

$$|\,\mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{G}_M,\mathcal{O},\mathcal{A}}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C,1^n)) = P(C)] - \mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{G}_E\mathcal{O},\mathcal{A}}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C,1^n)) = P(C)] \leq \frac{1}{2^{0.0613n}}$$

Proof. Uses Lemmas 3.4.2 (recalling that the statement also holds for the pair $\mathcal{G}_M, \mathcal{G}_E$) and 3.4.5 plugged into the reduction from Lemma 3.4.1.

From Lemma 3.4.2 (recalling that the statement also holds for the pair $\mathcal{G}_M, \mathcal{G}_S$) we have that:

$$\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] = \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}) = 1]$$

as long as all queries to the generic group oracles are *identical* as defined in Definition 11.

Lemma 3.4.3 tells us that the probabilities of all queries not being identical during the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_E)$ is at most $\frac{2}{2^{0.0613n}}$.

Therefore, the difference $\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}) = 1] - \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$ is at most $\frac{2}{2^{0.0613n}}$, and so an adversary's advantage in the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_E)$ and $(\mathcal{G}_M, \mathcal{G}_M)$ is:

$$\begin{aligned}\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = \mathcal{G}_*] &= \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_E]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}) = 1] \\ &\quad + \mathbf{Pr}[\mathcal{G}_* = \mathcal{G}_M]\,\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 0] \\ &= \frac{1}{2} + \frac{1}{2}(\mathbf{Pr}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}) = 1] - \mathbf{Pr}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]) \\ &\leq \frac{1}{2} + \frac{2}{2^{0.0613n}}\end{aligned}$$

This, plugged into the reduction from Lemma 3.4.1, tells us that for all adversaries:

$$\left| Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}) = 1] \right| \leq \frac{1}{2^{0.0613n}}$$

□

Theorem 14. *The obfuscator for pattern matching with wildcards defined in Section 3.3 satisfies distributional VBB security for the ensemble of uniform distributions over* $\{0, 1\}^n$.

Proof. For any adversary $\mathcal{A}$ in the Distributional VBB game (in the generic group model), consider the following Simulator $\mathcal{S}$ which simply runs $\mathcal{A}$ on input produced by and interacted with like in Oracle *End* and outputs the same. Note that none of the behavior in Oracle *End* is dependent on the actual function $f_{\vec{y},\mathcal{W}}$ obfuscated. Therefore a simulator with no access to the function $f_{\vec{y},\mathcal{W}}$ drawn from the distribution is able to simulate $\mathcal{A}$ as described.

$\mathcal{S}$ then perfectly simulates the behavior of $\mathcal{A}$ interacting with oracle $\mathcal{O}_E$:

$$\mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{S}}[\mathcal{S}^C(1^{|C|}, 1^n) = P(C)] = \mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{G}_E,\mathcal{O},\mathcal{A}}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C, 1^n)) = P(C)]$$

From Lemma 3.4.6, we have that the difference in output probabilities between $\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})$ and

$\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})$ in the distributional VBB game in the generic group model is at most $\frac{1}{2^{0.0613n}}$:

$$|\,\mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C, 1^n)) = P(C)] - \mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C, 1^n)) = P(C)]| \leq \frac{1}{2^{0.0613n}}$$

From Lemma 3.4.4, we have that the difference in output probabilities between $\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})$ and $\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S})$ in the distributional VBB game in the generic group model is at most $\dfrac{(Q+2n)^2}{2^n}$:

$$|\,\mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C, 1^n)) = P(C)] - \mathbf{Pr}_C[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}(C, 1^n)) = P(C)]| \leq \frac{(Q+2n)^2}{2^n}$$

Now, recall that $\mathcal{G}_S$ faithfully instantiates $\mathcal{O}$ in the generic group model. Therefore, using the triangle inquality we have:

$$\begin{aligned}&|\,\mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{G},\mathcal{O},\mathcal{A}}[\mathcal{A}^{\mathcal{G}}(\mathcal{O}^{\mathcal{G}}(C, 1^n)) = P(C)] - \mathbf{Pr}_{C\leftarrow\mathcal{D}_n,\mathcal{S}}[\mathcal{S}^{C}(1^{|C|}, 1^n) = P(C)]| \\ &\leq \frac{(Q+2n)^2}{2^n} + \frac{1}{2^{0.0613n}} = negl(n)\end{aligned}$$

since the number of an adversary's generic group queries Q is a polynomial function of n and so $\mathcal{O}$ satisfies distributional VBB security in the generic group model.

□

Chapter 4: Non-statistical algorithms in machine learning

Modern machine learning is primarily based on the statistical convergence properties of stochastic gradient descent and its variations. However, empirically it has been observed that stochastic gradient descent exhibits optimization bias which is not reflected in the scalar loss objective function itself. This has led to research and discussion on whether sharpness or flatness of the Hessian near the solution is a desirable property for generalization [70]. In this chapter, we first investigate this bias mathematically using nonisotropic stochastic differential equations. We show that stochastic gradient descent has an optimization bias which arises from the nonconstant covariance term, resulting in convergence to a stationary point different from the local minima of the scalar objective function itself.

We then investigate alternative approaches to solving parameter recovery problems which are not rooted in statistical loss minimization. Our goal is to construct provable algorithms for nonconvex optimization problems which were previously solved using iterative methods such as EM. These techniques avoid issues with non-statistical optimization biases of the stochastic gradient descent algorithm. We demonstrate the use of Lenstra-Lenstra-Lovasz lattice basis reduction algorithm for solving two discrete parameter recovery problems. This work was previously published in [6] and [7] and demonstrates the value of cryptanalysis tools for solving certain types of machine learning problems.

4.1 Analysis of bias in stochastic optimization

Empirical studies have shown that the solutions returned by stochastic gradient descent seem to exhibit additional properties not explicitly present in the objective function itself. SGD seems to concentrate around minima where the spectrum of the Hessian is sparse [71], and thus the minima is "flat" in most directions. There is some evidence to suggest these flat minima lead to better generalization performance, and modifications of stochastic gradient descent designed to explicitly maximize flatness have been studied empirically [72].

[73] introduced a framework for modeling the stochastic gradient descent algorithm as a continuous-time stochastic differential equation. Following that there has been a steady stream of work trying to glean insight in the behavior of stochastic gradient descent from this diffusion framework [74][75]. However these papers make the simplifying assumption that the diffusion matrix is isotropic (a scalar multiple of the identity), and the behavior of the diffusion process with a nonconstant, nonisotropic matrix is far more interesting, albeit mathematically challenging.

The physics community has been trying to understand the stationary behavior of such processes for a long time, and recently there has been a line of work developing a new theory of A-type stochastic integration for computation of the implicit potential arising from a stochastic differential equation by a stationary analysis [76]. Recently [77] compares the Fokker-Planck equation derived by A-type stochastic integration with the Fokker-Planck equation arising from the usual Ito integral in order to obtain an explicit functional form for the difference between the implicit gradient and the function gradient. While their result is very general, it relies on the assumption that two Fokker-Planck equations with the same stationary distribution must be the same equation. Indeed the implication is only one way - given a FP-equation, its stationary distribution is uniquely determined, but conversely a stationary distribution does not uniquely determine a FP equation.

In our work, we seek a dynamical description of the stochastic gradient descent iterates rather than a stationary analysis. We start by giving a complete picture of the behavior in one dimension from a simple derivation from first principles and show that:

1. SGD can be rewritten as Langevin diffusion with respect to a different implicit potential function.

2. SGD with a large enough constant step size concentrates around the minima of the implicit potential function, which are different from the minima of the original objective function.

3. The implicit potential function obtained from this purely Ito integration perspective is not the same as that obtained from using results from A-type stochastic integration.

In contrast to the claim by [77], in this case the difference between the function minima and the implicit minim arises purely from the nonconstant diffusion term, since in one dimension the antisymmetric component is necessarily zero. Note that claim (1) goes beyond just a stationary analysis of the implicit potential function; we compute the exact form of the implicit gradient and the scaling in order for a Langevin diffusion process with unit covariance to track stochastic gradient descent.

The implicit potential function we derive is different from that described by [77]. Using the temperature $\beta = 2$, the implicit gradient we derive is $\frac{f'(\mathbf{z})+D'(\mathbf{z})/4}{\sqrt{D(\mathbf{z})}}$, whereas [77] obtains $\frac{f'(\mathbf{z})+D'(\mathbf{z})/2}{D(\mathbf{z})}$. Empirically we show on a toy example that Langevin diffusion with our implicit potential actually tracks stochastic gradient descent closely, whereas Langevin diffusion with the potential function of [77] exhibits very different behavior. The square root term in the denominator makes this potential function flatter with respect to the stochastic gradient noise.

Following the one dimensional picture, we point out some directions for future work to extend this analysis to the general multivariate case.

4.1.a Stochastic differential equations

Let $p(x)$ be a distribution over data points x. The stochastic gradient descent update with step size α can be written as

$$\mathbf{z}_{t+1} = \mathbf{z}_t - \alpha \mathop{\mathbf{E}}_{x\sim p}\left[\nabla\ell(x;\mathbf{z}_t)\right] + \alpha\left(\mathop{\mathbf{E}}_{x\sim p}\left[\nabla\ell(x;\mathbf{z}_t)\right] - \nabla\ell(\mathbf{x}_t;\mathbf{z}_t)\right)$$

where $\mathbf{x}_t \sim p$ is a sample drawn at time t. This is a Euler discretization of the following continuous-time diffusion process

$$d\mathbf{z} = A(\mathbf{z})\,dt + B(\mathbf{z})\,dw \tag{4.1}$$

where dw represents integration with respect to Brownian motion, and the quantities $A(\mathbf{z})$ and $B(\mathbf{z})$ are given by

$$\begin{aligned}
A(\mathbf{z}) &= -\alpha \mathop{\mathbf{E}}_{x\sim p}\left[\nabla f(x;\mathbf{z})\right] = -\alpha \nabla f(\mathbf{z}) \\
B(\mathbf{z})B(\mathbf{z})^T &= \mathop{\mathbf{E}}_{\mathbf{x}\sim p}\left[\alpha^2 \left(\mathop{\mathbf{E}}_{x\sim p}\left[\nabla \ell(x;\mathbf{z})\right] - \nabla \ell(\mathbf{x};\mathbf{z})\right)^{\otimes 2}\right] \\
&= \frac{\alpha^2}{N}\sum_{i=1}^{N} \nabla \ell(x_i;\mathbf{z})^{\otimes 2} - \alpha^2 \left(\frac{1}{N}\sum_{i=1}^{N} \nabla \ell(x_i;\mathbf{z})\right)^{\otimes 2} \qquad =: D(\mathbf{z})
\end{aligned}$$

where $D(\mathbf{z})$ is the empirical covariance matrix of the data points.

Lemma 4.1.1 (Fokker-Planck Equation)**.** *For any diffusion equation* $d\boldsymbol{z} = A(\boldsymbol{z})dt + B(\boldsymbol{z})d\boldsymbol{w}$*, the time evolution of the probability density* $\rho(\boldsymbol{z})$ *is given by the Fokker-Planck equation:*

$$\frac{\partial \rho(\boldsymbol{z})}{\partial t} = -\sum_{i=1}^{d} \frac{\partial}{\partial z_i}\left[A_i(\boldsymbol{z})\rho(\boldsymbol{z},t)\right] + \frac{1}{2}\sum_{i,j=1}^{d} \frac{\partial^2}{\partial z_i \partial z_j}\left[D_{ij}(\boldsymbol{z})\rho(\boldsymbol{z},t)\right] \tag{4.2}$$

The limiting stationary distribution is obtained by solving for the steady-state condition for $\rho(\mathbf{z},t)$:

$$0 = -\sum_{i=1}^{N} \frac{\partial}{\partial z_i}\left[A_i(\mathbf{z})\rho(\mathbf{z},t)\right] + \frac{1}{2}\sum_{i,j=1}^{N} \frac{\partial^2}{\partial z_i \partial z_j}\left[D_{ij}(\mathbf{z})\rho(\mathbf{z},t)\right]$$

An important result in transforming stochastic differential equations is the stochastic chain rule, also known as Ito's Lemma. Here we state the one-dimensional result where A, B are both scalar functions.

Lemma 4.1.2 (Ito's Lemma)**.** *Let $dz = A(z)dt + B(z)dw$ be a diffusion equation in one dimension. Let $g(z)$ be a twice-differentiable scalar function. Then the stochastic differential equation describing the evolution of $g(z)$ is given by*

$$dg = \left(\frac{\partial g}{\partial t} + A(z)\frac{\partial g}{\partial z} + \frac{B(z)^2}{2}\frac{\partial^2 g}{\partial z^2}\right) dt + B(z)\frac{\partial g}{\partial z}dw \tag{4.3}$$

The condition for the stationary distribution is

$$\begin{aligned} 0 = \frac{\partial \rho}{\partial t} &= \frac{\partial}{\partial \mathbf{z}}\left(\left[\frac{\partial f}{\partial \mathbf{z}}\rho(\mathbf{z})\right] + \frac{1}{2}\frac{\partial}{\partial \mathbf{z}}\left[D(\mathbf{z})\rho(\mathbf{z})\right]\right) \\ c &= \left[\frac{\partial f}{\partial \mathbf{z}}\rho(\mathbf{z}, t)\right] + \frac{1}{2}\frac{\partial}{\partial \mathbf{z}}\left[D(\mathbf{z})\rho(\mathbf{z})\right] =: S(\mathbf{z}) \end{aligned}$$

for some constant c, where $S(\mathbf{z})$ is a quantity called the probability current that must be constant in one dimension. Because the usual parameter domain in machine learning is unbounded, then this means the probability current must be exactly 0 in order for the distribution to be normalizable. If the normalization $D(\mathbf{z}) = 1$ has already been done, then this reduces to the Gibbs distribution:

$$\begin{aligned} 0 &= \frac{\partial f}{\partial \mathbf{z}}\rho(\mathbf{z}, t) + \frac{1}{2}\frac{\partial \rho}{\partial \mathbf{z}} \\ \rho(\mathbf{z}) &= \exp\left(-2f(\mathbf{z})\right) \end{aligned}$$

4.1.b Analysis for one-dimensional SGD

In this section we will characterize the effect of the nonhomogeneous term $D(\mathbf{z})$ on both the infinitesimal dynamics as well as the limiting stationary distribution.

The diffusion scalar in one dimension is particularly simple - it just modulates the magnitude of the random noise term arising from Brownian motion. If $D(\mathbf{z}) = D$ were constant, we could just divide it out of the entire diffusion equation, normalizing it to obtain

$$d\tilde{\mathbf{z}} = -\frac{A(\tilde{\mathbf{z}})}{\sqrt{D}} dt + dw$$

This describes a stochastic differential equation that is essentially slowed down by a factor of $\sqrt{D}$ from the original equation. The distribution of sample paths of $\tilde{\mathbf{z}}$ is exactly the same as that of $\mathbf{z}$, only the speed at which the diffusion travels along the path is changed.

To understand the behavior of stochastic gradient descent with nonhomogeneous variance, we will apply the same normalization idea to transform the stochastic differential equation (4.1) into one where $B(\tilde{\mathbf{z}}) = 1$ is constant.

As done in [78], define the change of variables

$$\tilde{\mathbf{z}} = g(\mathbf{z}) = \int_{z_0}^{\mathbf{z}} \frac{dz}{\sqrt{D(z)}} \tag{4.4}$$

Note that $d\tilde{\mathbf{z}}$ has the same sign as $d\mathbf{z}$ since the scaling term $1/\sqrt{D(\mathbf{z})}$ is always positive. Thus the path taken by $\tilde{\mathbf{z}}$ is a scaled path taken by $\mathbf{z}$.

By Ito's Lemma (4.3) the evolution of $\tilde{\mathbf{z}}$ is given by

$$\begin{aligned} d\tilde{\mathbf{z}} &= \left(\frac{\partial g}{\partial t} + \frac{\partial A}{\partial z}\frac{\partial g}{\partial z} + \frac{B(\mathbf{z})^2}{2}\frac{\partial^2 g}{\partial z^2} \right) dt + B(\mathbf{z})\frac{\partial g}{\partial z} dw \\ &= \left(\frac{\partial A}{\partial z}\frac{1}{\sqrt{D(\mathbf{z})}} + \frac{D(\mathbf{z})}{2}\frac{\partial}{\partial z}\left[\frac{1}{\sqrt{D(\mathbf{z})}} \right] \right) dt + B(\mathbf{z})\frac{1}{\sqrt{D(\mathbf{z})}} dw \\ &= \frac{1}{\sqrt{D(\mathbf{z})}}\frac{\partial}{\partial z}\left[A(\mathbf{z}) - \frac{1}{4}D(\mathbf{z}) \right] (\mathbf{z}) dt + dw \end{aligned}$$

This gives an equivalent Langevin equation with an isotropic noise term. We now substitute the description of stochastic gradient descent into the functions $A(\mathbf{z}), B(\mathbf{z})$ to obtain

$$d\tilde{\mathbf{z}} = -\frac{1}{\sqrt{\widehat{D}(\mathbf{z})}} \frac{\partial}{\partial \mathbf{z}} \left[f(\mathbf{z}) + \frac{\alpha}{4} \widehat{D}(\mathbf{z}) \right] dt + dw \tag{4.5}$$

We immediately notice that the dynamics are not gradient descent of f, but rather look more like gradient descent with respect to the function $f(\mathbf{z}) + \alpha \widehat{D}(\mathbf{z})/4$ with step size $1/\sqrt{\widehat{D}(\mathbf{z})}$. In one dimension we can write this diffusion term as

$$\widehat{D}(\mathbf{z}) = \frac{1}{N} \sum_{i=1}^{N} f'(x_i; \mathbf{z})^2 - f'(\mathbf{z})^2$$
$$\widehat{D}'(\mathbf{z}) = \frac{2}{N} \sum_{i=1}^{N} f'(x_i; \mathbf{z}) f''(x_i; \mathbf{z}) - 2 f'(\mathbf{z}) f''(\mathbf{z})$$

This suggests that stochastic gradient descent is actually implicitly a second-order optimization algorithm whose gradient steps are of the form

$$\Delta \mathbf{z} = \frac{1}{\sqrt{D(\mathbf{z})}} \left(-f'(\mathbf{z}) + \frac{\alpha}{2} f'(\mathbf{z}) f''(\mathbf{z}) - \frac{2\alpha}{N} \sum_{i=1}^{N} f'(x_i; \mathbf{z}) f''(x_i; \mathbf{z}) \right) \tag{4.6}$$

We can write the implicit potential function of stochastic gradient descent as a formal path integral of the gradient

$$\Phi(\mathbf{z}) = \int_{z_0}^{\mathbf{z}} \frac{f'(v) + \alpha \widehat{D}'(v)/4}{\sqrt{\widehat{D}(v)}} \, dv$$

where the choice of z_0 does not matter as long as $D(\mathbf{z})$ does not vanish anywhere in the parameter space. Note that in the $d = 1$ case, this condition holds as long as there are more than 3 distinct data points and the function f has no singularities (which is usually the case).

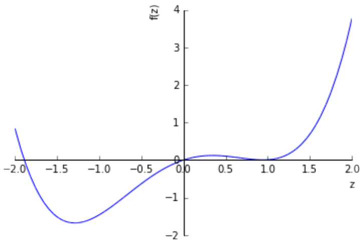

Figure 4.1: Function value

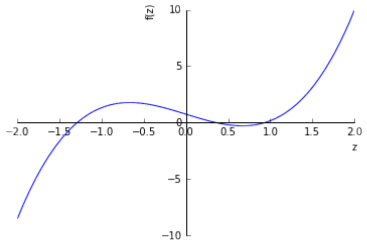

Figure 4.2: Function gradient

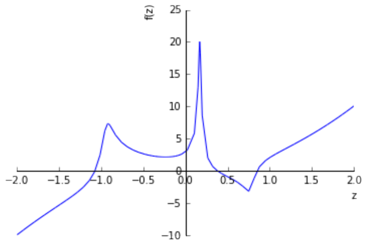

Figure 4.3: Implicit gradient

This suggests that stochastic gradient descent with a fixed step size of α does not actually concentrate around the true minima of the function, but rather around a minima of a different implicit function. Note that there is no contradiction with classical convergence results of stochastic optimization, since those require a decaying step size which satisfies $\sum_{t=1}^{\infty} \frac{1}{\alpha_t^2} < \infty$.

4.1.c Empirical observations of the implicit gradient

We consider the function $f(x; \mathbf{z}) = (\mathbf{zx} - 1)^2(\mathbf{zx} + 1)^2 + (\mathbf{zx} - 1)$ with three data points $x \in \{0.5, 0.7, 1\}$. This function has a local minima at approximately 0.94 and a global minima at approximately -1.29 and is plotted in Figure (4.1).

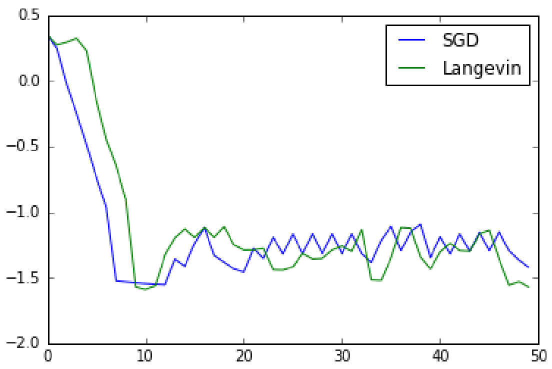

Figure 4.4: Trajectories taken by scaled SGD and the equivalent normalized Langevin diffusion

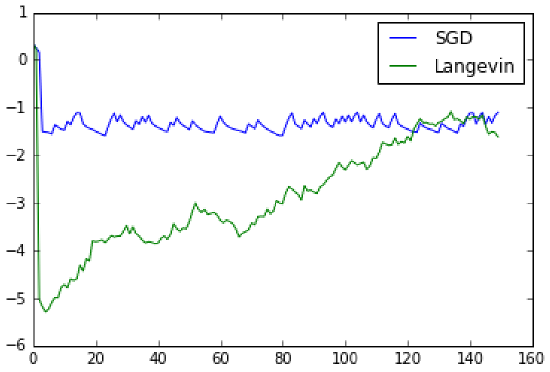

Figure 4.5: Trajectory taken by Langevin diffusion obtained from A-type stochastic integration

We show some empirical evidence that the implicit update rule derived in (4.6) tracks the actual trajectory taken by SGD. We initialize both algorithms at the local maxima at $z \approx 0.3475$ and run both algorithms with step size $\alpha = 0.1$. The updates of SGD are scaled by $1/\sqrt{D(\mathbf{z})}$

according to the scaling derived in (4.4). Figure (4.4) shows that the trajectory of the equivalent Langevin diffusion update closely tracks the trajectory of stochastic gradient descent. In contrast, Figure (4.5) shows that the implicit potential function obtained from A-type stochastic integration eventuallys converge to the same location but has very different initial behavior.

The learning rate $\alpha = 0.1$ for these examples is chosen to be as large as possible while still maintaining stability of the learning problem, and this is in line with the usual way learning rates are chosen.

4.1.d Characterization of the limiting stationary distribution

The formal stationary distribution can be computed by using the implicit potential function in the Gibbs distribution:

$$\rho_{ss}(\mathbf{z}) \propto \exp\left(-2\int_{z_0}^{\mathbf{z}} \frac{f'(v) + \widehat{D}'(v)/4}{\sqrt{\widehat{D}(v)}}\,dv\right)$$

To better understand the behavior of this potential, we can also analyze the stationary condition of the non-normalized form of the 1-dimensional steady-state Fokker-Planck equation (4.2):

$$0 = \frac{\partial}{\partial \mathbf{z}}\left[\frac{\partial f}{\partial \mathbf{z}}\rho(\mathbf{z},t)\right] + \frac{\alpha}{2}\frac{\partial^2}{\partial \mathbf{z}^2}\left[\widehat{D}(\mathbf{z})\frac{\partial \rho}{\partial \mathbf{z}}\right]$$

The 1D Fokker-Planck equation becomes the following homogeneous second-order differential equation:

$$\frac{\partial \rho}{\partial t} = \alpha L \rho(\mathbf{z})$$

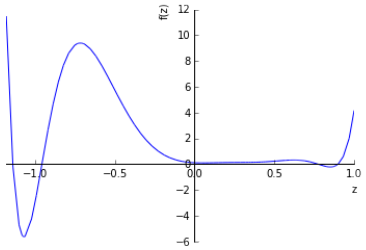

Figure 4.6: Discriminant

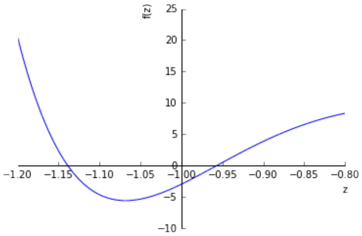

Figure 4.7: At global minimum

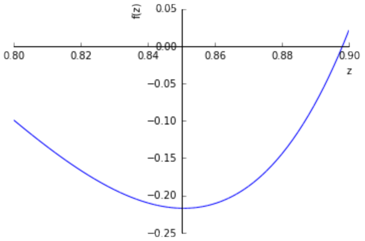

Figure 4.8: At local minimum

where

$$L = \frac{\alpha}{2}\widehat{D}(\mathbf{z})\frac{\partial^2}{\partial \mathbf{z}^2} + \left(f'(\mathbf{z}) + \alpha\widehat{D}'(\mathbf{z})\right)\frac{\partial}{\partial \mathbf{z}} + \left(f''(\mathbf{z}) + \frac{1}{2}\alpha\widehat{D}''(\mathbf{z})\right)$$

thus the characteristic polynomial of this differential equation is

$$P(\lambda) = \frac{\alpha}{2}\widehat{D}(\mathbf{z})\lambda^2 + \left(f'(\mathbf{z}) + \alpha\widehat{D}'(\mathbf{z})\right)\lambda + \left(f''(\mathbf{z}) + \frac{1}{2}\alpha\widehat{D}''(\mathbf{z})\right)$$

The roots of the characteristic polynomial are given by the quadratic formula

$$\lambda = \frac{-f'(\mathbf{z}) - \alpha\widehat{D}'(\mathbf{z}) \pm \sqrt{\left(f'(\mathbf{z}) + \alpha\widehat{D}'(\mathbf{z})\right)^2 - 2\alpha\widehat{D}(\mathbf{z})\left(f''(\mathbf{z}) + \frac{\alpha}{2}\widehat{D}''(\mathbf{z})\right)}}{\alpha^2\widehat{D}(\mathbf{z})^2}$$

The sign of the discriminant

$$\left(f'(\mathbf{z}) + \alpha\widehat{D}'(\mathbf{z})\right)^2 - 2\alpha\widehat{D}(\mathbf{z})\left(f''(\mathbf{z}) + \frac{\alpha}{2}\widehat{D}''(\mathbf{z})\right) \tag{4.7}$$

characterizes the solution to the stationary condition. A positive discriminant corresponds to a real root λ which yields the Gibbs distribution. A negative discriminant corresponds to a complex root λ which introduces sinusoidal terms.

As seen from Figure (4.6), the discriminant is positive throughout most of the parameter space and is positive at the function's actual local minima. However, at both of the implicit local minima (where the implicit gradient equals zero), the discriminant is actually negative.

Thus the stationary distribution of stochastic gradient descent around the minima of the function that it implicitly minimizes is not a pure Gibbs distribution. The roots λ_1, λ_2 are complex conjugate pairs $\nu(\mathbf{z}) \pm i\mu(\mathbf{z})$ for real-valued functions ν, μ, so the stationary distribution can be written a combination of the two fundamental solutions $e^{\nu(zt)}\cos(\mu(zt))$ and $e^{\nu(zt)}\sin(\mu(zt))$.

Next we compare the stationary distributions achieved by stochastic gradient descent and the equivalent normalized Langevin diffusion. We run both algorithms over a cover of 2000 samples initialized uniformly in the interval $[-2, 2]$ at intervals of 0.001. The same step size of $\alpha = 0.1$ is used, but this time we do not scale the updates of SGD, because we are only interested in the stationary distribution. The stationary distributions are plotted around the global minimum at ≈ 1.34.

Figure (4.9) shows the stationary distribution of stochastic gradient descent algorithm. Figure (4.10) shows the stationary distribution of the normalized Langevin diffusion equation derived from the stochastic gradient descent update. Note that both distributions are skewed towards the right of the true global minima.

Figure (4.11) shows the stationary distribution of the unnormalized Langevin diffusion equation, which is just the constant D approximation studied in previous results. Here independent, identical Gaussian noise is added to each full gradient descent step, so as expected the distribution looks like a Gaussian around the function's true global minimum. Note the stark contrast between the stationary behavior of SGD from this distribution.

Using instead a very small step size (e.g. $\alpha = 0.01$) causes the stationary distribution of both stochastic gradient descent and normalized Langevin diffusion to look exactly like Figure (4.11). The concentration around the implicit potential function's minima is exclusively a large step size phenomenon.

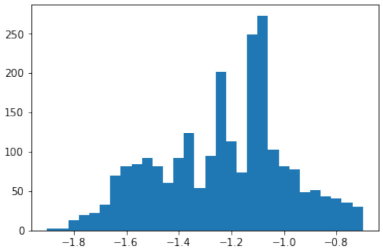

Figure 4.9: Stochastic gradient descent

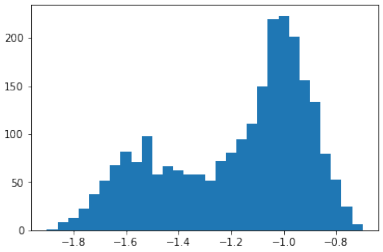

Figure 4.10: Normalized Langevin diffusion

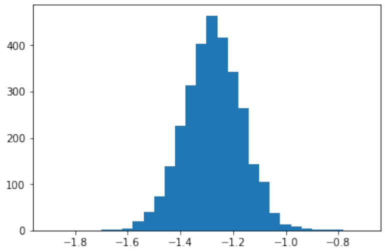

Figure 4.11: Unnormalized Langevin diffusion

4.2 The correspondence retrieval problem

In (the real-variant of) the phase retrieval problem, an unknown vector $\boldsymbol{x} \in \mathbb{R}^d$ is to be recovered, up to sign, from magnitudes of projections $\left|\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle\right|$ onto n known measurement vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n \in \mathbb{R}^d$. The phase retrieval problem has a rich history in several engineering and scientific domains, especially when the $\boldsymbol{w}_i$ are Fourier basis vectors [see, e.g., 79, 80, for an overview]. The setting where the $\boldsymbol{w}_i$ are independent draws from certain probability distributions has been intensely studied in the past several years. Many algorithms based on numerical optimization (e.g., semidefinite programming, local optimization of convex and non-convex objectives) have been proven to solve the problem with high probability when provided enough measurements [81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96].

In this paper, we consider a generalization of phase retrieval, which we call *correspondence retrieval*: a set of k distinct but unknown points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k \in \mathbb{R}^d$ are to be recovered from the unordered collection of projection values $\langle \boldsymbol{w}_i, \boldsymbol{x}_1\rangle, \langle \boldsymbol{w}_i, \boldsymbol{x}_2\rangle, \ldots, \langle \boldsymbol{w}_i, \boldsymbol{x}_k\rangle$ onto n known measurement vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n$. Importantly, the correspondence of the k projections $\{\langle \boldsymbol{w}_i, \boldsymbol{x}_j\rangle\}_{j=1}^k$ across different measurements is unknown. Phase retrieval, as described above, is the special case where $k = 2$ and $\boldsymbol{x}_1 = -\boldsymbol{x}_2$; the two real numbers corresponding to each measurement are additive inverses ($\langle \boldsymbol{w}_i, \boldsymbol{x}_1\rangle$ and $\langle \boldsymbol{w}_i, \boldsymbol{x}_2\rangle = -\langle \boldsymbol{w}_i, \boldsymbol{x}_1\rangle$).

We propose an algorithm for the case of independent standard Gaussian measurement vectors. For general k, the algorithm correctly recovers the unknown points with high probability from

$n = d + 1$ measurements, assuming that the points are linearly independent. For the phase retrieval setting, a variant of the algorithm has the same guarantee without the linear independence assumption. Our algorithms are based on reductions to the Shortest Vector Problem [97] on certain random lattices; we prove that vectors provided by the Lenstra-Lenstra-Lovász basis reduction algorithm [henceforth LLL; 98] yield to the correct solution for the correspondence retrieval problem. Our reduction generalizes an algorithm of [99] for solving random instances of the Subset Sum Problem [100, pg. 223]. We note that [101] establish the hardness of the phase retrieval via reduction *from* the Subset Sum Problem. Our algorithmic result can be viewed as a reduction in the other direction.

In the phase retrieval setting, our results show a gap between the number of measurement vectors required for all vectors $\boldsymbol{x} \in \mathbb{R}^d$ to be recoverable, and the number of random measurements sufficient for any particular vector to be recoverable. This is the same distinction between the "for all" and "for each" guarantees studied in the context of compressive sensing [102]. [103] prove that $n = 2d - 1$ measurement vectors are necessary for the "for all" guarantee, and also that the same number of typical measurement vectors are sufficient. Previous algorithmic results for phase retrieval require $n \geq Cd$ for some sufficiently large constant $C \geq 2$ or even $n \geq d\operatorname{poly}\log(d)$. Our algorithmic result has the "for each" guarantee: the $n = d+1$ measurements suffice with high probability for the particular unknown vector of interest. Note that in the general correspondence retrieval problem, each measurement is comprised of k unordered real numbers, so the sufficiency of $d + 1$ measurements even when $k > 2$ is sensible.

We also describe an algorithm that works even when the measurements are corrupted by additive mean-zero Gaussian noise.[1] The algorithm is essentially the same as one proposed by [104] for the related parameter estimation problem in the mixtures of linear regressions model; the main technique used is the method-of-moments and orthogonal tensor decomposition [105]. We observe that the moments used in the algorithm are invariant to the noise variance, and hence the algorithm is noise-robust in this sense. However, the number of measurements required by this

[1] In phase retrieval, noise is typically added to the square (magnitude) $\left|\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle\right|^2$ of the projections [86, 82]. In our setting, independent noise is added to the k projections $\{\langle \boldsymbol{w}_i, \boldsymbol{x}_j\rangle\}_{j=1}^k$ themselves.

algorithm, even when the noise is absent, is larger than that of the lattice-based algorithm. The moment-based algorithm appears to ignore consistency constraints across measurements that the lattice-based algorithm is able to exploit.

4.2.a Setting and notations

This section describes the correspondence retrieval problem, notations and results concerning lattices and tensors, and the non-degeneracy condition required by the proposed algorithms.

Correspondence retrieval problem

In an instance of the correspondence retrieval problem, k distinct but unknown points in $\mathbb{R}^d$, denoted by $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k \in \mathbb{R}^d$, are revealed through collections of noisy linear measurements.

The n measurement vectors, denoted by $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n$, are i.i.d. random vectors in $\mathbb{R}^d$ with the standard multivariate Gaussian distribution $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_{d\times d})$. For each $i \in \{1, 2, \ldots, n\}$, the i-th measurement is the unordered (multi-)set of k (Euclidean) inner products between $\boldsymbol{w}_i$ and the k points, corrupted by additive zero-mean Gaussian noise with variance σ^2:

$$\mathcal{M}_i^\sigma \; := \; \{\langle \boldsymbol{w}_i, \boldsymbol{x}_1\rangle + \sigma\varepsilon_{i,1}, \langle \boldsymbol{w}_i, \boldsymbol{x}_2\rangle + \sigma\varepsilon_{i,2}, \ldots, \langle \boldsymbol{w}_i, \boldsymbol{x}_k\rangle + \sigma\varepsilon_{i,k}\},$$

where the $\{\varepsilon_{i,j}\}_{1\le i\le n, 1\le j\le k}$ are i.i.d. $\mathrm{N}(0,1)$ random variables. The noiseless version of the problem has $\sigma^2 = 0$, and the measurements are denoted by $\mathcal{M}_i := \mathcal{M}_i^0$ for $i \in \{1, 2, \ldots, n\}$.

The goal is to (approximately) reconstruct the set of k unknown points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$ (i.e., reconstruct up to reordering), from the data $(\boldsymbol{w}_1, \mathcal{M}_1^\sigma), (\boldsymbol{w}_2, \mathcal{M}_2^\sigma), \ldots, (\boldsymbol{w}_n, \mathcal{M}_n^\sigma)$.

Notations

The first m positive integers are denoted by $[m] := \{1, 2, \ldots, m\}$. The Euclidean inner product between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is denoted by $\langle \boldsymbol{u}, \boldsymbol{v}\rangle$, and the Euclidean norm is $\|\boldsymbol{v}\|_2 := \sqrt{\langle \boldsymbol{v}, \boldsymbol{v}\rangle}$. The i-th largest singular value of a matrix $\boldsymbol{M}$ is denoted by $\sigma_i(\boldsymbol{M})$; the spectral norm (i.e., largest singular value) is also denoted by $\|\boldsymbol{M}\|_2$.

Lattices

An ordered basis $\boldsymbol{B} = [\boldsymbol{b}_1|\boldsymbol{b}_2|\cdots|\boldsymbol{b}_n] \in \mathbb{R}^{m\times n}$, arranged as columns in a rank n matrix, generates a lattice

$$\Lambda(\boldsymbol{B}) := \left\{ \sum_{i=1}^{n} z_i \boldsymbol{b}_i : z_1, z_2, \ldots, z_r \in \mathbb{Z} \right\} \subset \mathbb{R}^m ,$$

where $\mathbb{Z}$ denotes the set of integers. The Shortest Vector Problem is to find the shortest non-zero vector in the lattice:

$$\operatorname*{arg\,min}_{\boldsymbol{v}\in\Lambda(\boldsymbol{B})\setminus\{\boldsymbol{0}\}} \|\boldsymbol{v}\|_2 .$$

The length of the shortest vector is denoted by $\lambda(\boldsymbol{B})$.

Current techniques for this problem involve "reducing" the input basis $\boldsymbol{B}$ so that it is at least somewhat well-conditioned in a certain sense. [98] show that the first vector $\boldsymbol{b}_1$ in a suitably reduced basis $\boldsymbol{B}$ has length at most $2^{(n-1)/2} \cdot \lambda(\boldsymbol{B})$. They also give an algorithm (LLL) that, given a basis $\boldsymbol{B} \in \mathbb{Z}^{m\times n}$ with integer coefficients, computes a reduced basis $\boldsymbol{B}'$ with $\Lambda(\boldsymbol{B}') = \Lambda(\boldsymbol{B})$ in time polynomial in m, n, and $\log(\|\boldsymbol{B}\|_\infty)$, where $\|\boldsymbol{B}\|_\infty$ denotes the magnitude of the largest entry in $\boldsymbol{B}$. In this sense, LLL is a $2^{(n-1)/2}$-approximation algorithm for the Shortest Vector Problem.

An important concern with the use of LLL on bases with real-valued coefficients is numerical precision. There are two cases where precision needs to be considered: precision in the measurements, and precision in the internal arithmetic operations in LLL. We discuss these issues in Appendix 4.2.f. To simplify the foregoing discussion, we assume that LLL may be run on input bases with real-valued coefficients.

Tensors

For a positive integer p, a real order-p tensor $\boldsymbol{T} \in \bigotimes_{i=1}^{p} \mathbb{R}^n$ is a p-linear function $\boldsymbol{T} \colon \mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n \to \mathbb{R}$. We only require tensors of order two (i.e., matrices) and order three. The rank-one tensor $\boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \cdots \otimes \boldsymbol{v}_p$, for vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_p \in \mathbb{R}^n$, is the p-linear function satisfying

$$(\boldsymbol{v}_1 \otimes \boldsymbol{v}_2 \otimes \cdots \otimes \boldsymbol{v}_p)(\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_p) \;=\; \langle \boldsymbol{v}_1, \boldsymbol{u}_1 \rangle \langle \boldsymbol{v}_2, \boldsymbol{u}_2 \rangle \cdots \langle \boldsymbol{v}_p, \boldsymbol{u}_p \rangle \,, \quad \boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_p \in \mathbb{R}^n \,.$$

We use the shorthand notation $\boldsymbol{v}^{\otimes p}$ for $\boldsymbol{v} \in \mathbb{R}^n$ to denote the (symmetric) rank-one tensor $\boldsymbol{v} \otimes \boldsymbol{v} \otimes \cdots \otimes \boldsymbol{v} \in \bigotimes_{j=1}^{p} \mathbb{R}^n$. For $p = 2$, this is the symmetric outer product of a vector: $\boldsymbol{v}^{\otimes 2} = \boldsymbol{v}\boldsymbol{v}^\top$. We may also identify a tensor $\boldsymbol{T} \in \bigotimes_{i=1}^{p} \mathbb{R}^n$ with a multi-index array of n^p real numbers; the $(i_1, i_2, \ldots, i_p)$-th entry is $\boldsymbol{T}(\boldsymbol{e}_{i_1}, \boldsymbol{e}_{i_2}, \ldots, \boldsymbol{e}_{i_p})$, where $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_n$ are the standard coordinate basis vectors for $\mathbb{R}^n$.

Non-degeneracy conditions

Arrange the k unknown points in the matrix $\boldsymbol{X} := [\boldsymbol{x}_1 | \boldsymbol{x}_2 | \cdots | \boldsymbol{x}_k] \in \mathbb{R}^{d \times k}$. Our main algorithms require $\boldsymbol{X}$ to have $\operatorname{rank}(\boldsymbol{X}) = k$—i.e., the points must be linearly independent.

We measure how ill-conditioned $\boldsymbol{X}$ is in two ways. The first is based on the singular values $\sigma_1(\boldsymbol{X}) \geq \sigma_2(\boldsymbol{X}) \geq \cdots \geq \sigma_k(\boldsymbol{X})$ of $\boldsymbol{X}$, primarily through the ratio $\kappa(\boldsymbol{X}) := \sigma_1(\boldsymbol{X}) / \sigma_k(\boldsymbol{X})$. The second is $\lambda(\boldsymbol{X})$, the length of the shortest non-zero vector in the lattice $\Lambda(\boldsymbol{X})$. The quantities $\kappa(\boldsymbol{X})$ and $\lambda(\boldsymbol{X})$ are related in the following proposition, which is proved in Appendix 4.2.h.

Proposition 4.2.1. $\lambda(\boldsymbol{X}) \geq \min_{i \in [k]} \|\boldsymbol{x}_i\|_2 \cdot 2\kappa(\boldsymbol{X}) / (\kappa(\boldsymbol{X})^2 + 1)$.

For $k = 2$ (the phase retrieval setting), a variant of our lattice-based algorithm requires $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$, but permits the points to be linearly dependent.

4.2.b Noiseless correspondence retrieval

This section describes lattice-based algorithms for the noiseless correspondence retrieval problem.

Algorithm description

Our main algorithm, specified in Algorithm 1, is based on reductions to the Shortest Vector Problem in lattices. Using information from $d+1$ measurements and the input parameter $\beta > 0$, the algorithm constructs k lattice bases with the following properties. First, for each $t \in [k]$, the only short vectors in the t-th lattice reveal which elements in the first d measurements correspond to the unknown vector $\boldsymbol{x}_t$. Second, when β is sufficiently large, all other vectors in the lattices are longer by exponentially-large factors. This lattice construction is based on the algorithm of [99] for solving random instances of the Subset Sum Problem via reduction to the Shortest Vector Problem. Our algorithm similarly approximately solves these Shortest Vector Problem instances using LLL to obtain the correspondence information, and then recovers all of the k unknown points by solving systems of linear equations from the first d measurements.

Main result and analysis

The main performance guarantee for Algorithm 1 is given in Theorem 20 below.

Theorem 15. *Assume* $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_k] \in \mathbb{R}^{d\times k}$ *has* $\operatorname{rank}(\boldsymbol{X}) = k$*. For any* $\delta \in (0,1)$*, if*

$$\beta \;\geq\; \frac{16 \cdot \left(2 \cdot 2^{dk/2} \cdot \sqrt{d+1} + 1\right)^{dk+1} \cdot 2^{dk/2} \cdot d \cdot \sqrt{d+1} \cdot \left(2\sqrt{d} + \sqrt{2\ln(8/\delta)}\right) \cdot k^2}{\pi \cdot \delta^2 \cdot \lambda(\boldsymbol{X})},$$

then with probability at least $1-\delta$*, Algorithm 1 returns* $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$*.*

Numerical issues and running time are discussed in Appendix 4.2.f. The rest of this subsubsection is devoted to the proof of Theorem 20.

Let $R := 2^{dk/2} \cdot \sqrt{d+1}$, and let $\mathcal{Z}_R := \{(z_0, \boldsymbol{z}) \in \mathbb{Z} \times \mathbb{Z}^{dk} : 0 < z_0^2 + \|\boldsymbol{z}\|_2^2 \leq R^2\}$. For $\delta \in (0,1)$, define

$$r_\delta \;:=\; \sqrt{\frac{\pi}{2} \cdot \left(\frac{\delta}{k|\mathcal{Z}_R|}\right)^2 \cdot \frac{\lambda(\boldsymbol{X})^2 \cdot \frac{\pi}{2} \cdot \left(\frac{\delta}{2dk|\mathcal{Z}_R|}\right)^2}{\left(2\sqrt{d} + \sqrt{2\ln(4/\delta)}\right)^2}}\,.$$

Algorithm 1 Lattice-based algorithm for noiseless correspondence retrieval

```
input  Data $(\boldsymbol{w}_i, \mathcal{M}_i)$ for $i \in [d+1]$, parameter $\beta > 0$.
output  Set of points $\{\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{x}}_2, \dots, \widehat{\boldsymbol{x}}_k\}$, or “failure”.
if $\boldsymbol{W} := [\boldsymbol{w}_1|\boldsymbol{w}_2|\cdots|\boldsymbol{w}_d]^\top$ is singular then
    return “failure”.
end if
Let $y_{i,1}, y_{i,2}, \dots, y_{i,k}$ be an arbitrary ordering the elements of $\mathcal{M}_i$, for each $i \in [d+1]$.
Define $\boldsymbol{a} = (a_{i,j} : i \in [d], j \in [k]) \in \mathbb{R}^{dk}$ by
    $$a_{i,j} \;:=\; \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle y_{i,j} \,,$$
    where $\tilde{\boldsymbol{w}}_i$ is the $i$-th column of $\boldsymbol{W}^{-1}$.
for $t = 1, 2, \dots, k$ do
    Construct basis
        $$\boldsymbol{B}^{(t)} \;=\; \begin{bmatrix} \boldsymbol{b}_0^{(t)} & \boldsymbol{b}_{1,1}^{(t)} & \cdots & \boldsymbol{b}_{d,k}^{(t)} \end{bmatrix} \;:=\; \left[\begin{array}{c|c} \multicolumn{2}{c}{\boldsymbol{I}_{dk+1}} \\ \hline \beta y_{d+1,t} & -\beta \boldsymbol{a}^\top \end{array}\right] \;\in\; \mathbb{R}^{(dk+2)\times(dk+1)} \,.$$
    Let $L^{(t)}(\widehat{z}_0, \widehat{\boldsymbol{z}}) := \widehat{z}_0 \boldsymbol{b}_0^{(t)} + \sum_{i,j} \widehat{z}_{i,j} \boldsymbol{b}_{i,j}^{(t)} \in \Lambda(\boldsymbol{B}^{(t)})$ for $(\widehat{z}_0, \widehat{\boldsymbol{z}}) \in \mathbb{Z} \times \mathbb{Z}^{dk}$ be the vector
        returned by LLL as an approximate solution to Shortest Vector Problem for $\Lambda(\boldsymbol{B}^{(t)})$.
    if the $(dk+2)$-th coordinate of $L^{(t)}(\widehat{z}_0, \widehat{\boldsymbol{z}})$ is non-zero then
        return “failure”.
    end if
    Let $\widehat{\boldsymbol{x}}_t$ be a solution to the system of linear equations (in $\boldsymbol{x} \in \mathbb{R}^d$)
        $$\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle \;=\; y_{i,j} \,, \qquad (i,j) \in [d] \times [k] \,\mathbf{.}\, \widehat{z}_{i,j} \neq 0 \,,$$
        or $\mathbf{0}$ if no solution exists.
end for
return $\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{x}}_2, \dots, \widehat{\boldsymbol{x}}_k$.
```

The coefficient vectors in $\mathcal{Z}_R$ include all those that could potentially determine lattice vectors in $\Lambda(\boldsymbol{B}^{(t)})$ for $t \in [k]$ with length at most R. Below, we prove that these lattice vectors either provide the correspondence information needed to recover the unknown points (and have length $\ll R$), or they have length more than βr_δ with high probability. A crude bound on the cardinality of $\mathcal{Z}_R$ is

$$|\mathcal{Z}_R| \;\le\; \left|\{-\lfloor R \rfloor, -\lfloor R \rfloor + 1, \dots, \lfloor R \rfloor - 1, \lfloor R \rfloor\}\right|^{dk+1} \;\le\; \left(2 \cdot 2^{(dk+1)/2} \cdot \sqrt{d+1} + 1\right)^{dk+1} \,.$$

For each $i \in [d]$, let $\pi_i \colon [k] \to [k]$ denote the (unknown) permutation on $[k]$ that determines the arbitrary ordering of $\mathcal{M}_i$ from Algorithm 1:

$$y_{i,j} \; = \; \langle \boldsymbol{w}_i, \boldsymbol{x}_{\pi_i(j)} \rangle \,, \quad i \in [d], j \in [k] \,.$$

Also, for $\delta \in (0,1)$, let $\mathcal{E}_\delta$ be the event that

1. the smallest singular value of $\boldsymbol{W}$ is bounded from below: $\sigma_d(\boldsymbol{W}) \geq \delta/(4\sqrt{d})$;
2. the spectral norm of $\boldsymbol{W}$ is bounded from above: $\|\boldsymbol{W}\|_2 \leq 2\sqrt{d} + \sqrt{2\ln(4/\delta)}$;
3. for each $i \in [d]$, $j \in [k]$, and $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$ such that $|z_{i,j} - z_0| + \sum_{j' \neq j} |z_{i,j'}| > 0$,

$$\left\langle \boldsymbol{w}_i, (z_{i,j} - z_0)\boldsymbol{x}_{\pi_i(j)} + \sum_{j' \neq j} z_{i,j'} \boldsymbol{x}_{\pi_i(j')} \right\rangle^2 \; \geq \; \lambda(\boldsymbol{X})^2 \cdot \frac{\pi}{2} \cdot \left(\frac{\delta}{2dk|\mathcal{Z}_R|} \right)^2 . \tag{4.8}$$

This event characterizes the properties needed from the first d measurements; Lemma 4.2.2 shows that it has large probability mass. The proof, given in Appendix 4.2.h, is based on known properties of Gaussian random matrices.

Lemma 4.2.2. *For any* $\delta \in (0,1)$, $\mathbf{Pr}\left(\mathcal{E}_\delta\right) \geq 1 - \delta$.

We now show in Lemma 4.2.3 that, for each $t \in [k]$, there is a relatively short vector in $\Lambda(\boldsymbol{B}^{(t)})$ that provides the correspondence information needed to recover $\boldsymbol{x}_t$. We also show in Lemma 4.2.4 that when β is sufficiently large, other vectors in $\Lambda(\boldsymbol{B}^{(t)})$ are considerably longer, and hence cannot be returned by LLL.

To simplify notation, assume that $\pi_{d+1}(j) = j$ for each $j \in [k]$, so we have $y_{d+1,t} = \langle \boldsymbol{w}_{d+1}, \boldsymbol{x}_t \rangle$ for each $t \in [k]$. Using this notation, define $\boldsymbol{z}^{(t)} = (z^{(t)}_{i,j} : i \in [d], j \in [k]) \in \mathbb{Z}^{dk}$ for each $t \in [k]$ by

$$z^{(t)}_{i,j} \; := \; \begin{cases} 1 & \text{if } \pi_i(j) = t \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

Recall that for each $t \in [k]$, the lattice vector in $\Lambda(\boldsymbol{B}^{(t)})$ determined by coefficient vector $(z_0, \boldsymbol{z}) \in$

$\mathbb{Z} \times \mathbb{Z}^{dk}$ is denoted by

$$L^{(t)}(z_0, \boldsymbol{z}) \;=\; z_0 \boldsymbol{b}_0^{(t)} + \sum_{i,j} z_{i,j} \boldsymbol{b}_{i,j}^{(t)} \,.$$

Observe that the coefficient vector $(z_0, \boldsymbol{z})$ is revealed in the first $dk + 1$ coordinates of the lattice vector $L^{(t)}(z_0, \boldsymbol{z})$; the final coordinate of the lattice vector is used to make some vectors very long.

Lemma 4.2.3. *On the event $\mathcal{E}_\delta$, for each $t \in [k]$, $y_{d+1,t} = \langle \mathbf{w}_{d+1}, \boldsymbol{x}_t \rangle = \sum_{i,j} a_{i,j} z_{i,j}^{(t)}$. Also on this event, for each $t \in [k]$,*

$$\begin{aligned} L^{(t)}(1, \boldsymbol{z}^{(t)}) \;&=\; \begin{bmatrix} 1 \\ \boldsymbol{z}^{(t)} \\ -\beta y_{d+1,t} + \beta \sum_{i,j} a_{i,j} z_{i,j}^{(t)} \end{bmatrix} \;=\; \begin{bmatrix} 1 \\ \boldsymbol{z}^{(t)} \\ 0 \end{bmatrix} \,, \\ \left\| L^{(t)}(1, \boldsymbol{z}^{(t)}) \right\|_2 \;&=\; \sqrt{d+1} \,. \end{aligned}$$

Proof. Assume $\mathcal{E}_\delta$ holds, which guarantees the existence of $\boldsymbol{W}^{-1}$ and thus permits the $\tilde{\boldsymbol{w}}_i$ to be well-defined. In this event, $\sum_{i=1}^d \tilde{\boldsymbol{w}}_i \boldsymbol{w}_i^\top = \boldsymbol{W}^{-1}\boldsymbol{W} = \boldsymbol{I}_d$. Therefore,

$$\begin{aligned} \sum_{i,j} a_{i,j} z_{i,j}^{(t)} \;&=\; \sum_{i=1}^{d} \sum_{j=1}^{k} \langle \mathbf{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle \langle \boldsymbol{w}_i, \boldsymbol{x}_{\pi_i(j)} \rangle z_{i,j}^{(t)} \\ &=\; \sum_{i=1}^{d} \langle \mathbf{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle \langle \boldsymbol{w}_i, \boldsymbol{x}_t \rangle \\ &=\; \mathbf{w}_{d+1}^\top \left(\sum_{i=1}^{d} \tilde{\boldsymbol{w}}_i \boldsymbol{w}_i^\top \right) \boldsymbol{x}_t \;=\; \langle \mathbf{w}_{d+1}, \boldsymbol{x}_t \rangle \,. \end{aligned}$$

The claim now follows by direct computation, using the above identity and the definition of $\boldsymbol{z}^{(t)}$.

□

Lemma 4.2.4. *For any $\delta \in (0, 1)$, conditional on the event $\mathcal{E}_\delta$, with probability at least $1 - \delta$ (over the choice of $\mathbf{w}_{d+1}$), for each $t \in [k]$, every coefficient vector $(z_0, \boldsymbol{z}) \in \mathbb{Z} \times \mathbb{Z}^{dk}$ that is not an* *integer multiple of* $(1, \boldsymbol{z}^{(t)})$ *satisfies*

$$\left\|L^{(t)}(z_0, \boldsymbol{z})\right\|_2 \;>\; \min\left\{R,\; \sqrt{z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_\delta^2}\right\} .$$

Proof. Assume $\mathcal{E}_\delta$ holds. This implies, in particular, that $\boldsymbol{W}^{-1}$ and the $\tilde{\boldsymbol{w}}_i$ are well-defined. Fix $t \in [k]$, and let $\mathbb{Z}(1, \boldsymbol{z}^{(t)})$ denote the set of integer multiples of $(1, \boldsymbol{z}^{(t)})$. For any coefficient vector $(z_0, \boldsymbol{z}) \in \mathbb{Z} \times \mathbb{Z}^{dk}$,

$$\|L^{(t)}(z_0, \boldsymbol{z})\|_2^2 \;=\; \left\| z_0 \boldsymbol{b}_0^{(t)} + \sum_{i,j} z_{i,j} \boldsymbol{b}_{i,j}^{(t)} \right\|_2^2 \;=\; z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 \left(\sum_{i,j} a_{i,j} z_{i,j} - y_{d+1,t} z_0 \right)^2 . \quad (4.9)$$

Observe that $\|L^{(t)}(z_0, \boldsymbol{z})\|_2 > R$ for all $(z_0, \boldsymbol{z}) \in (\mathbb{Z} \times \mathbb{Z}^{dk}) \setminus \mathcal{Z}_R$. Below, we prove that with probability at least $1 - \delta/k$, $\|L^{(t)}(z_0, \boldsymbol{z})\|_2^2 > z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_\delta^2$ for every $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R \setminus \mathbb{Z}(1, \boldsymbol{z}^{(t)})$. Combining this with a union bound over all choices of $t \in [k]$ proves the lemma.

Fix any such $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$, and consider the parenthesized term in Eq. (4.9) (without the squaring). By Lemma 4.2.3, the term expands to

$$\begin{aligned} \sum_{i,j} a_{i,j} z_{i,j} - y_{d+1,t} z_0 \;&=\; \sum_{i,j} a_{i,j} \left(z_{i,j} - z_{i,j}^{(t)} z_0 \right) \\ &=\; \sum_{i,j} \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle \langle \boldsymbol{w}_i, \boldsymbol{x}_{\pi_i(j)} \rangle \left(z_{i,j} - z_{i,j}^{(t)} z_0 \right) \;=\; \langle \boldsymbol{w}_{d+1}, \boldsymbol{v} \rangle , \end{aligned}$$

where

$$\boldsymbol{v} \;:=\; \sum_{i,j} \langle \boldsymbol{w}_i, \boldsymbol{x}_{\pi_i(j)} \rangle \left(z_{i,j} - z_{i,j}^{(t)} z_0 \right) \tilde{\boldsymbol{w}}_i .$$

Because $\boldsymbol{w}_{d+1} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, the final expression is a $\mathrm{N}(0, \|\boldsymbol{v}\|_2^2)$ random variable, and hence by Proposition 4.2.8 (given in Appendix 4.2.e),

$$\mathbf{Pr}\left(\langle \boldsymbol{w}_{d+1}, \boldsymbol{v} \rangle^2 \leq \frac{\pi}{2} \cdot \left(\frac{\delta}{k|\mathcal{Z}_R|} \right)^2 \cdot \|\boldsymbol{v}\|_2^2 \right) \;\leq\; \frac{\delta}{k|\mathcal{Z}_R|} . \quad (4.10)$$

We show below that, on the event $\mathcal{E}_\delta$,

$$\|\boldsymbol{v}\|_2^2 \;\geq\; \frac{\lambda(\boldsymbol{X})^2 \cdot \frac{\pi}{2} \cdot \left(\frac{\delta}{2dk|\mathcal{Z}_R|}\right)^2}{\left(2\sqrt{d} + \sqrt{2\ln(2/\delta)}\right)^2} \,. \tag{4.11}$$

Using this bound with Eq. (4.10) and a union bound, it follows that with probability at least $1-\delta/k$, we have $\|L^{(t)}(z_0, \boldsymbol{z})\|_2^2 > z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_\delta^2$ for all $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R \setminus \mathbb{Z}(1, \boldsymbol{z}^{(t)})$.

We now prove the bound in Eq. (4.11) on the event $\mathcal{E}_\delta$. Because the $\tilde{\boldsymbol{w}}_i$ are the columns of $\boldsymbol{W}^{-1}$, we may write $\boldsymbol{v} = \boldsymbol{W}^{-1}\boldsymbol{c}$ for $\boldsymbol{c} = (c_1, c_2, \ldots, c_d)$, where

$$c_i \;:=\; \sum_{j=1}^{k} \langle \boldsymbol{w}_i, \boldsymbol{x}_{\pi_i(j)} \rangle \left(z_{i,j} - z_{i,j}^{(t)} z_0\right) \;=\; \left\langle \boldsymbol{w}_i, \left(z_{i,\pi_i^{-1}(t)} - z_0\right)\boldsymbol{x}_t + \sum_{j \in [k] : \pi_i(j) \neq t} z_{i,j}\boldsymbol{x}_{\pi_i(j)} \right\rangle$$

for each $i \in [d]$. Therefore, $\|\boldsymbol{v}\|_2^2$ may be bounded below as

$$\|\boldsymbol{v}\|_2^2 \;\geq\; \sigma_d(\boldsymbol{W}^{-1})^2 \cdot \sum_{i=1}^{d} c_i^2 \;=\; \frac{1}{\|\boldsymbol{W}\|_2^2} \cdot \sum_{i=1}^{d} c_i^2 \,.$$

Since $(z_0, \boldsymbol{z}) \notin \mathbb{Z}(1, \boldsymbol{z}^{(t)})$, at least one of the following is true:

1. there exists $i \in [d]$ such that $z_{i,\pi_i^{-1}(t)} \neq z_0$;

2. there exists $i \in [d]$ and $j \in [k] \setminus \{\pi_i^{-1}(t)\}$ such that $z_{i,\pi_i(j)} \neq 0$.

In either case, there exists $i \in [d]$ such that $|z_{i,\pi_i^{-1}(t)} - z_0| + \sum_{j \in [k]:\pi_i(j) \neq t} |z_{i,\pi_i(j)}| > 0$, so using the third condition in the event $\mathcal{E}_\delta$,

$$\sum_{i=1}^{d} c_i^2 \;\geq\; \lambda(\boldsymbol{X})^2 \cdot \frac{\pi}{2} \cdot \left(\frac{\delta}{2dk|\mathcal{Z}_R|}\right)^2 \,.$$

Combining this with the upper-bound $\|\boldsymbol{W}\|_2 \leq 2\sqrt{d} + \sqrt{2\ln(2/\delta)}$ from the second condition in the event $\mathcal{E}_\delta$ proves the required lower-bound on $\|\boldsymbol{v}\|_2^2$ from Eq. (4.11). □

We now prove Theorem 20. With probability at least $1-\delta/2$ (over the choice of $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d$),

1. Event $\mathcal{E}_{\delta/2}$ holds (Lemma 4.2.2).

Moreover, conditional on $\mathcal{E}_{\delta/2}$,

2. $\|L^{(t)}(1, \boldsymbol{z}^{(t)})\|_2 = \sqrt{d+1}$ for each $t \in [k]$ (Lemma 4.2.3);

and, with probability at least $1 - \delta/2$ (over the choice of $\boldsymbol{w}_{d+1}$),

3. for each $t \in [k]$, every non-zero vector in $L^{(t)}(z_0, \boldsymbol{z}) \in \Lambda(\boldsymbol{B}^{(t)})$ for $(z_0, \boldsymbol{z}) \in \mathbb{Z} \times \mathbb{Z}^{dk}$ with length at most $R = 2^{(dk+1)/2}\sqrt{d+1}$ is either an integer multiple of $L^{(t)}(1, \boldsymbol{z}^{(t)})$, or has length $\|L^{(t)}(z_0, \boldsymbol{z})\|_2 > \sqrt{z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_{\delta/2}^2}$; the length in this latter case is more than R when $\beta \geq R/r_{\delta/2}$ (Lemma 4.2.4).

Statements 1–3 above hold together with probability at least $1 - \delta$, so we assume that they hold. In particular, Algorithm 1 does not return "failure" upon checking if $\boldsymbol{W}$ singular. As long as $\beta \geq R/r_{\delta/2}$, for each $t \in [k]$, the approximate solution returned by LLL for $\Lambda(\boldsymbol{B}^{(t)})$ is $L^{(t)}(\widehat{z}_0, \widehat{\boldsymbol{z}}) = L^{(t)}(c, c\boldsymbol{z}^{(t)})$ for some $c \neq 0$. The $(dk+2)$-th coordinate of this vector is zero—so Algorithm 1 does not return "failure" on account of this check—and $\widehat{\boldsymbol{x}}_t$ is obtained as a solution to the system of linear equations

$$\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle \;=\; y_{i,j}\,, \qquad (i,j) \in [d] \times [k] \,\boldsymbol{.}\, cz_{i,j}^{(t)} \neq 0\,.$$

By the definition of $\boldsymbol{z}^{(t)}$ and non-singularity of $\boldsymbol{W}$, we have $\widehat{\boldsymbol{x}}_t = \boldsymbol{x}_t$ for all $t \in [k]$. This completes the proof of Theorem 20.

Phase retrieval

The special case of correspondence retrieval where $k = 2$ and $\boldsymbol{x}_1 = -\boldsymbol{x}_2 \neq \boldsymbol{0}$ is known as the (real-valued) phase retrieval problem, as described in the introduction. Indeed, it is easy to see that the general $k = 2$ correspondence retrieval problem may be reduced to this case by "centering" the measurements. However, the unknown points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are no longer linearly independent, so Algorithm 1 is not directly applicable.

A simple fix is to pick a random vector $\boldsymbol{z} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, and replace each unordered measurement $\mathcal{M}_i = \{\langle \boldsymbol{w}_i, \boldsymbol{x}_1 \rangle, \langle \boldsymbol{w}_i, \boldsymbol{x}_2 \rangle\}$ with $\mathcal{M}'_i := \{\langle \boldsymbol{w}_i, \boldsymbol{z} \rangle + \langle \boldsymbol{w}_i, \boldsymbol{x}_1 \rangle, \langle \boldsymbol{w}_i, \boldsymbol{z} \rangle + \langle \boldsymbol{w}_i, \boldsymbol{x}_2 \rangle\}$. The points to recover become $\boldsymbol{z} + \boldsymbol{x}$ and $\boldsymbol{z} - \boldsymbol{x}$, where $\boldsymbol{x} := \boldsymbol{x}_1 = -\boldsymbol{x}_2$. Let $\tilde{\boldsymbol{X}} := [\boldsymbol{z} + \boldsymbol{x} | \boldsymbol{z} - \boldsymbol{x}] \in \mathbb{R}^{d \times 2}$. The following proposition gives a bound on $\kappa(\tilde{\boldsymbol{X}}) = \sigma_1(\tilde{\boldsymbol{X}})/\sigma_2(\tilde{\boldsymbol{X}})$; its proof is given in Appendix 4.2.h.

Proposition 4.2.5. *For any vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^d$, the matrix $\boldsymbol{M} := [\boldsymbol{a} + \boldsymbol{b} | \boldsymbol{a} - \boldsymbol{b}] \in \mathbb{R}^{d \times 2}$ satisfies*

$$\frac{\sigma_1(\boldsymbol{M})}{\sigma_2(\boldsymbol{M})} \leq \frac{r + 1/r}{|\sin(\theta)|} ,$$

where $r := \|\boldsymbol{a}\|_2 / \|\boldsymbol{b}\|_2$, and θ is the angle between $\boldsymbol{a}$ and $\boldsymbol{b}$.

It is easy to see that

$$\kappa(\tilde{\boldsymbol{X}}) \leq \frac{r + 1/r}{|\sin(\theta)|} \leq O\left(\frac{\|\boldsymbol{x}\|_2}{\sqrt{d}} + \frac{\sqrt{d}}{\|\boldsymbol{x}\|_2} \right)$$

with high probability, and hence Algorithm 1 may be applied.

We can also give a direct algorithm for solving the phase retrieval problem via LLL, with qualitatively the same guarantees as Algorithm 1, where $\|\boldsymbol{x}\|_2$ replaces the role of $\lambda(\boldsymbol{X})$. The details are given in Appendix 4.2.g.

Number of measurements. Our algorithms require $n = d + 1$ measurements for exact recovery, which is the best possible (in dimension $d \geq 2$), even in this phase retrieval setting. With only d linearly independent measurement vectors, no algorithm can distinguish among 2^{d-1} distinct solutions (of the form $\{\boldsymbol{W}^{-1} \operatorname{diag}(\boldsymbol{s}) \boldsymbol{W} \boldsymbol{x}, -\boldsymbol{W}^{-1} \operatorname{diag}(\boldsymbol{s}) \boldsymbol{W} \boldsymbol{x}\}$ for $\boldsymbol{s} \in \{\pm 1\}^d$) that give rise to the same d measurements.

As discussed in the introduction, [103] prove that $n = 2d - 1$ measurement vectors (whether random or deterministic) are necessary to ensure that every non-zero $\boldsymbol{x} \in \mathbb{R}^d$ can be recovered, up to sign, from measurements with these measurement vectors. Because our algorithms only use $d + 1$ (Gaussian) measurement vectors, they must be insufficient for recovering some $\boldsymbol{x}$ up to sign (in dimension $d \geq 3$), even though for any fixed $\boldsymbol{x}$, they suffice with high probability.

4.2.c Noisy correspondence retrieval

This section sketches a moment-based algorithm for the noisy correspondence retrieval problem.

Main idea

The algorithm is based on decomposing the following moments involving the k unknown points:

$$\boldsymbol{M} := \sum_{j=1}^{k} \boldsymbol{x}_j^{\otimes 2} \in \mathbb{R}^{d \times d} \qquad \text{and} \qquad \boldsymbol{T} := \sum_{j=1}^{k} \boldsymbol{x}_j^{\otimes 3} \in \mathbb{R}^{d \times d \times d}.$$

Under the condition $\operatorname{rank}(\boldsymbol{X}) = k$, there is an efficient algorithm based on tensor decompositions that, if given $\boldsymbol{M}$ and $\boldsymbol{T}$ up to some sufficiently small error as inputs, returns accurate estimates of the points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k$ up to reordering [see, e.g., 105].

The crucial idea is that the moment matrix $\boldsymbol{M}$ and tensor $\boldsymbol{T}$ can be estimated from the data $(\boldsymbol{w}_1, \mathcal{M}_1^\sigma), (\boldsymbol{w}_2, \mathcal{M}_2^\sigma), \ldots, (\boldsymbol{w}_n, \mathcal{M}_n^\sigma)$, even though the measurements are unordered. This was observed by [104] in the case of a related (and indeed, more difficult) model of mixtures of linear regressions. In their model, there is no noise (i.e., $\sigma = 0$), but instead of observing all of $\mathcal{M}_i$, only a random element of $\mathcal{M}_i$ is observed (and this random choice is independent of the random measurement vectors, and identically distributed across all n measurements). Yi, Caramanis, and Sanghavi give an algorithm for learning the k unknown points when n is sufficiently large (nearly linear in d, polynomial in k and $\kappa(\boldsymbol{X})$).[2] Therefore, it is clear that the noiseless correspondence retrieval problem may be reduced to their noiseless mixtures of linear regressions problem.

Our main observation is that the same estimators designed for the noiseless setting may also be applied in the noisy setting.

[2] [104] also give a hybrid algorithm that combines alternating minimization with the moment-based algorithm. This hybrid algorithm can exactly recover the k unknown points in the noiseless setting.

Moment estimators

To estimate $\boldsymbol{M}$ and $\boldsymbol{T}$, we use

$$\begin{aligned} \widehat{\boldsymbol{M}} &:= \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{1}{2}\sum_{j=1}^{k}\left(\langle \boldsymbol{w}_i, \boldsymbol{x}_j\rangle + \sigma\varepsilon_{i,j}\right)^2\left(\boldsymbol{w}_i^{\otimes 2} - \boldsymbol{I}_d\right)\right\} \\ \text{and} \quad \widehat{\boldsymbol{T}} &:= \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{1}{6}\sum_{j=1}^{k}\left(\langle \boldsymbol{w}_i, \boldsymbol{x}_j\rangle + \sigma\varepsilon_{i,j}\right)^3\left(\boldsymbol{w}_i^{\otimes 3} - \mathcal{T}(\boldsymbol{w}_i)\right)\right\}, \end{aligned}$$

respectively. Here, for any vector $\boldsymbol{v} \in \mathbb{R}^d$, the third-order tensor $\mathcal{T}(\boldsymbol{v})$ is defined by $\mathcal{T}(\boldsymbol{v}) := \sum_{j=1}^{d}\left(\boldsymbol{v}\otimes\boldsymbol{e}_j\otimes\boldsymbol{e}_j + \boldsymbol{e}_j\otimes\boldsymbol{v}\otimes\boldsymbol{e}_j + \boldsymbol{e}_j\otimes\boldsymbol{e}_j\otimes\boldsymbol{v}\right)$, where $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_d$ is any fixed orthonormal basis for $\mathbb{R}^d$. The i-th term in each of $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{T}}$ is symmetric with respect to the k values in $\mathcal{M}_i^{\sigma}$, and hence can be formed using just the unordered measurements.

The unbiasedness of $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{T}}$ in the noiseless case ($\sigma = 0$) follows immediately from the following proposition. We give a simple proof in Appendix 4.2.h for completeness.

Proposition 4.2.6 ([104])**.** *Let $\boldsymbol{w} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. For any vector $\boldsymbol{u} \in \mathbb{R}^d$,*

$$\mathbf{E}\left[\frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^2\left(\boldsymbol{w}^{\otimes 2} - \boldsymbol{I}_d\right)\right] = \boldsymbol{u}^{\otimes 2}, \qquad \mathbf{E}\left[\frac{1}{6}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^3\left(\boldsymbol{w}^{\otimes 3} - \mathcal{T}(\boldsymbol{w})\right)\right] = \boldsymbol{u}^{\otimes 3}.$$

In the noisy case, we have the following analogous proposition, which implies the unbiasedness of $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{T}}$ for any noise level $\sigma \geq 0$.

Proposition 4.2.7. *Let $\boldsymbol{w} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and $\varepsilon \sim \mathrm{N}(0, 1)$ be independent. For any vector $\boldsymbol{u} \in \mathbb{R}^d$ and any $\sigma \geq 0$,*

$$\mathbf{E}\left[\frac{1}{2}\left(\langle \boldsymbol{w}, \boldsymbol{u}\rangle + \sigma\varepsilon\right)^2\left(\boldsymbol{w}^{\otimes 2} - \boldsymbol{I}_d\right)\right] = \boldsymbol{u}^{\otimes 2}, \quad \mathbf{E}\left[\frac{1}{6}\left(\langle \boldsymbol{w}, \boldsymbol{u}\rangle + \sigma\varepsilon\right)^3\left(\boldsymbol{w}^{\otimes 3} - \mathcal{T}(\boldsymbol{w})\right)\right] = \boldsymbol{u}^{\otimes 3}.$$

Proof. This follows from Proposition 4.2.6 by replacing $\boldsymbol{w}$ and $\boldsymbol{u}$, respectively, with $\tilde{\boldsymbol{w}} := (\boldsymbol{w}, \varepsilon) \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_{d+1})$ and $\tilde{\boldsymbol{u}} := (\boldsymbol{u}, \sigma) \in \mathbb{R}^{d+1}$; and considering the appropriate sub-matrix and sub-tensor. □

Proposition 4.2.7 justifies the use of essentially the same moment-based algorithm of Yi, Caramanis, and Sanghavi for the noisy correspondence retrieval problem:

1. Compute the estimates $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{T}}$ from $(\boldsymbol{w}_1, \mathcal{M}_1^\sigma), (\boldsymbol{w}_2, \mathcal{M}_2^\sigma), \ldots, (\boldsymbol{w}_n, \mathcal{M}_n^\sigma)$.
2. Apply the tensor decomposition algorithm of [105], and return the vectors from the approximate decomposition $\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{x}}_2, \ldots, \widehat{\boldsymbol{x}}_k$.

The analysis of Yi, Caramanis, and Sanghavi can be used to give a bound on the number of measurements needed to accurately estimate the k unknown points: assuming $\max_{j\in k]} \|\boldsymbol{x}_j\|_2 = 1$, for any $\varepsilon, \delta \in (0,1)$, if the number of measurements n satisfies

$$n \;\geq\; \tilde{O}\left(d \cdot \operatorname{poly}\left(\frac{k}{\sigma_k(\boldsymbol{X}_\sigma)}\right) \cdot \frac{\log(1/\delta)}{\varepsilon^2} + \frac{k^2}{\delta}\right) ,$$

then the algorithm returns $\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{x}}_2, \ldots, \widehat{\boldsymbol{x}}_k \in \mathbb{R}^d$ satisfying

$$\min_{\pi} \max_{j\in[k]} \|\widehat{\boldsymbol{x}}_{\pi(j)} - \boldsymbol{x}_j\|_2 \;\leq\; \varepsilon ,$$

with probability at least $1-\delta$, where the $\min$ is over permutations $\pi \colon [k] \to [k]$. Here, the $\tilde{O}(\cdot)$ hides factors that are poly-logarithmic in those that appear, and $\boldsymbol{X}_\sigma$ is the $(d+1)\times k$ matrix that appends a row to $\boldsymbol{X}$ with all entries equal to σ. We omit a detailed bound and analysis because they are based entirely on the results of Yi, Caramanis, and Sanghavi, and the result is not comparable to the results we obtain in the noiseless setting with the lattice-based algorithms.

4.2.d Discussion

The moment-based algorithm for the correspondence retrieval problem does not appear to efficiently use the information contained in individual measurements. By averaging over the measurements in the computation of $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{T}}$, critical constraint information is lost. In contrast, the lattice-based algorithm does not average over the projection values nor the measurements themselves. It would be interesting to understand if there is indeed a gap between these distinct types of algorithms.

It would also be interesting to consider other classes of measurement vectors. Assuming a Gaussian distribution is convenient for analysis of our lattice-based algorithm, although it is plausible that other distributions satisfying some kind of anti-concentration condition at every point would also suffice. Handling certain discrete distributions would also simplify the numerical precision issues. The moment-based algorithm, however, critically relies on higher-order moment calculations specific to the Gaussian distribution. It is not clear to what extent that algorithm would work with other classes of measurement vectors. A plausible alternative is to use semidefinite programming to recover $\boldsymbol{M}$ and $\boldsymbol{T}$ (or other related moment tensors). Indeed, the results of [106] imply that $\boldsymbol{M}$ can be recovered from $O(dk)$ measurements, where the distribution of the measurement vectors may be Gaussian or from a certain class of finitely-supported distributions.

Our lattice-based algorithm cannot handle measurement noise, with the cryptographic hardness of the Shortest Vector Problem being the main barrier. There is also cryptographic evidence that even deterministic measurement errors make related problems computationally intractable [107]. In practice, LLL has been observed to find the shortest vector in lattices in low dimensions, and in high dimensions, its empirical performance is somewhat better than the worst-case approximation factor [108]. Nevertheless, it is desirable to find different algorithms for phase retrieval and correspondence retrieval that do not use LLL but still work with the same optimal number of measurements.

4.2.e Gaussian inequalities

Theorem 16 ([109, 110])**.** *Let $\boldsymbol{Z}$ be an $n \times n$ matrix whose entries are i.i.d.* $\mathrm{N}(0,1)$ *random variables. For any $\eta \in (0,1)$,*

$$\mathbf{Pr}\left(\sigma_n(\boldsymbol{Z}) \le \frac{\eta}{\sqrt{n}}\right) \le \eta \,,$$

and

$$\mathbf{Pr}\left(\sigma_1(\boldsymbol{Z}) \geq 2\sqrt{d} + \sqrt{2\ln(1/\eta)}\right) \;\leq\; \eta\,.$$

The following proposition is based on elementary properties of the Gaussian distribution.

Proposition 4.2.8. *Let $Z \sim \mathrm{N}(0,1)$. For any $\eta \in (0,1)$, $\mathbf{Pr}(Z^2 \leq \pi\eta^2/2) \leq \eta$, and $\mathbf{Pr}(|Z| > \sqrt{2\ln(2/\eta)}) \leq \eta$.*

Proof. The first bound is a standard Gaussian anti-concentration bound:

$$\mathbf{Pr}\left(|Z| \leq \sqrt{\frac{\pi}{2}}\eta\right) \;=\; \int_{-\sqrt{\frac{\pi}{2}}\eta}^{\sqrt{\frac{\pi}{2}}\eta} \frac{1}{\sqrt{2\pi}} e^{-z^2/2}\,\mathrm{d}z \;\leq\; \int_{-\sqrt{\frac{\pi}{2}}\eta}^{\sqrt{\frac{\pi}{2}}\eta} \frac{1}{\sqrt{2\pi}}\,\mathrm{d}z \;=\; \eta\,.$$

The second bound is a standard upper-bound on the Gaussian tail. □

4.2.f Numerical issues

In this section, we discuss the numerical issues with Algorithm 1. We assume that the coefficients of the measurement vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d$ and the k unknown points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k$ are represented with sufficiently fine precision—say, with B bits of precision—and that the projection values $\langle \boldsymbol{w}_i, \boldsymbol{x}_j \rangle$ in the measurements are exact. Here, B should be large enough so that the Gaussian anti-concentration properties in the proof of Theorem 20 still hold (say, within a constant multiplicative factor). The anti-concentration property from Proposition 4.2.8 is used with η no smaller than $\lambda(\boldsymbol{X}) \cdot 2^{-\mathrm{poly}(d,k,\log(1/\delta))}$, so the number of bits needed is $\mathrm{poly}(d,k,\log(1/\delta))$ plus the number of bits needed to represent $\lambda(\boldsymbol{X})$, the length of the shortest vector in $\Lambda(\boldsymbol{X})$. Recall that $\lambda(\boldsymbol{X})$ is no larger than $\min_{j\in[k]} \|\boldsymbol{x}_j\|_2$ and, by Proposition 4.2.1, no smaller than $\min_{j\in[k]} \|\boldsymbol{x}_j\|_2 / \kappa(\boldsymbol{X})$.

The numerical work performed by Algorithm 1 is dominated by the calls to LLL and the solving of linear systems. Lemma 4.2.9 bounds how much smaller or larger the coefficients of the lattice basis (used in the calls to LLL) are relative to the projection values. Lemma 4.2.9 also bounds the condition number of the matrix involved in the linear system that is used to solve for the unknown points.

Lemma 4.2.9. *With probability at least* $1 - \delta$,

$$\begin{aligned} \sigma_1(\boldsymbol{W}) \ &\leq \ 2\sqrt{d} + \sqrt{2\ln(4/\delta)}\,, \\ \sigma_d(\boldsymbol{W}) \ &\geq \ \frac{\delta}{4\sqrt{d}}\,, \\ \beta \cdot |a_{i,j}| \ &\in \ \left[\frac{\beta \cdot \sqrt{\pi}\delta \cdot |y_{i,j}|}{4\sqrt{2}d\left(2\sqrt{d} + \sqrt{2\ln(4/\delta)}\right)}, \frac{\beta \cdot 4\sqrt{2d\ln(8d/\delta)} \cdot |y_{i,j}|}{\delta} \right], \quad i \in [d], j \in [k]\,. \end{aligned}$$

Proof. From Theorem 16, it follows that

$$\mathbf{Pr}\left(\sigma_1(\boldsymbol{W}) \leq 2\sqrt{d} + \sqrt{2\ln(4/\delta)} \quad \text{and} \quad \sigma_d(\boldsymbol{W}) \geq \delta/(4\sqrt{d})\right) \ \geq \ 1 - \frac{\delta}{2}\,.$$

Condition on this $1 - \delta/2$ probability event. Recall that $\tilde{\boldsymbol{w}}_i$ is the i-th column of $\boldsymbol{W}^{-1}$. The distribution of each $\langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle$ is $\mathrm{N}(0,\|\tilde{\boldsymbol{w}}_i\|_2^2)$, and

$$\frac{1}{\sigma_1(\boldsymbol{W})} \ \leq \ \|\tilde{\boldsymbol{w}}_i\|_2 \ \leq \ \frac{1}{\sigma_d(\boldsymbol{W})}\,.$$

So, by Proposition 4.2.8 and union bound,

$$\mathbf{Pr}\left(\forall i \in [d] \centerdot \frac{\sqrt{\pi}\delta}{4\sqrt{2}d\sigma_1(\boldsymbol{W})} \leq |\langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle| \leq \frac{\sqrt{2\ln(8d/\delta)}}{\sigma_d(\boldsymbol{W})}\right) \ \geq \ 1 - \frac{\delta}{2}\,.$$

Since $a_{i,j} = \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle y_{i,j}$, combining these probability bounds proves the claim. □

When calling LLL, we may treat the lattice basis coefficients as integers by rescaling. By Lemma 4.2.9, the number of bits required to represent these coefficients may grow from B to

$$B + O\left(\log\max\left\{\frac{d^{3/2} + d\sqrt{\log(1/\delta)}}{\beta\delta}, \frac{\beta\sqrt{d\log(d/\delta)}}{\delta}\right\}\right)\,.$$

With the required value of β from the statement of Theorem 20, the running time of LLL—and also of Algorithm 1—is therefore $\mathrm{poly}(d, k, \log(B), \log(\kappa(\boldsymbol{X})), \log(1/\delta))$.

Algorithm 2 Lattice-based algorithm for phase retrieval

```
input  Data $(\boldsymbol{w}_i, y_i)$ for $i \in [d+1]$, parameter $\beta > 0$.
output  Hidden point $\widehat{\boldsymbol{x}}$ (up to a sign), or “failure”.
if $\boldsymbol{W} := [\boldsymbol{w}_1|\boldsymbol{w}_2|\cdots|\boldsymbol{w}_d]^\top$ is singular then
    return “failure”.
end if
Define $\boldsymbol{a} = (a_i : i \in [d]) \in \mathbb{R}^d$ by
    $$a_i \; := \; \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle y_i \, ,$$
    where $\tilde{\boldsymbol{w}}_i$ is the $i$-th column of $\boldsymbol{W}^{-1}$.
Construct basis
    $$\boldsymbol{B} \; = \; \begin{bmatrix} \boldsymbol{b}_0 & \boldsymbol{b}_1 & \cdots & \boldsymbol{b}_d \end{bmatrix} \; := \; \left[ \begin{array}{c|c} \multicolumn{2}{c}{\boldsymbol{I}_{d+1}} \\ \hline \beta y_{d+1} & -\beta \boldsymbol{a}^\top \end{array} \right] \; \in \; \mathbb{R}^{(d+2)\times(d+1)} \, .$$
Let $(\widehat{z}_0, \widehat{\boldsymbol{z}}) \in \mathbb{Z} \times \mathbb{Z}^d$ specify an approximate solution $\widehat{z}_0 \boldsymbol{b}_0 + \sum_i \widehat{z}_i \boldsymbol{b}_i \in \Lambda(\boldsymbol{B})$ to the Shortest
    Vector Problem for $\Lambda(\boldsymbol{B})$ using LLL.
if $|\widehat{z}_0| = |\widehat{z}_1| = |\widehat{z}_2| = \cdots = |\widehat{z}_d|$ is not true then
    return “failure”
end if
Let $\widehat{\boldsymbol{x}}$ be a solution to the system of linear equations (in $\boldsymbol{t} \in \mathbb{R}^d$)
    $$\langle \boldsymbol{w}_i, \boldsymbol{t} \rangle \; = \; \operatorname{sign}(\widehat{z}_i) y_i \, , \qquad i \in [d] \, .$$
return $\widehat{\boldsymbol{x}}$.
```

4.2.g Direct algorithm for phase retrieval

In the phase retrieval problem, there is a single hidden vector $\boldsymbol{x}$, and for each $i \in [d+1]$, we draw $\boldsymbol{w}_i \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and observe $y_i := \big|\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle\big|$. Our goal is to recover $\boldsymbol{x}$ by finding the vector of unknown signs $\boldsymbol{s} := (s_1, s_2, \ldots, s_d) \in \{\pm 1\}^d$, where $s_i := \operatorname{sign}(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ for each $i \in [d]$. A modified version of our main algorithm, specified in Algorithm 2, constructs a lattice where the shortest vector's coefficients are exactly the same as $\boldsymbol{s}$ or $-\boldsymbol{s}$.

The performance guarantee of this algorithm is given below in an analogous result to Theorem 20.

Theorem 17. *For any* $\delta \in (0,1)$, *if*

$$\beta \;\geq\; \frac{2^{d/2}\sqrt{d+1}\cdot 2d\left(2\sqrt{d}+\sqrt{2\ln(4/\delta)}\right)\cdot\left(2\cdot 2^{d/2}\sqrt{d+1}+1\right)^{d+1}}{\delta^2\|\boldsymbol{x}\|_2\,\pi}$$

then with probability at least $1-\delta$, *Algorithm 2 returns* $\widehat{\boldsymbol{x}}=\boldsymbol{x}$.

Let $\mathcal{E}_\delta$ be the event that

1. the smallest singular value of $\boldsymbol{W}$ is bounded from below: $\sigma_d(\boldsymbol{W}) \geq \delta/(4\sqrt{d})$;

2. the spectral norm of $\boldsymbol{W}$ is bounded from above: $\|\boldsymbol{W}\|_2 \leq 2\sqrt{d}+\sqrt{2\ln(4/\delta)}$;

3. for each $i \in [d]$ and $(z_0,\boldsymbol{z}) \in \mathcal{Z}_R$ such that $|z_i - z_0| > 0$,

$$\left\langle \boldsymbol{w}_i, (z_i - z_0)\boldsymbol{x}\right\rangle^2 \;\geq\; \|\boldsymbol{x}\|_2^2\cdot\frac{\pi}{2}\cdot\left(\frac{\delta}{2d|\mathcal{Z}_R|}\right)^2. \tag{4.12}$$

Also, let $R := 2^{d/2}\sqrt{d+1}$ and $\mathcal{Z}_R := \{(z_0,\boldsymbol{z}) \in \mathbb{Z}\times\mathbb{Z}^d : 0 < z_0^2 + \|\boldsymbol{z}\|_2^2 \leq R^2\}$.

Lemma 4.2.10. *For any* $\delta \in (0,1)$, $\mathbf{Pr}\left(\mathcal{E}_\delta\right) \geq 1-\delta$.

The proof of this lemma is completely analogous to that of Lemma 4.2.2, so we omit it.

Let $s_0 := \operatorname{sign}(\langle \boldsymbol{w}_{d+1}, \boldsymbol{x}\rangle)$. The following lemma shows there exists a short lattice vector which solves the recovery problem. Its proof is analogous to that of Lemma 4.2.3, so again we omit it.

Lemma 4.2.11. *On the event* $\mathcal{E}_\delta$

$$\begin{aligned} L(s_0,\boldsymbol{s}) \;&=\; \begin{bmatrix} 1 \\ \boldsymbol{s} \\ -\beta s_0 y_{d+1} + \beta\sum_i^d a_i s_i \end{bmatrix} \;=\; \begin{bmatrix} 1 \\ \boldsymbol{s} \\ 0 \end{bmatrix}, \\ \|L(s_0,\boldsymbol{s})\|_2 \;&=\; \sqrt{d+1}\,. \end{aligned}$$

Finally, we state a lemma that lower-bounds the length of lattice vectors that are not integer multiples of $L(s_0,\boldsymbol{s})$.

Lemma 4.2.12. *For any $\delta \in (0,1)$, conditioned on the event $\mathcal{E}_\delta$, for every coefficient vector $(z_0, \boldsymbol{z})$ that is not an integer multiple of $(s_0, \boldsymbol{s})$, we have*

$$\|L(z_0, \boldsymbol{z})\|_2^2 \; > \; \min\left\{R^2,\, z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_\delta^2\right\} .$$

where

$$r_\delta \; := \; \delta^2 \cdot \frac{\|\boldsymbol{x}\|_2 \, \pi}{2d|\mathcal{Z}_R|(2\sqrt{d} + \sqrt{2\ln(4/\delta)})} .$$

Proof. Let $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$ be any coefficient vector. Then the last coordinate of the corresponding lattice vector is

$$\sum_{i=1}^{d} a_i z_i - z_0 y_{d+1} \; = \; \sum_{i=1}^{d} a_i z_i - z_0 s_0 \sum_{i=1}^{d} a_i s_i$$

(using the relation from Lemma 4.2.11 and the fact that $s_0^2 = 1$)

$$\begin{aligned} &= \sum_{i=1}^{d} a_i \left(z_i - z_0 s_0 s_i\right) \\ &= \sum_{i=1}^{d} \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle \big|\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle\big| \left(z_i - z_0 s_0 s_i\right) . \end{aligned}$$

Because $\boldsymbol{z}$ is not an integer multiple of $\boldsymbol{s}$ and $z_0 s_0$ is an integer, there exists an index $i^* \in [d]$ such that $z_i - z_0 s_0 s_{i^*} \neq 0$. Then the sum can be rewritten as

$$\langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_{i^*} \rangle \big|\langle \boldsymbol{w}_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle\big| + \sum_{i \neq i^*} \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i \rangle \big|\langle \boldsymbol{w}_i, (z_i - z_0 s_0 s_i)\boldsymbol{x}\rangle\big| \; . \tag{4.13}$$

We now show that the first term puts small probability mass over any short interval independent of the value of the summation over $i \neq i^*$. This gives a lower bound on the absolute value of the last coordinate by considering an interval around the negative of the second term.

Since $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$ implies $(z_0 s_0 s_{i^*}, \boldsymbol{z}) \in \mathcal{Z}_R$ for either of the two values of s_0 and s_{i^*}, the third condition in $\mathcal{E}_\delta$ gives

$$\left|\langle \boldsymbol{w}_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle\right| \;\geq\; \|\boldsymbol{x}\|_2^2 \cdot \frac{\pi}{2} \cdot \left(\frac{\delta}{2d|\mathcal{Z}_R|}\right)^2 .$$

Since $\boldsymbol{W}^{-1}$ is full rank, there is a component of $\tilde{\boldsymbol{w}}_{i^*}$ which is orthogonal to the span of $\{\tilde{\boldsymbol{w}}_i\}_{i \neq i^*}$. We write this as

$$\boldsymbol{u} \;=\; \tilde{\boldsymbol{w}}_{i^*} + \sum_{i \neq i^*} a_i \boldsymbol{w}_i$$

where $\langle \boldsymbol{u}, \boldsymbol{w}_i\rangle = 0$ for all $i \neq i^*$. Now let a_i for $i \neq i^*$ be the coefficients above, and let $a_{i^*} := 1$. Then $\boldsymbol{u} = \boldsymbol{W}^{-1}\boldsymbol{a}$ for $\boldsymbol{a} = (a_1, a_2, \ldots, a_d)$, and thus

$$\|\boldsymbol{u}\|_2 \;\geq\; \frac{1}{2\sqrt{d} + \sqrt{2\ln(4/\delta)}} \cdot \|\boldsymbol{a}\|_2 \;\geq\; \frac{1}{2\sqrt{d} + \sqrt{2\ln(4/\delta)}} ,$$

where the first inequality follows from

$$\sigma_d(W^{-1}) \;\geq\; \frac{1}{2\sqrt{d} + \sqrt{2\ln(4/\delta)}}$$

on the event $\mathcal{E}_\delta$ and the second inequality from $a_{i^*} = 1$.

Thus Eq. (4.13) can be rewritten as the sum of two independent terms

$$\begin{aligned}&\langle \boldsymbol{w}_{d+1}, \boldsymbol{u}\rangle \left|\langle \boldsymbol{w}_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle\right| + \\ &\quad \sum_{i \neq i^*} \Big(\langle \boldsymbol{w}_{d+1}, -a_i \tilde{\boldsymbol{w}}_i\rangle \left|\langle \boldsymbol{w}_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle + \langle \boldsymbol{w}_{d+1}, \tilde{\boldsymbol{w}}_i\rangle \right|\langle \boldsymbol{w}_i, (z_i - z_0 s_0 s_i)\boldsymbol{x}\rangle\big| \Big) \quad (4.14)\end{aligned}$$

The first term, $\langle \boldsymbol{w}_{d+1}, \boldsymbol{u}\rangle \big|\langle \boldsymbol{w}_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle\big|$, has distribution $\mathrm{N}(0, \sigma^2)$, where

$$\begin{aligned} \sigma^2 \;&\geq\; \|\boldsymbol{u}\|_2 |\langle w_{i^*}, (z_{i^*} - z_0 s_0 s_{i^*})\boldsymbol{x}\rangle| \\ &\geq\; \frac{\|\boldsymbol{x}\|_2^2 \frac{\pi}{2} \left(\frac{\delta}{2d|\mathcal{Z}_R|}\right)^2}{\left(2\sqrt{d} + \sqrt{2\ln(4/\delta)}\right)^2} . \end{aligned}$$

The event that Eq. (4.13) is small is when the Gaussian distribution returns a value in the interval of length $2r_\delta$ centered around the second term. The probability of this event is no more than

$$\frac{1}{\sqrt{2\pi\sigma^2}} \cdot 2r_\delta \;\leq\; \frac{2r_\delta}{\pi} \cdot \frac{2\sqrt{d} + \sqrt{2\ln(4/\delta)}}{\|\boldsymbol{x}\|_2 \left(\frac{\delta}{2d|\mathcal{Z}_R|}\right)} \;\leq\; \delta$$

by the choice of r_δ. Therefore, with probability at least $1 - \delta$, the quantity in 4.13 is at least r_δ, so the contribution of the last coordinate to the norm of the lattice vector is at least $\beta^2 r_\delta^2$, so the norm of this lattice vector is at least $\sqrt{z_0^2 + \|\boldsymbol{z}\|_2^2 + \beta^2 r_\delta^2}$. To complete the proof we note that for all $(z_0, \boldsymbol{z}) \notin \mathcal{Z}_R$, by definition the norm of $\|\boldsymbol{z}\|_2$ is at least R. □

We now prove Theorem 17. By the choices of R and β and Lemma 4.2.12, every incorrect coefficient vector has norm at least $2^{d/2}\sqrt{d+1}$, so it will not be returned by the LLL algorithm. By Lemma 4.2.11 there exists a short vector with coefficients $(s_0, \boldsymbol{s})$, so LLL recovers the correct signs.

4.2.h Omitted proofs

Proof of Proposition 4.2.1

Claim 4.2.13. $\lambda(\boldsymbol{X}) \geq \min_{i\in[k]} \left\|\boldsymbol{x}_i - \boldsymbol{\Pi}_{(-i)}\boldsymbol{x}_i\right\|_2$ *where* $\boldsymbol{\Pi}_{(-i)}$ *is the orthogonal projection to the span of* $\{\boldsymbol{x}_j\}_{j\neq i}$.

Proof. Let $\boldsymbol{v}$ be a non-zero vector in the lattice with basis $\boldsymbol{X}$. Write $\boldsymbol{v} = \sum_{j=1}^{k} z_j \boldsymbol{x}_j$, where

$z_1, z_2, \ldots, z_k \in \mathbb{Z}$. Pick any $i \in [k]$ such that $z_i \neq 0$, and let $\boldsymbol{r} := -\sum_{j \neq i} z_j \boldsymbol{x}_j$, so

$$\|\boldsymbol{v}\|_2^2 \;=\; \|z_i \boldsymbol{x}_i - \boldsymbol{r}\|_2^2 \;\geq\; \left\| z_i \boldsymbol{x}_i - \boldsymbol{\Pi}_{(-i)} z_i \boldsymbol{x}_i \right\|_2^2 \;=\; |z_i| \left\| \boldsymbol{x}_i - \boldsymbol{\Pi}_{(-i)} \boldsymbol{x}_i \right\|_2^2 \;\geq\; \left\| \boldsymbol{x}_i - \boldsymbol{\Pi}_{(-i)} \boldsymbol{x}_i \right\|_2^2 .$$

Above, the first inequality follows from the Pythagorean theorem, and the second inequality follows because $z_i \in \mathbb{Z} \setminus \{0\}$. □

By Claim 4.2.13, it suffices to lower-bound the distance between $\boldsymbol{x}_i$ and the subspace spanned by $\{\boldsymbol{x}_j\}_{j \neq i}$, for every $i \in [k]$. So fix $i \in [k]$, and any non-zero vector $\boldsymbol{r}_i$ in the span of $\{\boldsymbol{x}_j\}_{j \neq i}$. Let the singular value decomposition of $\boldsymbol{X}$ be given by $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{d \times k}$ has orthonormal columns, $\boldsymbol{S} = \operatorname{diag}(\sigma_1(\boldsymbol{X}), \sigma_2(\boldsymbol{X}), \ldots, \sigma_k(\boldsymbol{X})) \succ \boldsymbol{0}$ is diagonal, and $\boldsymbol{V} \in \mathbb{R}^{k \times k}$ is orthogonal. Let $\boldsymbol{\alpha}_j \in \mathbb{R}^k$ denote the j-th column of $\boldsymbol{V}^\top$. Then $\boldsymbol{x}_i = \boldsymbol{U}\boldsymbol{S}\boldsymbol{\alpha}_i$, and there exists non-zero $\boldsymbol{\beta}_i \in \mathbb{R}^k$ orthogonal to $\boldsymbol{\alpha}_i$ such that $\boldsymbol{r}_i = \boldsymbol{U}\boldsymbol{S}\boldsymbol{\beta}_i$. Moreover,

$$\frac{\langle \boldsymbol{x}_i, \boldsymbol{r}_i \rangle^2}{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle \langle \boldsymbol{r}_i, \boldsymbol{r}_i \rangle} \;=\; \frac{\left(\boldsymbol{\alpha}_i^\top \boldsymbol{S}\boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{S}\boldsymbol{\beta}_i\right)^2}{\left(\boldsymbol{\alpha}_i^\top \boldsymbol{S}\boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{S}\boldsymbol{\alpha}_i\right)\left(\boldsymbol{\beta}_i^\top \boldsymbol{S}\boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{S}\boldsymbol{\beta}_i\right)} \;=\; \frac{\left(\boldsymbol{\alpha}_i^\top \boldsymbol{S}^2 \boldsymbol{\beta}_i\right)^2}{\left(\boldsymbol{\alpha}_i^\top \boldsymbol{S}^2 \boldsymbol{\alpha}_i\right)\left(\boldsymbol{\beta}_i^\top \boldsymbol{S}^2 \boldsymbol{\beta}_i\right)} .$$

By Wielandt's inequality [111, p. 7.4.34], the ratio is bounded above by

$$\left(\frac{\sigma_1(\boldsymbol{S}^2)/\sigma_k(\boldsymbol{S}^2) - 1}{\sigma_1(\boldsymbol{S}^2)/\sigma_k(\boldsymbol{S}^2) + 1} \right)^2 \;=\; \left(\frac{\kappa(\boldsymbol{X})^2 - 1}{\kappa(\boldsymbol{X})^2 + 1} \right)^2 \;=:\; \phi .$$

By the Pythagorean theorem, the distance between $\boldsymbol{x}_i$ and the span of $\boldsymbol{r}_i$ is

$$\|\boldsymbol{x}_i\|_2 \left(1 - \frac{\langle \boldsymbol{x}_i, \boldsymbol{r}_i \rangle^2}{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle \langle \boldsymbol{r}_i, \boldsymbol{r}_i \rangle} \right)^{1/2} \;\geq\; \|\boldsymbol{x}_i\|_2 \sqrt{1 - \phi} .$$

Since this holds for any $\boldsymbol{r}_i$ in the span of $\{\boldsymbol{x}_j\}_{j \neq i}$, the distance between $\boldsymbol{x}_i$ and the span of $\{\boldsymbol{x}_j\}_{j \neq i}$ is also at least

$$\|\boldsymbol{x}_i\|_2 \sqrt{1 - \phi} \;=\; \|\boldsymbol{x}_i\|_2 \cdot \frac{2\kappa(\boldsymbol{X})}{\kappa(\boldsymbol{X})^2 + 1} .$$

The claim in Proposition 4.2.1 follows.

Proof of Lemma 4.2.2

It suffices to show the following probability bounds: (i) $\mathbf{Pr}(\sigma_d(\boldsymbol{W}) \leq \delta/(4\sqrt{d})) \leq \delta/4$; (ii) $\mathbf{Pr}(\|\boldsymbol{W}\|_2 > 2\sqrt{d} + \sqrt{2\ln(4/\delta)}) \leq \delta/4$; (iii) $\mathbf{Pr}$(Eq. (4.12) does not hold) $\leq \delta/(2dk|\mathcal{Z}_R|)$ for each $i \in [d]$, $j \in [k]$, and $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$ such that $|z_{i,j} - z_0| + \sum_{j' \neq j} |z_{i,j'}| > 0$. Combining these bounds with a union bound proves the claim.

The first two bounds follow from Theorem 16. The third requires Proposition 4.2.8 and the observation that the inner product in Eq. (4.12) is distributed as $\mathrm{N}(0,\|\boldsymbol{v}\|_2^2)$, where $\boldsymbol{v} := (z_{i,j} - z_0)\boldsymbol{x}_{\pi_i(j)} + \sum_{j' \neq j} z_{i,j'}\boldsymbol{x}_{\pi_i(j')}$. The condition on $(z_0, \boldsymbol{z})$ implies that $\boldsymbol{v}$ is a non-zero vector in the lattice $\Lambda(\boldsymbol{X})$, which has $\|\boldsymbol{v}\|_2 \geq \lambda(\boldsymbol{X})$ by definition.

Proof of Proposition 4.2.5

Let $\tilde{\boldsymbol{M}} := [\boldsymbol{a}|\boldsymbol{b}] \in \mathbb{R}^{d \times 2}$. The non-zero singular values of the matrix $\boldsymbol{M} = [\boldsymbol{a} + \boldsymbol{b}|\boldsymbol{a} - \boldsymbol{b}]$ are the same as the square-roots of the non-zero eigenvalues of

$$\boldsymbol{M}\boldsymbol{M}^\top = 2\boldsymbol{a}\boldsymbol{a}^\top + 2\boldsymbol{b}\boldsymbol{b}^\top = 2\tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^\top .$$

This matrix, in turn, has the same non-zero eigenvalues as the matrix

$$2\tilde{\boldsymbol{M}}^\top\tilde{\boldsymbol{M}} = 2\begin{bmatrix} \|\boldsymbol{a}\|_2^2 & \langle \boldsymbol{a}, \boldsymbol{b} \rangle \\ \langle \boldsymbol{b}, \boldsymbol{a} \rangle & \|\boldsymbol{b}\|_2^2 \end{bmatrix} .$$

The eigenvalues $\lambda_1 \geq \lambda_2$ of this matrix can be computed explicitly:

$$\lambda_1 = \|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2 + \sqrt{\left(\|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2\right)^2 - 4\left(\|\boldsymbol{a}\|_2^2\|\boldsymbol{b}\|_2^2 - \langle \boldsymbol{a}, \boldsymbol{b} \rangle^2\right)} ,$$

$$\lambda_2 = \|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2 - \sqrt{\left(\|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2\right)^2 - 4\left(\|\boldsymbol{a}\|_2^2\|\boldsymbol{b}\|_2^2 - \langle \boldsymbol{a}, \boldsymbol{b} \rangle^2\right)} .$$

Their ratio is

$$\frac{\lambda_1}{\lambda_2} \;=\; \frac{1+\sqrt{1-\frac{4\sin^2(\theta)}{\left(r+1/r\right)^2}}}{1-\sqrt{1-\frac{4\sin^2(\theta)}{\left(r+1/r\right)^2}}}\,,$$

where $r = \|\boldsymbol{a}\|_2 / \|\boldsymbol{b}\|_2$, and θ is the angle between $\boldsymbol{a}$ and $\boldsymbol{b}$. The quantity $4\sin^2(\theta)/\left(r+1/r\right)^2$ is always in the interval $[0,1]$. A Taylor series expansion argument shows that

$$\frac{1+\sqrt{1-x}}{1-\sqrt{1-x}} \;\leq\; \frac{4}{x}\,, \quad x \in [0,1]\;,$$

so we conclude

$$\frac{\sigma_1(\boldsymbol{M})}{\sigma_2(\boldsymbol{M})} \;=\; \sqrt{\frac{\lambda_1}{\lambda_2}} \;\leq\; \frac{r+1/r}{\left|\sin(\theta)\right|}\,.$$

Proof of Proposition 4.2.6

By homogeneity, we may assume $\|\boldsymbol{u}\|_2 = 1$. Let $g := \langle \boldsymbol{w}, \boldsymbol{u}\rangle \sim \mathrm{N}(0,1)$, and let $\boldsymbol{y} := \boldsymbol{w} - g\boldsymbol{u}$. Observe that g and $\boldsymbol{y}$ are independent, and

$$\mathbf{E}\,\boldsymbol{y} \;=\; \boldsymbol{0}\,, \qquad \mathbf{E}\,\boldsymbol{y}^{\otimes 2} \;=\; \boldsymbol{I}_d - \boldsymbol{u}^{\otimes 2} \;=\; \sum_{j=1}^{d} \boldsymbol{e}_j^{\otimes 2} - \boldsymbol{u}^{\otimes 2}\,, \qquad \mathbf{E}\,\boldsymbol{y}^{\otimes 3} \;=\; \boldsymbol{0}\,.$$

Using these facts, we have

$$\mathbf{E}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^2 \left(\boldsymbol{w}^{\otimes 2} - \boldsymbol{I}_d\right) \;=\; \mathbf{E}\,g^2\left(g^2\boldsymbol{u}^{\otimes 2} + \boldsymbol{y}^{\otimes 2} - \boldsymbol{I}_d\right) \;=\; \mathbf{E}\,g^4\boldsymbol{u}^{\otimes 2} - \mathbf{E}\,g^2\boldsymbol{u}^{\otimes 2} \;=\; 2\boldsymbol{u}^{\otimes 2}\,,$$

$$
\begin{aligned}
\mathbf{E}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^3 \boldsymbol{w}^{\otimes 3} &= \mathbf{E}\, g^3 \left(g\boldsymbol{u} + \boldsymbol{y}\right)^{\otimes 3} \\
&= \mathbf{E}\, g^3 \left(g^3 \boldsymbol{u}^{\otimes 3} + g\left(\boldsymbol{u} \otimes \boldsymbol{y} \otimes \boldsymbol{y} + \boldsymbol{y} \otimes \boldsymbol{u} \otimes \boldsymbol{y} + \boldsymbol{y} \otimes \boldsymbol{y} \otimes \boldsymbol{u}\right)\right) \\
&= \mathbf{E}\, g^6 \boldsymbol{u}^{\otimes 3} + \mathbf{E}\, g^4 \sum_{j=1}^{d} \left(\boldsymbol{u} \otimes \boldsymbol{e}_j \otimes \boldsymbol{e}_j + \boldsymbol{e}_j \otimes \boldsymbol{u} \otimes \boldsymbol{e}_j + \boldsymbol{e}_j \otimes \boldsymbol{e}_j \otimes \boldsymbol{u} - 3\boldsymbol{u}^{\otimes 3}\right) \\
&= 6\boldsymbol{u}^{\otimes 3} + 3\mathcal{T}(\boldsymbol{u})\,, \\
\mathbf{E}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^3 \boldsymbol{w} &= \mathbf{E}\, g^3 \left(g\boldsymbol{u} + \boldsymbol{y}\right) \;=\; 3\boldsymbol{u}\,,
\end{aligned}
$$

so $\mathbf{E}\langle \boldsymbol{w}, \boldsymbol{u}\rangle^3 \left(\boldsymbol{w}^{\otimes 3} - \mathcal{T}(\boldsymbol{w})\right) = 6\boldsymbol{u}^{\otimes 3}$. This proves the claims in Proposition 4.2.6.

4.3 Linear regression with an unknown permutation

Consider the problem of recovering an unknown vector $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ from noisy linear measurements when the correspondence between the measurement vectors and the measurements themselves is unknown. The measurement vectors (i.e., covariates) from $\mathbb{R}^d$ are denoted by $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n$; for each $i \in [n] := \{1, 2, \dots, n\}$, the i-th measurement (i.e., response) y_i is obtained using $\boldsymbol{x}_{\bar{\pi}(i)}$:

$$
y_i \;=\; \bar{\boldsymbol{w}}^\top \boldsymbol{x}_{\bar{\pi}(i)} + \varepsilon_i\,, \quad i \in [n]\,. \tag{4.15}
$$

Above, $\bar{\pi}$ is an unknown permutation on $[n]$, and the $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$ are unknown measurement errors.

This problem (which has been called *unlabeled sensing* [112], *linear regression with an unknown permutation* [113], and *linear regression with shuffled labels* [114]) arises in many settings. For example, physical sensing limitations may create ambiguity in or lose the ordering of measurements. Or, the covariates and responses may be derived from separate databases that lack appropriate record linkage (perhaps for privacy reasons). See the aforementioned references for more details on these applications. The problem is also interesting because the missing correspondence makes an otherwise well-understood problem into one with very different computational and statistical properties.

Prior works. [112] study conditions on the measurement vectors that permit recovery of any target vector $\bar{\boldsymbol{w}}$ under noiseless measurements. They show that when the entries of the $\boldsymbol{x}_i$ are drawn i.i.d. from a continuous distribution, and $n \geq 2d$, then almost surely, every vector $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ is uniquely determined by noiseless correspondence-free measurements as in (4.15). (Under noisy measurements, it is shown that $\bar{\boldsymbol{w}}$ can be recovered when an appropriate signal-to-noise ratio tends to infinity.) It is also shown that $n \geq 2d$ is necessary for such a guarantee that holds *for all* vectors $\bar{\boldsymbol{w}} \in \mathbb{R}^d$.

[113] study statistical and computational limits on recovering the unknown permutation $\bar{\pi}$. On the statistical front, they consider necessary and sufficient conditions on the signal-to-noise ratio $\mathsf{SNR} := \|\bar{\boldsymbol{w}}\|_2^2 / \sigma^2$ when the measurement errors $(\varepsilon_i)_{i=1}^n$ are i.i.d. draws from the normal distribution $\mathrm{N}(0, \sigma^2)$ and the measurement vectors $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from the standard multivariate normal distribution $\mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. Roughly speaking, exact recovery of $\bar{\pi}$ is possible via maximum likelihood when $\mathsf{SNR} \geq n^c$ for some absolute constant $c > 0$, and approximate recovery is impossible for any method when $\mathsf{SNR} \leq n^{c'}$ for some other absolute constant $c' > 0$. On the computational front, they show that the least squares problem (which is equivalent to maximum likelihood problem)

$$\min_{\boldsymbol{w}, \pi} \sum_{i=1}^n \left(\boldsymbol{w}^\top \boldsymbol{x}_{\pi(i)} - y_i\right)^2 \tag{4.16}$$

given arbitrary $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ and $y_1, y_2, \ldots, y_n \in \mathbb{R}$ is NP-hard when $d = \Omega(n)$[3], but admits a polynomial-time algorithm (in fact, an $O(n \log n)$-time algorithm based on sorting) when $d = 1$.

[114] observe that the maximum likelihood estimator can be inconsistent for estimating $\bar{\boldsymbol{w}}$ in certain settings (including the normal setting of [113], with SNR fixed but $n \to \infty$). One of the alternative estimators they suggest is consistent under additional assumptions in dimension $d = 1$. [116] give a $O(dn^{d+1})$-time algorithm that, in dimension $d = 2$, is guaranteed to approximately

[3] [113] prove that PARTITION reduces to the problem of deciding if the optimal value of (4.16) is zero or non-zero. Note that PARTITION is weakly, but not strongly, NP-hard: it admits a pseudo-polynomial-time algorithm [115, Section 4.2]. In Section 4.3.d, we prove that the least squares problem is strongly NP-hard by reduction from 3-PARTITION (which is strongly NP-complete [115, Section 4.2.2]).

recover $\bar{\boldsymbol{w}}$ when the measurement vectors are chosen in a very particular way from the unit circle and the measurement errors are uniformly bounded.

Contributions. We make progress on both computational and statistical aspects of the problem.

1. We give an approximation algorithm for the least squares problem from (4.16) that, any given $(\boldsymbol{x}_i)_{i=1}^n$, $(y_i)_{i=1}^n$, and $\varepsilon \in (0,1)$, returns a solution with objective value at most $1+\varepsilon$ times that of the minimum in time $(n/\varepsilon)^{O(d)}$. This a fully polynomial-time approximation scheme for any constant dimension.

2. We give an algorithm that exactly recovers $\bar{\boldsymbol{w}}$ in the measurement model from (4.15), under the assumption that there are no measurement errors and the covariates $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$. The algorithm, which is based on a reduction to a lattice problem and employs the lattice basis reduction algorithm of [98], runs in $\mathrm{poly}(n,d)$ time when the covariate vectors $(\boldsymbol{x}_i)_{i=1}^n$ and target vector $\bar{\boldsymbol{w}}$ are appropriately quantized. This result may also be regarded as *for each*-type guarantee for exactly recovering a fixed vector $\bar{\boldsymbol{w}}$, which complements the *for all*-type results of [112] concerning the number of measurement vectors needed for recovering all possible vectors.

3. We show that in the measurement model from (4.15) where the measurement errors are i.i.d. draws from $\mathrm{N}(0,\sigma^2)$ and the covariate vectors are i.i.d. draws from $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$, then no algorithm can approximately recover $\bar{\boldsymbol{w}}$ unless $\mathsf{SNR} \geq C \min\{1,\, d/\log\log(n)\}$ for some absolute constant $C > 0$. We also show that when the covariate vectors are i.i.d. draws from the uniform distribution on $[-1/2, 1/2]^d$, then approximate recovery is impossible unless $\mathsf{SNR} \geq C'$ for some other absolute constant $C' > 0$.

Our algorithms are not meant for practical deployment, but instead are intended to shed light on the computational difficulty of the least squares problem and the average-case recovery problem. Indeed, note that a naïve brute-force search over permutations requires time $\Omega(n!) = n^{\Omega(n)}$, and the only other previous algorithms (already discussed above) were restricted to $d = 1$ [113] or only had some form of approximation guarantee when $d = 2$ [116]. We are not aware of previous algorithms for the average-case problem in general dimension d.[4]

Our lower bounds on SNR stand in contrast to what is achievable in the classical linear regression model (where the covariate/response correspondence is known): in that model, the SNR requirement for approximately recovering $\bar{\boldsymbol{w}}$ scales as d/n, and hence the problem becomes easier with n. The lack of correspondence thus drastically changes the difficulty of the problem.

4.3.a Approximation algorithm for the least squares problem

In this section, we consider the least squares problem from Equation (4.16). The inputs are an arbitrary matrix $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times d}$ and an arbitrary vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^\top \in \mathbb{R}^n$, and the goal is to find a vector $\boldsymbol{w} \in \mathbb{R}^d$ and permutation matrix $\boldsymbol{\Pi} \in \mathcal{P}_n$ (where $\mathcal{P}_n$ denotes the space of $n \times n$ permutation matrices[5]) to minimize $\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}^\top\boldsymbol{y}\|_2^2$. This problem is NP-hard in the case where $d = \Omega(n)$ [113] (see also Section 4.3.d). We give an approximation scheme that, for any $\varepsilon \in (0,1)$, returns a $(1+\varepsilon)$-approximation in time $(n/\varepsilon)^{O(k)} + \operatorname{poly}(n,d)$, where $k := \operatorname{rank}(\boldsymbol{X}) \le \min\{n,d\}$.

We assume without loss of generality that $\boldsymbol{X} \in \mathbb{R}^{n\times k}$ and $\boldsymbol{X}^\top\boldsymbol{X} = \boldsymbol{I}_k$. This is because we can always replace $\boldsymbol{X}$ with its matrix of left singular vectors $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, obtained via singular value decomposition $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{V}^\top\boldsymbol{V} = \boldsymbol{I}_k$ and $\boldsymbol{\Sigma} \succ 0$ is diagonal. A solution $(\boldsymbol{w}, \boldsymbol{\Pi})$ for $(\boldsymbol{U}, \boldsymbol{y})$ has the same cost as the solution $(\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{w}, \boldsymbol{\Pi})$ for $(\boldsymbol{X}, \boldsymbol{y})$, and a solution $(\boldsymbol{w}, \boldsymbol{\Pi})$ for $(\boldsymbol{X}, \boldsymbol{y})$ has the same cost as the solution $(\boldsymbol{\Sigma}\boldsymbol{V}^\top\boldsymbol{w}, \boldsymbol{\Pi})$ for $(\boldsymbol{U}, \boldsymbol{y})$.

Algorithm

Our approximation algorithm, shown as Algorithm 3, uses a careful enumeration to beat the naïve brute-force running time of $\Omega(|\mathcal{P}_n|) = \Omega(n!)$. It uses as a subroutine a "Row Sampling" algorithm of [118] (described in Section 4.3.e), which has the following property.

[4] A recent algorithm of [117] exploits a similar average-case setting but only for a somewhat easier variant of the problem where more information about the unknown correspondence is provided.

[5] Each permutation matrix $\boldsymbol{\Pi} \in \mathcal{P}_n$ corresponds to a permutation π on $[n]$; the (i,j)-th entry of $\boldsymbol{\Pi}$ is one if $\pi(i) = j$ and is zero otherwise.

Algorithm 3 Approximation algorithm for least squares problem

```
input Covariate matrix $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times k}$; response vector $\boldsymbol{y} = (y_1, y_2, \dots, y_n)^\top \in \mathbb{R}^n$; approximation parameter $\varepsilon \in (0,1)$.
assume $\boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{I}_k$.
output Weight vector $\widehat{\boldsymbol{w}} \in \mathbb{R}^k$ and permutation matrix $\widehat{\boldsymbol{\Pi}} \in \mathcal{P}_n$.
Run "Row Sampling" algorithm with input matrix $\boldsymbol{X}$ to obtain a matrix $\boldsymbol{S} \in \mathbb{R}^{r\times n}$ with $r = 4k$.
Let $\mathcal{B}$ be the set of vectors $\boldsymbol{b} = (b_1, b_2, \dots, b_n)^\top \in \mathbb{R}^n$ satisfying the following: for each $i \in [n]$,
    • if the $i$-th column of $\boldsymbol{S}$ is all zeros, then $b_i = 0$;
    • otherwise, $b_i \in \{y_1, y_2, \dots, y_n\}$.
Let $c := 1 + 4(1 + \sqrt{n/(4k)})^2$.
for each $\boldsymbol{b} \in \mathcal{B}$ do
    Compute $\tilde{\boldsymbol{w}}_{\boldsymbol{b}} \in \arg\min_{\boldsymbol{w}\in\mathbb{R}^k} \|\boldsymbol{S}(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{b})\|_2^2$, and let $r_{\boldsymbol{b}} := \min_{\boldsymbol{\Pi}\in\mathcal{P}_n} \|\boldsymbol{X}\tilde{\boldsymbol{w}}_{\boldsymbol{b}} - \boldsymbol{\Pi}^\top \boldsymbol{y}\|_2^2$.
    Construct a $\sqrt{\varepsilon r_{\boldsymbol{b}}/c}$-net $\mathcal{N}_{\boldsymbol{b}}$ for the Euclidean ball of radius $\sqrt{c r_{\boldsymbol{b}}}$ around $\tilde{\boldsymbol{w}}_{\boldsymbol{b}}$, so that for each $\boldsymbol{v} \in \mathbb{R}^k$ with $\|\boldsymbol{v} - \tilde{\boldsymbol{w}}_{\boldsymbol{b}}\|_2 \le \sqrt{c r_{\boldsymbol{b}}}$, there exists $\boldsymbol{v}' \in \mathcal{N}_{\boldsymbol{b}}$ such that $\|\boldsymbol{v} - \boldsymbol{v}'\|_2 \le \sqrt{\varepsilon r_{\boldsymbol{b}}/c}$.
end for
return $\widehat{\boldsymbol{w}} \in \arg\min_{\boldsymbol{w}\in\bigcup_{\boldsymbol{b}\in\mathcal{B}}\mathcal{N}_{\boldsymbol{b}}} \min_{\boldsymbol{\Pi}\in\mathcal{P}_n} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}^\top\boldsymbol{y}\|_2^2$ and $\widehat{\boldsymbol{\Pi}} \in \arg\min_{\boldsymbol{\Pi}\in\mathcal{P}_n} \|\boldsymbol{X}\widehat{\boldsymbol{w}} - \boldsymbol{\Pi}^\top\boldsymbol{y}\|_2^2$.
```

Theorem 18 (Specialization of Theorem 12 in [118])**.** *There is an algorithm ("Row Sampling") that, given any matrix $\boldsymbol{A} \in \mathbb{R}^{n\times k}$ with $n \ge k$, returns in $\operatorname{poly}(n,k)$ time a matrix $\boldsymbol{S} \in \mathbb{R}^{r\times n}$ with $r = 4k$ such that the following hold.*

1. *Every row of $\boldsymbol{S}$ has at most one non-zero entry.*
2. *For every $\boldsymbol{b} \in \mathbb{R}^n$, every $\boldsymbol{w}' \in \arg\min_{\boldsymbol{w}\in\mathbb{R}^k} \|\boldsymbol{S}(\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b})\|_2^2$ satisfies $\|\boldsymbol{A}\boldsymbol{w}' - \boldsymbol{b}\|_2^2 \le c \cdot \min_{\boldsymbol{w}\in\mathbb{R}^k} \|\boldsymbol{A}\boldsymbol{w} - \boldsymbol{b}\|_2^2$ for $c = 1 + 4(1 + \sqrt{n/(4k)})^2 = O(n/k)$.*

The matrix $\boldsymbol{S}$ returned by Row Sampling determines a (weighted) subset of $O(k)$ rows of $\boldsymbol{A}$ such that solving a (ordinary) least squares problem (with any right-hand side $\boldsymbol{b}$) on this subset of rows and corresponding right-hand side entries yields a $O(n/k)$-approximation to the least squares problem over all rows and right-hand side entries. Row Sampling does not directly apply to our problem because (1) it does not minimize over permutations of the right-hand side, and (2) the approximation factor is too large. However, we are able to use it to narrow the search space in our problem.

An alternative to Row Sampling is to simply enumerate all subsets of k rows of $\boldsymbol{X}$. This is justified by a recent result of [119], which shows that for any right-hand side $\boldsymbol{b} \in \mathbb{R}^n$, using "volume sampling" [120] to choose a matrix $\boldsymbol{S} \in \{0,1\}^{k \times k}$ (where each row has one non-zero entry) gives a similar guarantee as that of Row Sampling, except with the $O(n/k)$ factor replaced by $k+1$ in expectation.

Analysis

The approximation guarantee of Algorithm 3 is given in the following theorem.

Theorem 19. *Algorithm 3 returns* $\widehat{\boldsymbol{w}} \in \mathbb{R}^k$ *and* $\widehat{\boldsymbol{\Pi}} \in \mathcal{P}_n$ *satisfying*

$$\left\|\boldsymbol{X}\widehat{\boldsymbol{w}} - \widehat{\boldsymbol{\Pi}}^\top \boldsymbol{y}\right\|_2^2 \;\leq\; (1+\varepsilon) \min_{\boldsymbol{w} \in \mathbb{R}^k, \boldsymbol{\Pi} \in \mathcal{P}_n} \left\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}^\top \boldsymbol{y}\right\|_2^2 .$$

Proof. Let $\mathrm{opt} := \min_{\boldsymbol{w},\boldsymbol{\Pi}} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}^\top \boldsymbol{y}\|_2^2$ be the optimal cost, and let $(\boldsymbol{w}_\star, \boldsymbol{\Pi}_\star)$ denote a solution achieving this cost. The optimality implies that $\boldsymbol{w}_\star$ satisfies the normal equations $\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w}_\star = \boldsymbol{X}^\top \boldsymbol{\Pi}_\star^\top \boldsymbol{y}$. Observe that there exists a vector $\boldsymbol{b}_\star \in \mathcal{B}$ satisfying $\boldsymbol{S}\boldsymbol{b}_\star = \boldsymbol{S}\boldsymbol{\Pi}_\star^\top \boldsymbol{y}$. By Theorem 18 and the normal equations, the vector $\tilde{\boldsymbol{w}}_{\boldsymbol{b}_\star}$ and cost value $r_{\boldsymbol{b}_\star}$ satisfy

$$\mathrm{opt} \;\leq\; r_{\boldsymbol{b}_\star} \;\leq\; \left\|\boldsymbol{X}\tilde{\boldsymbol{w}}_{\boldsymbol{b}_\star} - \boldsymbol{\Pi}_\star^\top \boldsymbol{y}\right\|_2^2 \;=\; \left\|\boldsymbol{X}(\tilde{\boldsymbol{w}}_{\boldsymbol{b}_\star} - \boldsymbol{w}_\star)\right\|_2^2 + \mathrm{opt} \;\leq\; c \cdot \mathrm{opt} .$$

Moreover, since $\boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{I}_k$, we have that $\|\tilde{\boldsymbol{w}}_{\boldsymbol{b}_\star} - \boldsymbol{w}_\star\|_2 \leq \sqrt{(c-1)\mathrm{opt}} \leq \sqrt{c r_{\boldsymbol{b}_\star}}$. By construction of $\mathcal{N}_{\boldsymbol{b}_\star}$, there exists $\boldsymbol{w} \in \mathcal{N}_{\boldsymbol{b}_\star}$ satisfying $\|\boldsymbol{w} - \boldsymbol{w}_\star\|_2^2 = \|\boldsymbol{X}(\boldsymbol{w} - \boldsymbol{w}_\star)\|_2^2 \leq \varepsilon r_{\boldsymbol{b}_\star}/c \leq \varepsilon \mathrm{opt}$. For this $\boldsymbol{w}$, the normal equations imply

$$\min_{\boldsymbol{\Pi} \in \mathcal{P}_n} \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}^\top \boldsymbol{y}\|_2^2 \;\leq\; \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{\Pi}_\star^\top \boldsymbol{y}\|_2^2 \;=\; \|\boldsymbol{X}(\boldsymbol{w} - \boldsymbol{w}_\star)\|_2^2 + \mathrm{opt} \;\leq\; (1+\varepsilon)\mathrm{opt} .$$

Therefore, the solution returned by Algorithm 3 has cost no more than $(1+\varepsilon)\mathrm{opt}$. □

By the results of [113] for maximum likelihood estimation, our algorithm enjoys recovery guarantees for $\bar{\boldsymbol{w}}$ and $\bar{\pi}$ when the data come from the Gaussian measurement model (4.15). However, the approximation guarantee also holds for worst-case inputs without generative assumptions.

Running time. We now consider the running time of Algorithm 3. There is the initial cost for singular value decomposition (as discussed at the beginning of the section), and also for "Row Sampling"; both of these take $\operatorname{poly}(n,d)$ time. For the rest of the algorithm, we need to consider the size of $\mathcal{B}$ and the size of the net $\mathcal{N}_{\boldsymbol{b}}$ for each $\boldsymbol{b} \in \mathcal{B}$. First, we have $|\mathcal{B}| \leq n^r = n^{O(k)}$, since $\boldsymbol{S}$ has only $4k$ rows and each row has at most a single non-zero entry. Next, for each $\boldsymbol{b} \in \mathcal{B}$, we construct the δ-net $\mathcal{N}_{\boldsymbol{b}}$ (for $\delta := \sqrt{\varepsilon r_{\boldsymbol{b}}/c}$) by constructing a $\delta/\sqrt{k}$-net for the ℓ_∞-ball of radius $\sqrt{c r_{\boldsymbol{b}}}$ centered at $\tilde{\boldsymbol{w}}_{\boldsymbol{b}}$ (using an appropriate axis-aligned grid). This has size $|\mathcal{N}_{\boldsymbol{b}}| \leq (4c^2k/\varepsilon)^{k/2} = (n/\varepsilon)^{O(k)}$. Finally, each $\arg\min_{\boldsymbol{w}\in\mathbb{R}^k}$ computation takes $O(nk^2)$ time, and each $(\arg)\min_{\boldsymbol{\Pi}\in\mathcal{P}_n}$ takes $O(nk + n\log n)$ time [113] (also see Section 4.3.e). So, the overall running time is $(n/\varepsilon)^{O(k)} + \operatorname{poly}(n,d)$.

4.3.b Exact recovery algorithm in noiseless Gaussian setting

To counter the intractability of the least squares problem in (4.16) confronted in Section 4.3.a, it is natural to explore distributional assumptions that may lead to faster algorithms. In this section, we consider the noiseless measurement model where the $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ (as in [113]). We give an algorithm that exactly recovers $\bar{\boldsymbol{w}}$ with high probability when $n \geq d+1$. The algorithm runs in $\operatorname{poly}(n,d)$-time when $(\boldsymbol{x}_i)_{i=1}^n$ and $\bar{\boldsymbol{w}}$ are appropriately quantized.

It will be notationally simpler to consider $n+1$ covariate vectors and responses

$$y_i = \bar{\boldsymbol{w}}^\top \boldsymbol{x}_{\bar{\pi}(i)}\,, \quad i = 0, 1, \dots, n\,. \tag{4.17}$$

Here, $(\boldsymbol{x}_i)_{i=0}^n$ are $n+1$ i.i.d. draws from $\mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, the unknown permutation $\bar{\pi}$ is over $\{0, 1, \dots, n\}$, and the requirement of at least $d+1$ measurements is expressed as $n \geq d$.

In fact, we shall consider a variant of the problem in which we are given one of the values of the unknown permutation $\bar{\pi}$. Without loss of generality, assume we are given that $\bar{\pi}(0) = 0$. Solving this variant of the problem suffices because there are only $n+1$ possible values of $\bar{\pi}(0)$: we can

Algorithm 4 Find permutation

```
input Covariate vectors $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n$ in $\mathbb{R}^d$; response values $y_0, y_1, y_2, \dots, y_n$ in $\mathbb{R}$; confidence parameter $\delta \in (0, 1)$; lattice parameter $\beta > 0$.
assume there exists $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ and permutation $\bar{\pi}$ on $[n]$ such that $y_i = \bar{\boldsymbol{w}}^\top \boldsymbol{x}_{\bar{\pi}(i)}$ for each $i \in [n]$, and that $y_0 = \bar{\boldsymbol{w}}^\top \boldsymbol{x}_0$.
output Permutation $\widehat{\pi}$ on $[n]$ or failure.
Let $\boldsymbol{X} = [\boldsymbol{x}_1 | \boldsymbol{x}_2 | \cdots | \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$, and its pseudoinverse be $\boldsymbol{X}^\dagger = [\tilde{\boldsymbol{x}}_1 | \tilde{\boldsymbol{x}}_2 | \cdots | \tilde{\boldsymbol{x}}_n]$.
Create Subset Sum instance with $n^2$ source numbers $c_{i,j} := y_i \tilde{\boldsymbol{x}}_j^\top \boldsymbol{x}_0$ for $(i, j) \in [n] \times [n]$ and target sum $y_0$.
Run Algorithm 5 with Subset Sum instance and lattice parameter $\beta$.
if Algorithm 5 returns a solution $\mathcal{S} \subseteq [n] \times [n]$ then
    return any permutation $\widehat{\pi}$ on $[n]$ such that $\widehat{\pi}(i) = j$ implies $(i, j) \in \mathcal{S}$.
else
    return failure.
end if
```

try them all, incurring just a factor $n + 1$ in the computation time. So henceforth, we just consider $\bar{\pi}$ as an unknown permutation on $[n]$.

Algorithm

Our algorithm, shown as Algorithm 4, is based on a reduction to the Subset Sum problem. An instance of Subset Sum is specified by an unordered collection of source numbers $\{c_i\}_{i \in \mathcal{I}} \subset \mathbb{R}$, and a target sum $t \in \mathbb{R}$. The goal is to find a subset $\mathcal{S} \subseteq \mathcal{I}$ such that $\sum_{i \in \mathcal{S}} c_i = t$. Although Subset Sum is NP-hard in the worst case, it is tractable for certain structured instances [99, 121]. We prove that Algorithm 4 constructs such an instance with high probability. A similar algorithm based on such a reduction was recently used by [122] for a different but related problem.

Algorithm 4 proceeds by (i) solving a Subset Sum instance based on the covariate vectors and response values (using Algorithm 5), and (ii) constructing a permutation $\widehat{\pi}$ on $[n]$ based on the solution to the Subset Sum instance. With the permutation $\widehat{\pi}$ in hand, we (try to) find a solution $\boldsymbol{w} \in \mathbb{R}^d$ to the system of linear equations $y_i = \boldsymbol{w}^\top \boldsymbol{x}_{\widehat{\pi}(i)}$ for $i \in [n]$. If $\widehat{\pi} = \bar{\pi}$, then there is a unique such solution almost surely.

Algorithm 5 [99] subset sum algorithm

```
input Source numbers $\{c_i\}_{i \in \mathcal{I}} \subset \mathbb{R}$; target sum $t \in \mathbb{R}$; lattice parameter $\beta > 0$.
output Subset $\widehat{\mathcal{S}} \subseteq \mathcal{I}$ or failure.
Construct lattice basis $\boldsymbol{B} \in \mathbb{R}^{(|\mathcal{I}|+2)\times(|\mathcal{I}|+1)}$ where
    $$\boldsymbol{B} := \left[\begin{array}{c|c} \multicolumn{2}{c}{\boldsymbol{I}_{|\mathcal{I}|+1}} \\ \hline \beta t & -\beta c_i : i \in \mathcal{I} \end{array}\right] \in \mathbb{R}^{(|\mathcal{I}|+2)\times(|\mathcal{I}|+1)}.$$
Run basis reduction [e.g., 98] to find non-zero lattice vector $\boldsymbol{v}$ of length at most $2^{|\mathcal{I}|/2} \cdot \lambda_1(\boldsymbol{B})$.
if $\boldsymbol{v} = z(1, \boldsymbol{\chi}_{\widehat{\mathcal{S}}}^\top, 0)^\top$, with $z \in \mathbb{Z}$ and $\boldsymbol{\chi}_{\widehat{\mathcal{S}}} \in \{0,1\}^{\mathcal{I}}$ is characteristic vector for some $\widehat{\mathcal{S}} \subseteq \mathcal{I}$ then
    return $\widehat{\mathcal{S}}$.
else
    return failure.
end if
```

Analysis

The following theorem is the main recovery guarantee for Algorithm 4.

Theorem 20. *Pick any $\delta \in (0,1)$. Suppose $(\boldsymbol{x}_i)_{i=0}^n$ are i.i.d. draws from $\mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, and $(y_0)_{i=1}^n$ follow the noiseless measurement model from* (4.17) *for some $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ and permutation $\bar{\pi}$ on $[n]$ (and $\bar{\pi}(0) = 0$), and that $n \geq d$. Furthermore, suppose Algorithm 4 is run with inputs $(\boldsymbol{x}_i)_{i=0}^n$, $(y_i)_{i=0}^n$, δ, and β, and also that $\beta \geq 2^{n^2}/\varepsilon$ where ε is defined in Equation* (4.22)*. With probability at least $1 - \delta$, Algorithm 4 returns $\widehat{\pi} = \bar{\pi}$.*

Remark 21. The value of ε from Equation (4.22) is directly proportional to $\|\bar{\boldsymbol{w}}\|_2$, and Algorithm 4 requires a lower bound on ε (in the setting of the lattice parameter β). Hence, it suffices to determine a lower bound on $\|\bar{\boldsymbol{w}}\|_2$. Such a bound can be obtained from the measurement values: a standard tail bound (Lemma 4.3.7 in Section 4.3.f) shows that with high probability, $\sqrt{\sum_{i=1}^n y_i^2/(2n)}$ is a lower bound on $\|\bar{\boldsymbol{w}}\|_2$, and is within a constant factor of it as well.

Remark 22. Algorithm 4 strongly exploits the assumption of noiseless measurements, which is expected given the SNR lower bounds of [113] for recovering $\bar{\pi}$. The algorithm, however, is also very brittle and very likely fails in the presence of noise.

Remark 23. The recovery result does not contradict the results of [112], which show that a collection of $2d$ measurement vectors are necessary for recovering all $\bar{\boldsymbol{w}}$, even in the noiseless measurement model of (4.17). Indeed, our result shows that for a *fixed* $\bar{\boldsymbol{w}} \in \mathbb{R}^d$, with high probability $d+1$ measurements in the model of (4.17) suffice to permit exactly recovery of $\bar{\boldsymbol{w}}$, but this same set of measurement vectors (when $d+1 < 2d$) will fail for some other $\bar{\boldsymbol{w}}'$.

The proof of Theorem 20 is based on the following theorem—essentially due to [99] and [121]—concerning certain structured instances of Subset Sum that can be solved using the lattice basis reduction algorithm of [98]. Given a basis $\boldsymbol{B} = [\boldsymbol{b}_1 | \boldsymbol{b}_2 | \cdots | \boldsymbol{b}_k] \in \mathbb{R}^{m \times k}$ for a lattice

$$\mathcal{L}(\boldsymbol{B}) \ := \ \left\{ \sum_{i=1}^{k} z_i \boldsymbol{b}_i : z_1, z_2, \dots, z_k \in \mathbb{Z} \right\} \ \subset \ \mathbb{R}^m \,,$$

this algorithm can be used to find a non-zero vector $\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B}) \setminus \{\boldsymbol{0}\}$ whose length is at most $2^{(k-1)/2}$ times that of the shortest non-zero vector in the lattice

$$\lambda_1(\boldsymbol{B}) \ := \ \min_{\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B}) \setminus \{\boldsymbol{0}\}} \|\boldsymbol{v}\|_2 \ .$$

Theorem 24 ([99, 121])**.** *Suppose the Subset Sum instance specified by source numbers $\{c_i\}_{i \in \mathcal{I}} \subset \mathbb{R}$ and target sum $t \in \mathbb{R}$ satisfy the following properties.*

1. *There is a subset $\mathcal{S}^\star \subseteq \mathcal{I}$ such that $\sum_{i \in \mathcal{S}^\star} c_i = t$.*
2. *Define $R := 2^{|\mathcal{I}|/2} \sqrt{|\mathcal{S}^\star| + 1}$ and $\mathcal{Z}_R := \{(z_0, \boldsymbol{z}) \in \mathbb{Z} \times \mathbb{Z}^{\mathcal{I}} : 0 < z_0^2 + \sum_{i \in \mathcal{I}} z_i^2 \le R^2\}$. There exists $\varepsilon > 0$ such that $|z_0 \cdot t - \sum_{i \in \mathcal{I}} z_i \cdot c_i| \ge \varepsilon$ for each $(z_0, \boldsymbol{z}) \in \mathcal{Z}_R$ that is not an integer multiple of $(1, \boldsymbol{\chi}^\star)$, where $\boldsymbol{\chi}^\star \in \{0,1\}^{\mathcal{I}}$ is the characteristic vector for $\mathcal{S}^\star$.*

Let $\boldsymbol{B}$ be the lattice basis $\boldsymbol{B}$ constructed by Algorithm 5, and assume $\beta \ge 2^{|\mathcal{I}|/2}/\varepsilon$. Then every non-zero vector in the lattice $\Lambda(\boldsymbol{B})$ with length at most $2^{|\mathcal{I}|/2}$ times the length of the shortest non-zero vector in $\Lambda(\boldsymbol{B})$ is an integer multiple of the vector $(1, \boldsymbol{\chi}_{\mathcal{S}^\star}, 0)$, and the basis reduction algorithm of [98] returns such a non-zero vector.

The Subset Sum instance in Algorithm 4 has n^2 source numbers $\{c_{i,j} : (i,j) \in [n] \times [n]\}$ and target sum y_0. We need to show that it satisfies the two conditions of Theorem 24.

Let $\mathcal{S}_{\bar{\pi}} := \{(i, j) : \bar{\pi}(i) = j\} \subset [n]\times[n]$, and let $\bar{\mathbf{\Pi}} = (\bar{\Pi}_{i,j})_{(i,j)\in[n]\times[n]} \in \mathcal{P}_n$ be the permutation matrix with $\bar{\Pi}_{i,j} := \mathbf{1}\bar{\pi}(i) = j$ for all $(i, j) \in [n] \times [n]$. Note that $\bar{\mathbf{\Pi}}$ is the "characteristic vector" for $\mathcal{S}_{\bar{\pi}}$. Define $R := 2^{n^2/2}\sqrt{n+1}$ and

$$\mathcal{Z}_R \ := \ \left\{(z_0, \boldsymbol{Z}) \in \mathbb{Z} \times \mathbb{Z}^{n\times n} : 0 < z_0^2 + \sum_{1\leq i,j\leq n} Z_{i,j}^2 \leq R^2\right\} .$$

A crude bound shows that $|\mathcal{Z}_R| \leq 2^{O(n^4)}$.

The following lemma establishes the first required property in Theorem 24.

Lemma 4.3.1. *The random matrix $\boldsymbol{X}$ has rank d almost surely, and the subset $\mathcal{S}_{\bar{\pi}}$ satisfies $y_0 = \sum_{(i,j)\in\mathcal{S}_{\bar{\pi}}} c_{i,j}$.*

Proof. That $\boldsymbol{X}$ has rank d almost surely follows from the fact that the probability density of $\boldsymbol{X}$ is supported on all of $\mathbb{R}^{n\times d}$. This implies that $\boldsymbol{X}^\dagger\boldsymbol{X} = \sum_{j=1}^{n} \tilde{\boldsymbol{x}}_j\boldsymbol{x}_j^\top = \boldsymbol{I}_d$, and

$$y_0 \ = \ \sum_{j=1}^{n} \boldsymbol{x}_0^\top\tilde{\boldsymbol{x}}_j\boldsymbol{x}_j^\top\bar{\boldsymbol{w}} \ = \ \sum_{1\leq i,j\leq n} \boldsymbol{x}_0^\top\tilde{\boldsymbol{x}}_j \cdot y_i \cdot \mathbf{1}\bar{\pi}(i) = j \ = \ \sum_{1\leq i,j\leq n} c_{i,j} \cdot \mathbf{1}\bar{\pi}(i) = j \, . \qquad \square$$

The next lemma establishes the second required property in Theorem 24. Here, we use the fact that the Frobenius norm $\left\|z_0\bar{\mathbf{\Pi}} - \boldsymbol{Z}\right\|_F$ is at least one whenever $(z_0, \boldsymbol{Z}) \in \mathbb{Z} \times \mathbb{Z}^{n\times n}$ is not an integer multiple of $(1, \bar{\mathbf{\Pi}})$.

Lemma 4.3.2. *Pick any $\eta, \eta' > 0$ such that $3|\mathcal{Z}_R|\,\eta+\eta' < 1$. With probability at least $1-3|\mathcal{Z}_R|\,\eta-\eta'$, every $(z_0, \boldsymbol{Z}) \in \mathcal{Z}_R$ with $\boldsymbol{Z} = (Z_{i,j})_{(i,j)\in[n]\times[n]}$ satisfies*

$$\left|z_0 \cdot y_0 - \sum_{i,j} Z_{i,j} \cdot c_{i,j}\right| \ \geq \ \frac{(\pi/4) \cdot \sqrt{(d-1)/n} \cdot \eta^{2+\frac{1}{d-1}}}{\left(\sqrt{n} + \sqrt{d} + \sqrt{2\ln(1/\eta')}\right)^2} \cdot \left\|z_0\bar{\mathbf{\Pi}} - \boldsymbol{Z}\right\|_F \cdot \|\bar{\boldsymbol{w}}\|_2 \, .$$

Proof. By Lemma 4.3.1, the matrix $\bar{\mathbf{\Pi}}$ satisfies $y_0 = \sum_{i,j} \bar{\Pi}_{i,j} \cdot c_{i,j}$. Fix any $(z_0, \boldsymbol{Z}) \in \mathcal{Z}_R$ with $\boldsymbol{Z} = (Z_{i,j})_{(i,j)\in[n]\times[n]}$. Then

$$z_0 \cdot y_0 - \sum_{i,j} Z_{i,j} \cdot c_{i,j} \ = \ \sum_{i,j}(z_0 \cdot \bar{\Pi}_{i,j} - Z_{i,j}) \cdot \boldsymbol{x}_0^\top\tilde{\boldsymbol{x}}_j \cdot \bar{\boldsymbol{w}}^\top\boldsymbol{x}_{\bar{\pi}(i)} \, .$$

Using matrix and vector notations, this can be written compactly as the inner product $\boldsymbol{x}_0^\top(\boldsymbol{X}^\dagger(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}})$. Since $\boldsymbol{x}_0 \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and is independent of $\boldsymbol{X}$, the distribution of the inner product is normal with mean zero and standard deviation equal to $\|\boldsymbol{X}^\dagger(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\|_2$. By Lemma 4.3.8 (in Section 4.3.f), with probability at least $1-\eta$,

$$\left|\boldsymbol{x}_0^\top\left(\boldsymbol{X}^\dagger(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\right)\right| \;\geq\; \|\boldsymbol{X}^\dagger(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{\frac{\pi}{2}} \cdot \eta\,. \tag{4.18}$$

Observe that $\boldsymbol{X}^\dagger = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top$ since $\boldsymbol{X}$ has rank d by Lemma 4.3.1, so

$$\|\boldsymbol{X}^\dagger(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\|_2 \;\geq\; \frac{\|\boldsymbol{X}^\top(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\|_2}{\|\boldsymbol{X}\|_2^2}\,. \tag{4.19}$$

By Lemma 4.3.5 (in Section 4.3.f), with probability at least $1-\eta'$,

$$\|\boldsymbol{X}\|_2^2 \;\leq\; \left(\sqrt{n}+\sqrt{d}+\sqrt{2\ln(1/\eta')}\right)^2\,. \tag{4.20}$$

And by Lemma 4.3.10 (in Section 4.3.f), with probability at least $1-2\eta$,

$$\|\boldsymbol{X}^\top(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\boldsymbol{X}\bar{\boldsymbol{w}}\|_2 \;\geq\; \left\|(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\right\|_F \cdot \|\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{\frac{(d-1)\pi}{8n}} \cdot \eta^{1+1/(d-1)}\,. \tag{4.21}$$

Since $\bar{\boldsymbol{\Pi}}$ is orthogonal, we have that $\|(z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z})^\top\bar{\boldsymbol{\Pi}}\|_F = \|z_0\bar{\boldsymbol{\Pi}}-\boldsymbol{Z}\|_F$. Combining this with (4.18), (4.19), (4.20), and (4.21), and union bounds over all $(z_0, \boldsymbol{Z}) \in \mathcal{Z}_R$ proves the claim. □

Proof of Theorem 20. Lemma 4.3.1 and Lemma 4.3.2 (with $\eta' := \delta/2$ and $\eta := \delta/(6|\mathcal{Z}_R|)$) together imply that with probability at least $1-\delta$, the source numbers $\{c_{i,j} : (i,j) \in [n]\times[n]\}$ and target sum y_0 satisfy the conditions of Theorem 24 with

$$\begin{aligned} \mathcal{S}^\star &:= \{(i,j) \in [n] \times [n] : \bar{\pi}(i) = j\} \ , \\ \varepsilon &:= \frac{(\pi/4) \cdot \sqrt{(d-1)/n} \cdot (\delta/(6|\mathcal{Z}_R|))^{2+\frac{1}{d-1}}}{\left(\sqrt{n} + \sqrt{d} + \sqrt{2\ln(2/\delta)}\right)^2} \cdot \|\bar{\boldsymbol{w}}\|_2 \ \geq \ 2^{-\mathrm{poly}(n, \log(1/\delta))} \cdot \|\bar{\boldsymbol{w}}\|_2 \ . \end{aligned} \tag{4.22}$$

Thus, in this event, Algorithm 5 (with β satisfying $\beta \geq 2^{n^2/2}/\varepsilon$) returns $\widehat{\mathcal{S}} = \mathcal{S}^\star$, which uniquely determines the permutation $\widehat{\pi} = \bar{\pi}$ returned by Algorithm 4. □

Running time. The basis reduction algorithm of [98] is iterative, with each iteration primarily consisting of Gram-Schmidt orthogonalization and another efficient linear algebraic process called "size reduction". The total number of iterations required is

$$O\left(\frac{k(k+1)}{2} \log\left(\sqrt{k} \cdot \frac{\max_{i\in[k]} \|\boldsymbol{b}_i\|_2}{\lambda_1(\boldsymbol{B})}\right)\right) \ .$$

In our case, $k = n^2$ and $\lambda_1(\boldsymbol{B}) = \sqrt{n+1}$; and by Lemma 4.3.11 (in Section 4.3.f), each of the basis vectors constructed has squared length at most $1 + \beta^2 \cdot \mathrm{poly}(d, \log(n), 1/\delta) \cdot \|\bar{\boldsymbol{w}}\|_2^2$. Using the tight setting of β required in Theorem 20, this gives a $\mathrm{poly}(n, d, \log(1/\delta))$ bound on the total number of iterations as well as on the total running time.

However, the basis reduction algorithm requires both arithmetic and rounding operations, which are typically only available for finite precision rational inputs. Therefore, a formal running time analysis would require the idealized real-valued covariate vectors $(\boldsymbol{x}_i)_{i=0}^n$ and unknown target vector $\bar{\boldsymbol{w}}$ to be quantized to finite precision values. This is doable, and is similar to using a discretized Gaussian distribution for the distribution of the covariate vectors (and assuming $\bar{\boldsymbol{w}}$ is a vector of finite precision values), but leads to a messier analysis incomparable to the setup of previous works. Nevertheless, it would be desirable to find a different algorithm that avoids lattice basis reduction that still works with just $d+1$ measurements.

4.3.c Lower bounds on signal-to-noise for approximate recovery

In this section, we consider the measurement model from (4.15) where $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from either $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$ or the uniform distribution on $[-1/2, 1/2]^d$, and $(\varepsilon_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(0, \sigma^2)$. We establish lower bounds on the signal-to-noise ratio (SNR),

$$\mathsf{SNR} \;=\; \frac{\|\bar{\boldsymbol{w}}\|_2^2}{\sigma^2}\,,$$

required by any estimator $\widehat{\boldsymbol{w}} = \widehat{\boldsymbol{w}}((\boldsymbol{x}_i)_{i=1}^n, (y_i)_{i=1}^n)$ for $\bar{\boldsymbol{w}}$ to approximately recover $\bar{\boldsymbol{w}}$ in expectation. The estimators may have *a priori* knowledge of the values of $\|\bar{\boldsymbol{w}}\|_2$ and σ^2.

Theorem 25. *Assume $(\varepsilon_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(0, \sigma^2)$.*

1. *There is an absolute constant $C > 0$ such that the following holds. If $n \geq 3$, $d \geq 22$, $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$, $(y_i)_{i=1}^n$ follow the measurement model from (4.15), and*

$$\mathsf{SNR} \;\leq\; C \cdot \min\left\{\frac{d}{\log\log(n)}, 1\right\}\,,$$

   then for any estimator $\widehat{\boldsymbol{w}}$, there exists some $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ such that

$$\mathbf{E}\left[\|\widehat{\boldsymbol{w}} - \bar{\boldsymbol{w}}\|_2\right] \;\geq\; \frac{1}{24}\|\bar{\boldsymbol{w}}\|_2\,.$$

2. *If $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from the uniform distribution on $[-1/2, 1/2]^d$, and $(y_i)_{i=1}^n$ follow the measurement model from (4.15), and*

$$\mathsf{SNR} \;\leq\; 2\,,$$

   then for any estimator $\widehat{\boldsymbol{w}}$, there exists some $\bar{\boldsymbol{w}} \in \mathbb{R}^d$ such that

$$\mathbf{E}\left[\|\widehat{\boldsymbol{w}} - \bar{\boldsymbol{w}}\|_2\right] \;\geq\; \frac{1}{2}\left(1 - \frac{1}{\sqrt{2}}\right)\|\bar{\boldsymbol{w}}\|_2\,.$$

Note that in the classical linear regression model where $y_i = \bar{\boldsymbol{w}}^\top \boldsymbol{x}_i + \varepsilon_i$ for $i \in [n]$, the maximum likelihood estimator $\widehat{\boldsymbol{w}}_{\mathsf{mle}}$ satisfies $\mathbf{E}\|\widehat{\boldsymbol{w}}_{\mathsf{mle}} - \bar{\boldsymbol{w}}\|_2 \leq C\sigma\sqrt{d/n}$, where $C > 0$ is an absolute constant. Therefore, the SNR requirement to approximately recover $\bar{\boldsymbol{w}}$ up to (say) Euclidean distance $\|\bar{\boldsymbol{w}}\|_2 / 24$ is $\mathsf{SNR} \geq 24^2 Cd/n$. Compared to this setting, Theorem 25 implies that with the measurement model of (4.15), the SNR requirement (as a function of n) is at substantially higher ($d/\log\log(n)$ in the normal covariate case, or a constant not even decreasing with n in the uniform covariate case).

For the normal covariate case, [113] show that if $n > d$, $\varepsilon < \sqrt{n}$, and

$$\mathsf{SNR} \geq n^{c\cdot\frac{n}{n-d}+\varepsilon},$$

then the maximum likelihood estimator $(\widehat{\boldsymbol{w}}_{\mathsf{mle}}, \widehat{\pi}_{\mathsf{mle}})$ (i.e., any minimizer of (4.16)) satisfies $\widehat{\pi}_{\mathsf{mle}} = \bar{\pi}$ with probability at least $1 - c'n^{-2\varepsilon}$. (Here, $c > 0$ and $c' > 0$ are absolute constants.) It is straightforward to see that, on the same event, we have $\|\widehat{\boldsymbol{w}}_{\mathsf{mle}} - \bar{\boldsymbol{w}}\|_2 \leq C\sigma\sqrt{d/n}$ for some absolute constant $C > 0$. Therefore, the necessary and sufficient conditions on SNR for approximate recovery of $\bar{\boldsymbol{w}}$ lie between $C'd/\log\log(n)$ and $n^{C''}$ (for absolute constants $C', C'' > 0$). Narrowing this range remains an interesting open problem.

A sketch of the proof in the normal covariate case is as follows. Without loss of generality, we restrict attention to the case where $\bar{\boldsymbol{w}}$ is a unit vector. We construct a $1/\sqrt{2}$-packing of the unit sphere in $\mathbb{R}^d$; the target $\bar{\boldsymbol{w}}$ will be chosen from from this set. Observe that for any distinct $\boldsymbol{u}, \boldsymbol{u}' \in U$, each of $(\boldsymbol{x}_i^\top \boldsymbol{u})_{i=1}^n$ and $(\boldsymbol{x}_i^\top \boldsymbol{u}')_{i=1}^n$ is an i.i.d. sample from $\mathrm{N}(0,1)$ of size n; we prove that they therefore determine empirical distributions that are close to each other in Wasserstein-2 distance with high probability. We then prove that conditional on this event, the resulting distributions of $(y_i)_{i=1}^n$ under $\bar{\boldsymbol{x}} = \boldsymbol{u}$ and $\bar{\boldsymbol{x}} = \boldsymbol{u}'$ (for any pair $\boldsymbol{u}, \boldsymbol{u}' \in U$) are close in Kullback-Leibler divergence. Hence, by (a generalization of) Fano's inequality [see, e.g., 123], no estimator can determine the correct $\boldsymbol{u} \in U$ with high probability.

The proof for the uniform case is similar, using $U = \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}$ where $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$. The full proof of Theorem 25 is given in Section 4.3.g.

4.3.d Strong NP-hardness of the least squares problem

For a vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ and a permutation π on $[n]$, let $\boldsymbol{b}_\pi := (b_{\pi(1)}, b_{\pi(2)}, \ldots, b_{\pi(n)})^\top$.

Recall that in the 3-PARTITION problem, the input is $d = 3k$ integers $z_1, z_2, \ldots, z_d \in \mathbb{Z}$ that sum to Ck and satisfy $C/4 < z_i < C/2$ for all $i \in [d]$, and the problem is to decide if there is a partition of $[d]$ into k subsets $S_1, S_2, \ldots, S_k \subseteq [d]$ such that $|S_j| = 3$ and $\sum_{i \in S_j} z_i = C$ for each $j \in [k]$. 3-PARTITION is NP-complete in the strong sense of [115, Section 4.2.2].

The PERMUTED LINEAR SYSTEM problem (also considered by [113]) is defined as follows. The input is a matrix $\boldsymbol{A} \in \mathbb{Z}^{n \times d}$, and a vector $\boldsymbol{b} \in \mathbb{Q}^n$. The problem is to decide if there exist a vector $\boldsymbol{x} \in \mathbb{Q}^d$ and a permutation π on $[n]$ such that $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}_\pi$.

Proposition 4.3.3. PERMUTED LINEAR SYSTEM *is strongly NP-complete.*

Because PERMUTED LINEAR SYSTEM is equivalent to deciding if the optimal value of the least squares problem from (4.16) is zero, Lemma 4.3.3 implies that the least squares problem from (4.16) is strongly NP-hard.

Proof of Lemma 4.3.3. It is clear that PERMUTED LINEAR SYSTEM is in NP. We give an efficient reduction from 3-PARTITION to PERMUTED LINEAR SYSTEM. Given an instance $z_1, z_2, \ldots, z_d$

of 3-PARTITION, we construct the matrix $\boldsymbol{A} \in \mathbb{Z}^{n \times d}$ and vector $\boldsymbol{b} \in \mathbb{Z}^n$ with $n = d + k$ as follows:

$$\boldsymbol{A} := \left[\begin{array}{cccccccccc} 1 &&&&&&&&& \\ & 1 &&&&&&&& \\ && 1 &&&&&&& \\ &&& 1 &&&&&& \\ &&&& 1 &&&&& \\ &&&&& 1 &&&& \\ &&&&&& \ddots &&& \\ &&&&&&& 1 && \\ &&&&&&&& 1 & \\ &&&&&&&&& 1 \\ \hline 1 & 1 & 1 &&&&&&& \\ &&& 1 & 1 & 1 &&&& \\ &&&&&& \ddots &&& \\ &&&&&&& 1 & 1 & 1 \end{array}\right] , \quad \boldsymbol{b} := \left[\begin{array}{c} z_1 \\ z_2 \\ \\ \\ \vdots \\ \\ \\ z_d \\ \hline C \\ C \\ \vdots \\ C \end{array}\right] .$$

The system of equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}_\pi$ has a solution if and only if

$$b_{\pi(3j-2)} + b_{\pi(3j-1)} + b_{\pi(3j)} \ = \ C \, , \quad j \in [k] \, .$$

Any permutation π on $[n]$ satisfying these equations must satisfy the following two properties:

1. $\pi([d]) = [d]$.

   This holds because for $i > d$, we have $b_i = C$, and adding such b_i to any other $b_{i'}$ and $b_{i''}$ gives a sum larger than C.

2. $z_{\pi(3j-2)} + z_{\pi(3j-1)} + z_{\pi(3j)} = C$ for each $j \in [k]$.

   This holds because since $b_i = z_i$ for $i \in [d]$.

Any permutation π on $[n]$ with the two properties shown above gives k subsets $S_j = \{\pi(3j-2), \pi(3j-1), \pi(3j)\}$ for $j \in [k]$ such that $\sum_{i \in S_j} z_i = C$. $\square$

4.3.e Additional details for approximation algorithm

This section provides some additional details on subroutines used in Algorithm 3.

Row sampling. First, we give the details of the "Row Sampling" algorithm of [118] used in Section 4.3.a. The pseudocode is presented as Algorithm 6, and uses the following notations:

- For each $i \in [n]$, $\boldsymbol{e}_i$ is the i-th coordinate basis vector in $\mathbb{R}^n$.
- $L(\boldsymbol{x}, \delta_L, \boldsymbol{A}, \ell) \;:=\; \dfrac{\boldsymbol{x}^\top (\boldsymbol{A} - (\ell + \delta_L)\boldsymbol{I}_k)^{-2} \boldsymbol{x}}{\phi(\ell + \delta_L, \boldsymbol{A}) - \Phi(\ell, \boldsymbol{A})} - (\ell + \delta_L)\boldsymbol{I}_k)^{-1}\boldsymbol{x}\,,$

  where $\phi(\ell, \boldsymbol{A}) := \sum_{i=1}^{k} \frac{1}{\lambda_i(\boldsymbol{A}) - \ell}$ and $(\lambda_i(\boldsymbol{A}))_{i=1}^{k}$ are the eigenvalues of $\boldsymbol{A}$.
- $\widehat{U}(\boldsymbol{x}, \delta, \boldsymbol{B}, u) \;:=\; \dfrac{\boldsymbol{x}^\top (\boldsymbol{B} - u'\boldsymbol{I}_r)^{-2} \boldsymbol{x}}{\phi'(u, \boldsymbol{B}) - \phi'(u', \boldsymbol{B})} - \boldsymbol{x}^\top (\boldsymbol{B} - u'\boldsymbol{I}_r)^{-1}\boldsymbol{x}\,,$

  where $u' = u + \delta$ and $\phi'(u, \boldsymbol{B}) := \sum_{i=1}^{r} \frac{1}{u - \lambda_i(\boldsymbol{B})}$ and $(\lambda_i(\boldsymbol{B}))_{i=1}^{k}$ are the eigenvalues of $\boldsymbol{B}$.

Algorithm 6 "Row Sampling" algorithm of [118]

```
input  Matrix $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times k}$ such that $\boldsymbol{X}^\top\boldsymbol{X} = \boldsymbol{I}_k$; integer $r \geq k$.
output  Matrix $\boldsymbol{S} = (S_{i,j})_{(i,j) \in [r] \times [n]} \in \mathbb{R}^{r \times n}$.
Set $\boldsymbol{A}_0 = \boldsymbol{0}_{k \times k}$, $\boldsymbol{B}_0 = \boldsymbol{0}_{n \times n}$, $\boldsymbol{S} = \boldsymbol{0}_{r \times n}$, $\delta = (1 + n/r)(1 - \sqrt{k/r})^{-1}$ and $\delta_L = 1$.
for $\tau = 0$ to $r - 1$ do
    Let $\ell_\tau = \tau - \sqrt{rk}$ and $u_\tau = \delta(\tau + \sqrt{nr})$.
    Select $i_\tau \in [n]$ and number $t_\tau > 0$ such that $\widehat{U}(\boldsymbol{e}_{i_\tau}, \delta, \boldsymbol{B}_\tau, u_\tau) \leq \frac{1}{t_\tau} \leq L(\boldsymbol{x}_{i_\tau}, \delta_L, \boldsymbol{A}_\tau, \ell_\tau)$.
    Set $\boldsymbol{A}_{\tau+1} = \boldsymbol{A}_\tau + t_\tau \boldsymbol{x}_{i_\tau}\boldsymbol{x}_{i_\tau}^\top$, $\boldsymbol{B}_{\tau+1} = \boldsymbol{B}_\tau + t_\tau \boldsymbol{e}_{i_\tau}\boldsymbol{e}_{i_\tau}^\top$ and $S_{\tau+1, i_\tau} = \sqrt{r^{-1}(1 - \sqrt{k/r})}/\sqrt{t_\tau}$.
end for
return $\boldsymbol{S}$.
```

One may also consider using levarage score sampling (i.e., sample a row of $\boldsymbol{X}$ proportional to its squared length) instead of this Row Sampling algorithm. This would work, but would require selecting $O(k \log k)$ rows as opposed to just $O(k)$ [124]; this leads to an overall running time of $(n/\varepsilon)^{O(k \log k)} + \mathrm{poly}(n, d)$. Finally, as already mentioned in Section 4.3.a, it also suffices to simply enumerate all $\binom{n}{k}$ subsets of k rows of $\boldsymbol{X}$. This is slower than Algorithm 6 but yields a better approximation guarantee (specifically, the factor c from Theorem 18 can be replaced by $k+1$ on account of a result of [119]). However, the overall approximation guarantee and asymptotic running time of Algorithm 3 is the same.

One-dimensional permutation problem. Next, we explain how to solve the optimization problem

$$\min_{\mathbf{\Pi} \in \mathcal{P}_n} \left\| \boldsymbol{a} - \mathbf{\Pi}^\top \boldsymbol{b} \right\|_2^2$$

for any given $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$. Let $(a_{(i)})_{i=1}^n$ denote the non-decreasing ordering $a_{(1)} \le a_{(2)} \le \cdots \le a_{(n)}$ of the entries of $\boldsymbol{a}$, and let $(b_{(i)})_{i=1}^n$ be analogously defined. By Lemma 4.3.12, we have

$$\min_{\mathbf{\Pi} \in \mathcal{P}_n} \left\| \boldsymbol{a} - \mathbf{\Pi}^\top \boldsymbol{b} \right\|_2^2 = \sum_{i=1}^n \left(a_{(i)} - b_{(i)} \right)^2 .$$

Hence, if $\mathbf{\Pi}_{\boldsymbol{a}}$ (respectively, $\mathbf{\Pi}_{\boldsymbol{b}}$) is the permutation matrix that rearranges the entires of $\boldsymbol{a}$ (respectively, $\boldsymbol{b}$) in non-decreasing order, then

$$\sum_{i=1}^n \left(a_{(i)} - b_{(i)} \right)^2 = \left\| \mathbf{\Pi}_{\boldsymbol{a}} \boldsymbol{a} - \mathbf{\Pi}_{\boldsymbol{b}} \boldsymbol{b} \right\|_2^2 = \left\| \mathbf{\Pi}_{\boldsymbol{a}}^\top \left(\mathbf{\Pi}_{\boldsymbol{a}} \boldsymbol{a} - \mathbf{\Pi}_{\boldsymbol{b}} \boldsymbol{b} \right) \right\|_2^2 = \left\| \boldsymbol{a} - \mathbf{\Pi}_{\boldsymbol{a}}^\top \mathbf{\Pi}_{\boldsymbol{b}} \boldsymbol{b} \right\|_2^2 ,$$

where the second and third equalities use the fact that permutation matrices are orthogonal. Thus, the minimizing permutation matrix is $\mathbf{\Pi} = \mathbf{\Pi}_{\boldsymbol{b}}^\top \mathbf{\Pi}_{\boldsymbol{a}}$. This can be found by sorting the entries of $\boldsymbol{a}$ and of $\boldsymbol{b}$ in $O(n \log n)$ time.

4.3.f Probability inequalities

This section collects several probability inequalities used in the analysis of Algorithm 4. Let $\sigma_i(\boldsymbol{M})$ denote the i-th largest singular value of the matrix $\boldsymbol{M}$.

Extreme singular values of Gaussian random matrices.

Lemma 4.3.4 (Eq. 3.2 in [125]). *Let $\boldsymbol{A}$ be an $n \times d$ matrix whose entries are i.i.d.* $\mathrm{N}(0,1)$ *random* *variables and* $n \geq d$. *For any* $\eta \in (0,1)$,

$$\mathbf{Pr}\left(\sigma_d(\boldsymbol{A}) \leq \frac{\eta}{\sqrt{d}}\right) \;\leq\; \eta\,.$$

Lemma 4.3.5 (Theorem II.13 in [110])**.** *Let* $\boldsymbol{A}$ *be an* $n \times d$ *matrix whose entries are i.i.d.* $\mathrm{N}(0,1)$ *random variables. For any* $\eta \in (0,1)$,

$$\mathbf{Pr}\left(\sigma_1(\boldsymbol{A}) \geq \sqrt{n} + \sqrt{d} + \sqrt{2\ln(1/\eta)}\right) \;\leq\; \eta\,.$$

Tail bounds for Gaussian and χ^2 random variables.

Lemma 4.3.6. *Let* $Z \sim \mathrm{N}(0,1)$. *For any* $\eta \in (0,1)$, $\mathbf{Pr}(Z^2 \geq 2\ln(2/\delta)) \leq \eta$.

Proof. This follows from the standard Chernoff bounding method. □

Lemma 4.3.7 (Lemma 1 in [126])**.** *Let* $W \sim \chi^2_k$. *For any* $\eta \in (0,1)$, $\mathbf{Pr}(W \geq k + 2\sqrt{k\ln(1/\eta)} + 2\ln(1/\eta)) \leq \eta$.

Anti-concentration bounds for Gaussian and χ^2 random variables.

Lemma 4.3.8. *Let* $Z \sim \mathrm{N}(0,1)$. *For any* $\eta \in (0,1)$, $\mathbf{Pr}(Z^2 \leq \pi\eta^2/2) \leq \eta$.

Proof. This follows from direct integration. □

Lemma 4.3.9 (Lemma 9 in [113])**.** *Let* $W \sim \chi^2_k$. *For any* $\eta \in (0,1)$, $\mathbf{Pr}(W \leq k\eta^{2/k}/4) \leq \eta$.

Lemma 4.3.10. *Let* $\boldsymbol{x} \in \mathbb{R}^d$ *be any vector,* $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ *be any matrix, and* $\boldsymbol{A}$ *a random* $n \times d$ *matrix of i.i.d.* $\mathrm{N}(0,1)$ *random variables. For any* $\eta \in \big(0, 1/2\big)$,

$$\mathbf{Pr}\left(\|\boldsymbol{A}^\top \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}\|_2 \;\leq\; \|\boldsymbol{M}\|_F \cdot \|\boldsymbol{x}\|_2 \cdot \sqrt{\frac{(d-1)\pi}{8n}} \cdot \eta^{1+1/(d-1)}\right) \;\leq\; 2\eta\,.$$

Proof. Let $\boldsymbol{u}_1 := \boldsymbol{x}/\|\boldsymbol{x}\|_2$, and extend to an orthonormal basis $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_d$ for $\mathbb{R}^d$. Let $\boldsymbol{g}_i :=$

$\boldsymbol{A}\boldsymbol{u}_i$ for each $i \in [d]$, so $\boldsymbol{g}_1, \boldsymbol{g}_2, \dots, \boldsymbol{g}_d$ are i.i.d. $\mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_n)$ random vectors. We first show that

$$\mathbf{Pr}\left(\|\boldsymbol{M}\boldsymbol{g}_1\|_2 \;\leq\; \|\boldsymbol{M}\|_F \cdot \sqrt{\frac{\pi}{2n}} \cdot \eta \right) \;\leq\; \eta \,. \tag{4.23}$$

To see this, note that the distribution of $\|\boldsymbol{M}\boldsymbol{g}_1\|_2^2$ is the same as that of $\sum_{i=1}^n \sigma_i(\boldsymbol{M})^2 \cdot Z_i^2$, where $Z_1, Z_2, \dots, Z_n$ are i.i.d. $\mathrm{N}(0,1)$ random variables. Therefore, Lemma 4.3.8 and the fact $\|\boldsymbol{M}\|_2^2 \geq \|\boldsymbol{M}\|_F^2 / n$ proves the claim in (4.23).

Next, observe that

$$\begin{aligned} \|\boldsymbol{A}^\top \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}\|_2^2 &= \|\boldsymbol{x}\|_2^2 \cdot \boldsymbol{u}_1^\top \boldsymbol{A}^\top \boldsymbol{M}^\top \boldsymbol{A} \left(\sum_{i=1}^d \boldsymbol{u}_i \boldsymbol{u}_i^\top \right) \boldsymbol{A}^\top \boldsymbol{M}\boldsymbol{A}\boldsymbol{u}_1 \\ &= \|\boldsymbol{x}\|_2^2 \cdot \boldsymbol{g}_1^\top \boldsymbol{M}^\top \left(\sum_{i=1}^d \boldsymbol{g}_i \boldsymbol{g}_i^\top \right) \boldsymbol{M}\boldsymbol{g}_1 \\ &\geq \|\boldsymbol{x}\|_2^2 \cdot \sum_{i=2}^d \left(\boldsymbol{g}_i^\top \boldsymbol{M}\boldsymbol{g}_1 \right)^2 \,. \end{aligned} \tag{4.24}$$

Conditional on $\boldsymbol{g}_1$, the final right-hand side in (4.24) has the same distribution as $\|\boldsymbol{x}\|_2^2 \cdot \|\boldsymbol{M}\boldsymbol{g}_1\|_2^2 \cdot W$, where $W \sim \chi_{d-1}^2$ is a chi-squared random variable with $d-1$ degrees of freedom. Therefore, Lemma 4.3.9 implies

$$\mathbf{Pr}\left(\|\boldsymbol{A}^\top \boldsymbol{M}\boldsymbol{A}\boldsymbol{x}\|_2 \;\leq\; \|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{M}\boldsymbol{g}_1\|_2 \cdot \frac{\sqrt{d-1}}{2} \cdot \eta^{1/(d-1)} \right) \;\leq\; \eta \,.$$

Combining this inequality with the inequality from (4.23) and a union bound proves the claim. □

Lattice basis size. The following lemma is used to bound the size of the lattice basis vectors constructed by Algorithm 4 (via Algorithm 5). Recall that there are $n^2 + 1$ basis vectors; one has length $\sqrt{1 + \beta^2 y_0^2}$, and the remaining n^2 have length $\sqrt{1 + \beta^2 c_{i,j}^2}$ for $(i,j) \in [n] \times [n]$.

Lemma 4.3.11. *For any* $\eta \in \big(0, 1/5\big)$*, with probability at least* $1 - 5\eta$*,*

$$\begin{aligned}
|y_0| \; &\leq \; \|\bar{\boldsymbol{w}}\|_2 \sqrt{2\ln(2/\eta)} \, , \\
\big|c_{i,j}\big| \; &\leq \; \|\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{2\ln(2n/\eta)} \cdot \frac{d}{\eta^2} \cdot \sqrt{d + 2\sqrt{d\ln(n/\eta)} + 2\ln(n/\eta)} \cdot \sqrt{2\ln(2n/\eta)}
\end{aligned}$$

where $(i, j) \in [n] \times [n]$*.*

Proof. By Lemma 4.3.4, Lemma 4.3.6, and Lemma 4.3.7, with probability at least $1 - 5\eta$,

$$\begin{aligned}
\big\|(\boldsymbol{X}^\top \boldsymbol{X})^{-1}\big\|_2 \; &\leq \; \frac{d}{\eta^2} \, , \\
|\boldsymbol{x}_0^\top \bar{\boldsymbol{w}}| \; &\leq \; \|\bar{\boldsymbol{w}}\|_2 \sqrt{2\ln(2/\eta)} \, , \\
|\boldsymbol{x}_{\bar{\pi}(i)}^\top \bar{\boldsymbol{w}}| \; &\leq \; \|\bar{\boldsymbol{w}}\|_2 \sqrt{2\ln(2n/\eta)} \, , \quad i \in [n] \, , \\
|\tilde{\boldsymbol{x}}_j^\top \boldsymbol{x}_0| \; &\leq \; \|\tilde{\boldsymbol{x}}_j\|_2 \sqrt{2\ln(2n/\eta)} \, , \quad j \in [n] \, , \\
\|\boldsymbol{x}_j\|_2 \; &\leq \; \sqrt{d + 2\sqrt{d\ln(n/\eta)} + 2\ln(n/\eta)} \, , \quad j \in [n] \, .
\end{aligned}$$

In this event, we have for each $(i, j) \in [n] \times [n]$,

$$\begin{aligned}
\big|c_{i,j}\big| \; &= \; |\boldsymbol{x}_{\bar{\pi}(i)}^\top \bar{\boldsymbol{w}}| \cdot |\tilde{\boldsymbol{x}}_j^\top \boldsymbol{x}_0| \\
&\leq \; \|\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{2\ln(2n/\eta)} \cdot \|\boldsymbol{X}^\dagger \boldsymbol{e}_j\|_2 \cdot \sqrt{2\ln(2n/\eta)} \\
&= \; \|\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{2\ln(2n/\eta)} \cdot \|(\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{e}_j\|_2 \cdot \sqrt{2\ln(2n/\eta)} \\
&\leq \; \|\bar{\boldsymbol{w}}\|_2 \cdot \sqrt{2\ln(2n/\eta)} \cdot \frac{d}{\eta^2} \cdot \sqrt{d + 2\sqrt{d\ln(n/\eta)} + 2\ln(n/\eta)} \cdot \sqrt{2\ln(2n/\eta)} \, ,
\end{aligned}$$

and $|y_0| \leq \|\bar{\boldsymbol{w}}\|_2 \sqrt{2\ln(2/\eta)}$. □

4.3.g Proof of signal-to-noise lower bounds

This section provides the proof of Theorem 25.

Below, for any vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)^\top$, we use the notation $(a_{(i)})_{i=1}^n$ to denote the non-decreasing ordering $a_{(1)} \leq a_{(2)} \leq \cdots \leq a_{(n)}$ of its entries, and $(\boldsymbol{a})^\uparrow := (a_{(1)}, a_{(2)}, \ldots, a_{(n)})^\top$ to denote the vector of the entries in this order.

We use the following representation for the Kantorovich transport distance with respect to Euclidean metric (i.e., Wasserstein-2 distance, denoted by W_2).

Lemma 4.3.12 (Lemma 4.1 in [127])**.** *Let μ_n be the empirical measure on $a_1, a_2, \ldots, a_n \in \mathbb{R}$, and ν_n be the empirical measure on $b_1, b_2, \ldots, b_n \in \mathbb{R}$. Then*

$$W_2(\mu_n, \nu_n)^2 \;=\; \min_{\pi} \frac{1}{n} \sum_{i=1}^{n} (a_i - b_{\pi(i)})^2 \;=\; \frac{1}{n} \sum_{i=1}^{n} (a_{(i)} - b_{(i)})^2 \,,$$

where $\min_\pi$ denotes minimization over permutations π on $[n]$.

For probability measures μ and ν, we use $\mathbf{KL}(\mu, \nu)$ to denote the Kullback-Leibler divergence between μ and ν, and $\|\mu - \nu\|_{\mathsf{tv}}$ to denote the total variation distance between μ and ν.

Since $\bar{\pi}$ is unknown in the measurement model from (4.15), we may assume that $y_1, y_2, \ldots, y_n$ are provided as an unordered multiset, denoted by $\{y_i\}_{i=1}^n$. In fact, we shall use the following equivalent generative process:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n$ i.i.d. from either $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$ (in Section 4.3.g) or the uniform distribution on $[-1/2, 1/2]^d$ (in Section 4.3.g), and independently, draw $\boldsymbol{\varepsilon} \sim \mathrm{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}_n)$.

2. Set $\boldsymbol{h}_{\bar{\boldsymbol{w}}} := (\bar{\boldsymbol{w}}^\top \boldsymbol{x}_1, \bar{\boldsymbol{w}}^\top \boldsymbol{x}_2, \ldots, \bar{\boldsymbol{w}}^\top \boldsymbol{x}_n)^\top$.

3. Set $\boldsymbol{y} := \boldsymbol{h}_{\bar{\boldsymbol{w}}}^{\uparrow} + \boldsymbol{\varepsilon}$.

It is clear that $((\boldsymbol{x}_i)_{i=1}^n, \{y_i\}_{i=1}^n)$ has the same distribution under this model as under that from (4.15).

Normal case

We first consider the case where $(\boldsymbol{x}_i)_{i=1}^n$ are i.i.d. draws from $\mathrm{N}(\mathbf{0}, \boldsymbol{I}_d)$. By homogeneity, we may assume without loss of generality that $\|\bar{\boldsymbol{w}}\|_2 = 1$, so $\mathsf{SNR} = 1/\sigma^2$.

The proof is based on the Generalized Fano method of [123] as described by [128].

Lemma 4.3.13 (Lemma 3 in [128])**.** *Let (Θ, ρ) be a pseudometric space, and let $\widetilde{\Theta} \subseteq \Theta$ index a collection of probability measures $(P_\theta)_{\theta \in \widetilde{\Theta}}$ such that $\rho(\theta, \theta') \geq \alpha$ and $\mathbf{KL}(P_\theta, P_\theta) \leq \beta$ for all* *distinct pairs $\theta, \theta' \in \widetilde{\Theta}$. Then for any estimator $\widehat{\theta}$ taking values in Θ,*

$$\max_{\theta \in \widetilde{\Theta}} \mathbf{E}_{P_\theta}\left[\rho(\widehat{\theta}, \theta)\right] \;\geq\; \frac{\alpha}{2}\left(1 - \frac{\beta + \ln 2}{\ln |\widetilde{\Theta}|}\right) ,$$

where $\mathbf{E}_{P_\theta}$ denotes expectation with respect to data drawn from P_θ.

We apply Lemma 4.3.13 with $(\Theta, \rho) = (S^{d-1}, \|\cdot\|_2)$. We construct a packing U of the unit sphere $S^{d-1} := \{\boldsymbol{u} \in \mathbb{R}^d : \|\boldsymbol{u}\|_2 = 1\}$ using the following variant of the Gilbert-Varshamov bound.

Lemma 4.3.14 (Lemma 4.10 in [129]). *For every $h \in [d]$ such that $h \leq d/4$, there exists a subset C of $\{0,1\}^d$ such that (i) the Hamming weight of each $\boldsymbol{c} \in C$ is h, (ii) the Hamming distance between every distinct pair $\boldsymbol{c}, \boldsymbol{c}' \in C$ is more than $h/2$, and (iii) the cardinality of C satisfies $\ln |C| \geq 0.233 h \ln(d/h)$.*

We take $C \subseteq \{0,1\}^d$ as guaranteed by Lemma 4.3.14 with $h := \lfloor d/4 \rfloor$, and let

$$U \;:=\; \left\{\boldsymbol{c}/\sqrt{h} : \boldsymbol{c} \in C\right\} \;\subset\; S^{d-1} .$$

Observe that U is a $(1/\sqrt{2})$-packing of S^{d-1} (i.e., every distinct pair $\boldsymbol{u}, \boldsymbol{u}' \in U$ satisfies $\|\boldsymbol{u} - \boldsymbol{u}'\|_2 > 1/\sqrt{2}$), and

$$\ln |U| \;\geq\; 0.233\left(\frac{d}{4} - 1\right)\ln 4 .$$

For each $\boldsymbol{u} \in U$, let $P_{\boldsymbol{u}}$ denote the probability distribution of $((\boldsymbol{x}_i)_{i=1}^n, \{y_i\}_{i=1}^n)$ when $\bar{\boldsymbol{w}} = \boldsymbol{u}$. Also, define $Q_{\boldsymbol{u}}$ to be the corresponding conditional distribution of $\{y_i\}_{i=1}^n$ given $(\boldsymbol{x}_i)_{i=1}^n$, and $\tilde{Q}_{\boldsymbol{u}}$ to be the corresponding conditional distribution of $\boldsymbol{y}$ given $(\boldsymbol{x}_i)_{i=1}^n$.

For any $\boldsymbol{u}, \boldsymbol{u}' \in U$,

$$\mathbf{KL}(Q_{\boldsymbol{u}}, Q_{\boldsymbol{u}'}) \;\leq\; \mathbf{KL}(\tilde{Q}_{\boldsymbol{u}}, \tilde{Q}_{\boldsymbol{u}'}) \;=\; \frac{1}{2\sigma^2}\left\|\boldsymbol{h}_{\boldsymbol{u}}^{\uparrow} - \boldsymbol{h}_{\boldsymbol{u}'}^{\uparrow}\right\|_2^2 \tag{4.25}$$

by the data processing inequality for $\mathbf{KL}$-divergence and the properties of the multivariate Gaussian distribution. We define $\mathcal{E}$ to be the event in which

$$\left\|\boldsymbol{h}_{\boldsymbol{u}}^{\uparrow} - \boldsymbol{h}_{\boldsymbol{u}'}^{\uparrow}\right\|_2^2 \ \leq \ \left(\sqrt{C_0 \log\log(n)} + \sqrt{8\ln(|U|^2)}\right)^2$$

for all distinct $\boldsymbol{u}, \boldsymbol{u}' \in U$, where $C_0 > 0$ is the absolute constant from Lemma 4.3.16 (below). By Equation (4.25), Lemma 4.3.16, and a union bound, we have $\mathbf{Pr}(\mathcal{E}) \geq 1/2$. Therefore, by Lemma 4.3.13, for any estimator $\widehat{\boldsymbol{w}}$,

$$\begin{aligned}
\max_{\boldsymbol{u}\in U} \mathbf{E}_{P_{\boldsymbol{u}}}\left[\|\widehat{\boldsymbol{w}} - \boldsymbol{u}\|_2\right] \ &\geq \ \max_{\boldsymbol{u}\in U} \mathbf{E}_{P_{\boldsymbol{u}}}\left[\|\widehat{\boldsymbol{w}} - \boldsymbol{u}\|_2 \mid \mathcal{E}\right] \cdot \mathbf{Pr}(\mathcal{E}) \\
&\geq \ \frac{1}{2\sqrt{2}}\left(1 - \frac{C_0 \log\log(n) + 16\ln|U|}{\sigma^2 \ln|U|} - \frac{\ln 2}{\ln|U|}\right) \cdot \frac{1}{2} \\
&= \ \frac{1}{4\sqrt{2}}\left(1 - \frac{C_0 \log\log(n)}{\sigma^2 \ln|U|} - \frac{16}{\sigma^2} - \frac{\ln 2}{\ln|U|}\right) .
\end{aligned}$$

Plugging in the lower bound for $\ln|U|$ and the upper bound on $\mathsf{SNR} = 1/\sigma^2$ completes the proof.

□

Uniform case

We now consider the case where $(\boldsymbol{x}_i)_{i=1}^n$ are drawn i.i.d. from the uniform distribution on $[-1/2, 1/2]^d$.[6] Again, by homogeneity, we assume without loss of generality that $\|\bar{\boldsymbol{w}}\|_2 = 1$, so $\mathsf{SNR} = 1/\sigma^2$.

The proof is based on the two-point method of [130] as described by [128].

Lemma 4.3.15 (Lemma 1 in [128]). *Let (Θ, ρ) be a pseudometric space, and let $\theta_1, \theta_2 \in \Theta$ correspond to probability measures P_{θ_1} and P_{θ_2} on the same space. Then for any estimator $\widehat{\theta}$ taking values in Θ,*

$$\max_{\theta\in\{\theta_1,\theta_2\}} \mathbf{E}_{P_\theta}\left[\rho(\widehat{\theta}, \theta)\right] \ \geq \ \frac{1}{2}\rho(\theta_1, \theta_2)\left(1 - \|P_{\theta_1} - P_{\theta_2}\|_{\mathsf{tv}}\right) ,$$

where $\mathbf{E}_{P_\theta}$ denotes expectation with respect to data drawn from P_θ.

[6] We actually just need that the marginal distribution of the first coordinate of each $\boldsymbol{x}_i$ be uniform on $[-1/2, 1/2]$.

We apply Lemma 4.3.15 with $(\Theta, \rho) = (S^{d-1}, \|\cdot\|_2)$. As before, we define for each $\boldsymbol{u} \in \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}$:

- $P_{\boldsymbol{u}}$, the distribution of $((\boldsymbol{x}_i)_{i=1}^n, \{y_i\}_{i=1}^n)$ when $\bar{\boldsymbol{w}} = \boldsymbol{u}$;
- $Q_{\boldsymbol{u}}$, the corresponding conditional distribution of $\{y_i\}_{i=1}^n$ given $(\boldsymbol{x}_i)_{i=1}^n$;
- $\tilde{Q}_{\boldsymbol{u}}$, the corresponding conditional distribution of $\boldsymbol{y}$ given $(\boldsymbol{x}_i)_{i=1}^n$.

Let $\mathcal{E}$ be the event in which

$$\left\|\boldsymbol{h}^{\uparrow}_{\boldsymbol{e}_1} - \boldsymbol{h}^{\uparrow}_{-\boldsymbol{e}_1}\right\|_2^2 \leq 1 \,.$$

By Lemma 4.3.20 (below), $\mathbf{Pr}(\mathcal{E}) \geq 1/2$. Moreover, since $P_{\boldsymbol{e}_1}(\mathcal{E}) = P_{-\boldsymbol{e}_1}(\mathcal{E}) = \mathbf{Pr}(\mathcal{E})$,

$$\begin{aligned}
\|P_{\boldsymbol{e}_1} - P_{-\boldsymbol{e}_1}\|_{\mathsf{tv}} &\leq \left\|P_{\boldsymbol{e}_1}(\cdot \mid \mathcal{E}) - P_{-\boldsymbol{e}_1}(\cdot \mid \mathcal{E})\right\|_{\mathsf{tv}} \mathbf{Pr}(\mathcal{E}) + (1 - \mathbf{Pr}(\mathcal{E})) \\
&\leq \sqrt{\frac{1}{2}\mathbf{KL}(P_{\boldsymbol{e}_1}(\cdot \mid \mathcal{E}), P_{-\boldsymbol{e}_1}(\cdot \mid \mathcal{E}))}\, \mathbf{Pr}(\mathcal{E}) + (1 - \mathbf{Pr}(\mathcal{E})) \\
&\leq \sqrt{\frac{1}{2} \cdot \frac{1}{2\sigma^2}}\, \mathbf{Pr}(\mathcal{E}) + (1 - \mathbf{Pr}(\mathcal{E})) \\
&\leq \frac{1}{2}\left(1 + \frac{1}{\sqrt{2}}\right) .
\end{aligned}$$

Above, the second inequality follows from Pinsker's inequality; the third inequality uses (4.25) and the fact $\|\boldsymbol{h}^{\uparrow}_{\boldsymbol{e}_1} - \boldsymbol{h}^{\uparrow}_{-\boldsymbol{e}_1}\|_2^2 \leq 1$ on the event $\mathcal{E}$, the fourth inequality uses the assumption that $\mathsf{SNR} = 1/\sigma^2 \leq 2$ and the fact $\mathbf{Pr}(\mathcal{E}) \geq 1/2$. We conclude by Lemma 4.3.15 that

$$\max_{\boldsymbol{u} \in \{\boldsymbol{e}_1, -\boldsymbol{e}_2\}} \mathbf{E}_{P_{\boldsymbol{u}}}\left[\|\widehat{\boldsymbol{w}} - \boldsymbol{u}\|_2\right] \geq \frac{1}{2} \cdot 2 \cdot \left(1 - \frac{1}{2}\left(1 + \frac{1}{\sqrt{2}}\right)\right) = \frac{1}{2}\left(1 - \frac{1}{\sqrt{2}}\right) ,$$

completing the proof. □

Auxiliary results

Lemma 4.3.16. *There is an absolute constant* $C_0 > 0$ *such that the following holds. Let* $n \geq 3$*, and let* $\boldsymbol{X}$ *be a random* $n \times d$ *matrix of i.i.d.* $\mathrm{N}(0,1)$ *random variables. For any unit vectors*

$\boldsymbol{u}, \boldsymbol{u}' \in S^{d-1}$ and $\delta \in (0,1)$,

$$\mathbf{Pr}\left(\left\|(\boldsymbol{X}\boldsymbol{u})^{\uparrow} - (\boldsymbol{X}\boldsymbol{u}')^{\uparrow}\right\|_2 \geq \sqrt{C_0 \log\log(n)} + \sqrt{8\ln(1/\delta)}\right) \;\leq\; \delta\,.$$

The proof of Lemma 4.3.16 uses the following lemmas.

Lemma 4.3.17 (Corollary 6.14 in [127]). *There is an absolute constant $C > 0$ such that the following holds. If $n \geq 3$, μ is the standard Gaussian measure on $\mathbb{R}$, and μ_n is the empirical measure for a size-n i.i.d. sample from μ, then*

$$\mathbf{E}\left[W_2(\mu_n, \mu)^2\right] \;\leq\; \frac{C \log\log(n)}{n}\,.$$

Lemma 4.3.18 (Eq. 2.35 in [131]). *Let $\boldsymbol{Z} \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ be a standard normal random vector in $\mathbb{R}^p$, and $f\colon \mathbb{R}^p \to \mathbb{R}$ be L-Lipschitz with respect to the Euclidean metric. Then for any $t > 0$,*

$$\mathbf{Pr}\left(f(\boldsymbol{Z}) \geq \mathbf{E}\, f(\boldsymbol{Z}) + t\right) \;\leq\; e^{-t^2/(2L^2)}\,.$$

Proof of Lemma 4.3.16. Fix unit vectors $\boldsymbol{u}$ and $\boldsymbol{u}'$. Observe that the entries of each of $\boldsymbol{X}\boldsymbol{u}$ and $\boldsymbol{X}\boldsymbol{u}'$ comprises an i.i.d. sample from $\mathrm{N}(0,1) =: \mu$; let μ_n and ν_n denote the respective empirical measures. Define the function $f\colon \mathbb{R}^{n\times d} \to \mathbb{R}$ by

$$f(\boldsymbol{A}) \;:=\; \left\|(\boldsymbol{A}\boldsymbol{u})^{\uparrow} - (\boldsymbol{A}\boldsymbol{u}')^{\uparrow}\right\|_2\,.$$

Then, by Lemma 4.3.12, the triangle inequality, Jensen's inequality, and Lemma 4.3.17,

$$\frac{\mathbf{E}\, f(\boldsymbol{X})}{\sqrt{n}} \;=\; \mathbf{E}\, W_2(\mu_n, \nu_n) \;\leq\; \mathbf{E}\, W_2(\mu_n, \mu) + \mathbf{E}\, W_2(\nu_n, \mu) \;\leq\; 2\sqrt{\mathbf{E}\, W_2(\mu_n, \mu)^2} \;\leq\; \sqrt{\frac{C_0 \log\log(n)}{n}}\,.$$

Moreover, for any $\boldsymbol{A}, \boldsymbol{A}' \in \mathbb{R}^{n\times d}$,

$$\begin{aligned} f(\boldsymbol{A}) - f(\boldsymbol{A}') &\leq \left\|(\boldsymbol{Au})^{\uparrow} - (\boldsymbol{Au}')^{\uparrow} - (\boldsymbol{A}'\boldsymbol{u})^{\uparrow} + (\boldsymbol{A}'\boldsymbol{u}')^{\uparrow}\right\|_2 \\ &\leq \left\|(\boldsymbol{Au})^{\uparrow} - (\boldsymbol{A}'\boldsymbol{u})^{\uparrow}\right\|_2 + \left\|(\boldsymbol{Au}')^{\uparrow} - (\boldsymbol{A}'\boldsymbol{u}')^{\uparrow}\right\|_2 \\ &\leq \left\|\boldsymbol{Au} - \boldsymbol{A}'\boldsymbol{u}\right\|_2 + \left\|\boldsymbol{Au}' - \boldsymbol{A}'\boldsymbol{u}'\right\|_2 \\ &\leq 2\left\|\boldsymbol{A} - \boldsymbol{A}'\right\|_F , \end{aligned}$$

where the first two steps follow from the triangle inequality, the third step uses Lemma 4.3.12, and $\|\cdot\|_F$ denotes the Frobenius norm. Therefore, f is 2-Lipschitz with respect to the Euclidean metric on $\mathbb{R}^{n\times d}$. By Lemma 4.3.18, for any $\delta \in (0,1)$,

$$\mathbf{Pr}\left(f(\boldsymbol{X}) \geq \mathbf{E}\, f(\boldsymbol{X}) + \sqrt{8\ln(1/\delta)}\right) \leq \delta .$$

Combining this with the upper bound on $\mathbf{E}\, f(\boldsymbol{X})$ completes the proof. □

Lemma 4.3.19 (Eqs. 1.7.3 and 1.7.5 in [132]). *Let $X_1, X_2, \ldots, X_n$ be i.i.d. draws from the uniform distribution on $[0,1]$. For any $r \in [n]$,*

$$\mathbf{E}[X_{(r)}] = \frac{r}{n+1} ,$$

and for any $r, s \in [n]$ with $r \leq s$,

$$\operatorname{cov}(X_{(r)}, X_{(s)}) = \frac{r}{n+1} \cdot \left(1 - \frac{s}{n+1}\right) \cdot \frac{1}{n+2} .$$

Lemma 4.3.20. *Let $U_1, U_2, \ldots, U_n$ be i.i.d. draws from the uniform distribution on $[-1/2, 1/2]$. Then*

$$\mathbf{Pr}\left(\sum_{i=1}^{n} \left(U_{(1)} + U_{(n+1-i)}\right)^2 \geq 1\right) \leq \frac{1}{2} .$$

Proof. It suffices to show the expectation bound

$$\mathbf{E}\left[\sum_{i=1}^{n}\left(U_{(1)}+U_{(n+1-i)}\right)^2\right] \;\leq\; \frac{1}{2}\,,$$

since the claim then follows by Markov's inequality. Expanding the square and using linearity of expectation gives

$$\begin{aligned}\mathbf{E}\left[\sum_{i=1}^{n}\left(U_{(1)}+U_{(n+1-i)}\right)^2\right] \;&=\; 2\sum_{i=1}^{n}\mathbf{E}\left[U_i^2\right]+2\sum_{i=1}^{n}\mathbf{E}\left[U_{(i)}U_{(n+1-i)}\right]\\ &=\; \frac{n}{6}+2\sum_{i=1}^{n}\mathbf{E}\left[U_{(i)}U_{(n+1-i)}\right]\;.\end{aligned}$$

By Lemma 4.3.19, we have for $i \leq (n+1)/2$,

$$\mathbf{E}\left[U_{(i)}U_{(n+1-i)}\right] \;=\; -\left(\frac{i}{n+1}-\frac{1}{2}\right)^2+\frac{i^2}{(n+1)^2(n+2)}\,,$$

and for $i > (n+1)/2$,

$$\mathbf{E}\left[U_{(i)}U_{(n+1-i)}\right] \;=\; -\left(\frac{i}{n+1}-\frac{1}{2}\right)^2+\frac{(n+1-i)^2}{(n+1)^2(n+2)}\,.$$

Plugging-in and simplifying gives

$$\mathbf{E}\left[\sum_{i=1}^{n}\left(U_{(1)}+U_{(n+1-i)}\right)^2\right] \;=\; \begin{cases}\frac{1}{2}\left(1-\frac{1}{n+1}\right) & \text{if } n \text{ is even}\,,\\ \frac{1}{2}\left(1-\frac{1}{n+2}\right) & \text{if } n \text{ is odd}\,.\end{cases}$$

□

References

[1] C. Zhang, B. Recht, S. Bengio, M. Hardt, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” in *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2019. arXiv: 1611.03530.

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” pp. 1–10, 2013. arXiv: 1312.6199.

[3] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, “Robustness may be at odds with accuracy,” *7th International Conference on Learning Representations, ICLR 2019*, pp. 1–24, 2019. arXiv: 1805.12152.

[4] K. Shi, D. Hsu, and A. Bishop, “A cryptographic approach to black box adversarial machine learning,” 2019. arXiv: 1906.03231.

[5] A. Bishop, L. Kowalczyk, T. Malkin, V. Pastro, M. Raykova, and K. Shi, “A Simple Obfuscation Scheme for Pattern-Matching with Wildcards,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10993 LNCS, no. 1, pp. 731–752, 2018.

[6] A. Andoni, D. Hsu, K. Shi, and X. Sun, “Correspondence retrieval,” *Proceedings of the 2017 Conference on Learning Theory*, vol. 65, pp. 105–126, 2017.

[7] D. Hsu, K. Shi, and X. Sun, “Linear regression without correspondence,” *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 1532–1541, 2017. arXiv: 1705.07048.

[8] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” no. c, pp. 1–14, 2016. arXiv: 1607.02533.

[9] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting Adversarial Attacks with Momentum,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 9185–9193, 2018. arXiv: 1710.06081.

[10] N. Carlini and D. Wagner, “Towards Evaluating the Robustness of Neural Networks,” *Proceedings - IEEE Symposium on Security and Privacy*, pp. 39–57, 2017. arXiv: 1608.04644.

[11] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples,” 2016. arXiv: 1605.07277.

[12] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical Black-Box Attacks against Machine Learning,” 2016. arXiv: `1602.02697`.

[13] P. Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C. J. Hsieh, “ZOO: Zeroth order optimization based black-box atacks to deep neural networks without training substitute models,” in *AISec 2017 - Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, co-located with CCS 2017*, 2017, ISBN: 9781450352024.

[14] W. Brendel, J. Rauber, and M. Bethge, “Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018, pp. 1–12. arXiv: `1712.04248`.

[15] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards Deep Learning Models Resistant to Adversarial Attacks,” pp. 1–27, 2017. arXiv: `1706.06083`.

[16] L. Schott, J. Rauber, M. Bethge, and W. Brendel, “Towards the first adversarially robust neural network model on MNIST,” vol. 3, pp. 1–16, 2018. arXiv: `1805.09190`.

[17] I. G. Jacob Buckman, Aurko Roy, Colin Raffell, “Thermometer Encoding: One Hot Way To Resist Adversarial Examples,” *Iclr*, vol. 19, no. 1, pp. 92–97, 2018.

[18] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, “Mitigating Adversarial Effects Through Randomization,” pp. 1–16, 2017. arXiv: `1711.01991`.

[19] Y. Sharma and P.-Y. Chen, “Attacking the Madry Defense Model with L_1-based Adversarial Examples,” pp. 1–9, 2017. arXiv: `1710.10733`.

[20] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples,” in *Icml*, 2018. arXiv: `1802.00420`.

[21] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” *International Conference on Learning Representations*, pp. 1–11, 2015. arXiv: `1412.6572`.

[22] J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. Goodfellow, “Adversarial Spheres,” 2018. arXiv: `1801.02774`.

[23] N. Ford, J. Gilmer, N. Carlini, and D. Cubuk, “Adversarial Examples Are a Natural Consequence of Test Error in Noise,” 2019. arXiv: `1901.10513`.

[24] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial Examples Are Not Bugs, They Are Features,” 2019. arXiv: `1905.02175`.

[25] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble Adversarial Training: Attacks and Defenses," pp. 1–20, 2017. arXiv: `1705.07204`.

[26] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, R. Long, and P. McDaniel, "Technical Report on the CleverHans v2.1.0 Adversarial Examples Library," pp. 1–12, 2016. arXiv: `1610.00768`.

[27] T. G. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Output Codes," *Journal of Artificial Intelligence Research*, vol. 2, 1994. arXiv: `9501101 [cs]`.

[28] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. 1978, p. 309.

[29] Y LeCun, C Cortes, and C. J. C. Burges, "The MNIST dataset of handwritten digits," *http://yann.lecun.com/exdb/mnist/*, 1998.

[30] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," *arXiv 2009*, 2009. arXiv: `arXiv:1011.1669v3`.

[31] A. Mądry, "MadryLab CIFAR10 Adversarial Examples Challenge," 2017.

[32] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," 2016. arXiv: `1605.07146`.

[33] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019. arXiv: `arXiv:1812.11118v2`.

[34] A. J. Wyner, M. Olson, J. Bleich, and D. Mease, "Explaining the Success of AdaBoost and Random Forests as Interpolating Classifiers," vol. 18, pp. 1–33, 2015. arXiv: `1504.07676`.

[35] P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler, "Benign Overfitting in Linear Regression," 2019. arXiv: `1906.11300`.

[36] V. Vapnik and O. Chapelle, "Bounds on error expectation for support vector machines," *Neural Computation*, vol. 12, no. 9, pp. 2013–2036, 2000.

[37] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang, "On the (im)possibility of obfuscating programs," in *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, 2001, pp. 1–18.

[38] B. Lynn, M. Prabhakaran, and A. Sahai, “Positive results and techniques for obfuscation,” in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, 2004, pp. 20–39.

[39] H. Wee, “On obfuscating point functions,” in *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, 2005, pp. 523–532.

[40] R. Canetti, G. N. Rothblum, and M. Varia, “Obfuscation of hyperplane membership,” in *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, 2010, pp. 72–89.

[41] S. Garg, C. Gentry, and S. Halevi, “Candidate multilinear maps from ideal lattices,” in *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, 2013, pp. 1–17.

[42] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, “Candidate indistinguishability obfuscation and functional encryption for all circuits,” in *FOCS*, 2013.

[43] A. Sahai and B. Waters, “How to use indistinguishability obfuscation: Deniable encryption, and more,” in *STOC*, 2014.

[44] C. Gentry, A. B. Lewko, A. Sahai, and B. Waters, “Indistinguishability obfuscation from the multilinear subgroup elimination assumption,” in *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, 2015, pp. 151–170.

[45] H. Lin and V. Vaikuntanathan, “Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings,” in *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, 2016, pp. 11–20.

[46] H. Lin, “Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 prgs,” in *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, 2017, pp. 599–629.

[47] H. Lin and S. Tessaro, “Indistinguishability obfuscation from trilinear maps and block-wise local prgs,” in *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, 2017, pp. 630–660.

[48] B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai, "Protecting obfuscation against algebraic attacks," in *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, 2014, pp. 221–238.

[49] J. Zimmerman, "How to obfuscate programs directly," in *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, 2015, pp. 439–467.

[50] B. Applebaum and Z. Brakerski, "Obfuscating circuits via composite-order graded encoding," in *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, 2015, pp. 528–556.

[51] S. Badrinarayanan, E. Miles, A. Sahai, and M. Zhandry, "Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits," in *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, 2016, pp. 764–791.

[52] H. Lin, "Indistinguishability obfuscation from constant-degree graded encoding schemes," in *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, 2016, pp. 28–57.

[53] S. Garg, E. Miles, P. Mukherjee, A. Sahai, A. Srinivasan, and M. Zhandry, "Secure obfuscation in a weak multilinear map model," in *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, 2016, pp. 241–268.

[54] P. Ananth and A. Sahai, "Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps," in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, 2017, pp. 152–181.

[55] J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé, "Cryptanalysis of the multilinear map over the integers," in *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, 2015, pp. 3–12.

[56] J. Coron, C. Gentry, S. Halevi, T. Lepoint, H. K. Maji, E. Miles, M. Raykova, A. Sahai, and M. Tibouchi, "Zeroizing without low-level zeroes: New MMAP attacks and their limitations," in *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, 2015, pp. 247–266.

[57] E. Miles, A. Sahai, and M. Zhandry, "Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13," in *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, 2016, pp. 629–658.

[58] Y. Chen, C. Gentry, and S. Halevi, "Cryptanalyses of candidate branching program obfuscators," in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, 2017, pp. 278–307.

[59] J. Coron, M. S. Lee, T. Lepoint, and M. Tibouchi, "Zeroizing attacks on indistinguishability obfuscation over CLT13," in *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part I*, 2017, pp. 41–58.

[60] D. Apon, N. Döttling, S. Garg, and P. Mukherjee, "Cryptanalysis of indistinguishability obfuscations of circuits over GGH13," in *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, 2017, 38:1–38:16.

[61] P. Ananth, A. Jain, and A. Sahai, "Patchable indistinguishability obfuscation: IO for evolving software," in *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, 2017, pp. 127–155.

[62] R. Goyal, V. Koppula, and B. Waters, "Lockable obfuscation," in *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, 2017, pp. 612–621.

[63] D. Hofheinz, T. Jager, D. Khurana, A. Sahai, B. Waters, and M. Zhandry, "How to generate and use universal samplers," in *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, pp. 715–744.

[64] Z. Brakerski and G. N. Rothblum, "Obfuscating conjunctions," in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, 2013, pp. 416–434.

[65] Z. Brakerski, V. Vaikuntanathan, H. Wee, and D. Wichs, "Obfuscating conjunctions under entropic ring LWE," in *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, 2016, pp. 147–156.

[66] D. Wichs and G. Zirdelis, "Obfuscating compute-and-compare programs under LWE," *IACR Cryptology ePrint Archive*, vol. 2017, p. 276, 2017.

[67] C. Peikert, “On error correction in the exponent,” in *Proceedings of the Third Conference on Theory of Cryptography*, ser. TCC’06, New York, NY: Springer-Verlag, 2006, pp. 167–183, ISBN: 3-540-32731-2, 978-3-540-32731-8.

[68] S. Brands, “Untraceable off-line cash in wallet with observers,” in *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO ’93, Santa Barbara, California, USA: Springer-Verlag, 1994, pp. 302–318, ISBN: 0-387-57766-1.

[69] D. Boneh, X. Boyen, and E. Goh, “Hierarchical identity based encryption with constant size ciphertext,” in *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, 2005, pp. 440–456.

[70] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio, “Sharp Minima Can Generalize For Deep Nets,” 2017. arXiv: `1703.04933`.

[71] L. Sagun, L. Bottou, and Y. LeCun, “Eigenvalues of the Hessian in Deep Learning: Singularity and Beyond,” pp. 1–8, 2016. arXiv: `1611.07476`.

[72] P. Chaudhari, A. Choromanska, S. Soatto, Y. LeCun, C. Baldassi, C. Borgs, J. Chayes, L. Sagun, and R. Zecchina, “Entropy-SGD: Biasing Gradient Descent Into Wide Valleys,” in *International Conference on Learning Representations*, 2017, pp. 1–19, ISBN: 978-3-642-04273-7. arXiv: `1611.01838`.

[73] S. Mandt, M. D. Hoffman, and D. M. Blei, “Stochastic Gradient Descent as Approximate Bayesian Inference,” *Journal of Machine Learning Research*, vol. 18, pp. 1–35, 2017. arXiv: `1704.04289`.

[74] W. Hu, C. J. Li, L. Li, and J.-G. Liu, “On the diffusion approximation of nonconvex stochastic gradient descent,” 2017. arXiv: `1705.07562`.

[75] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, “Don’t Decay the Learning Rate, Increase the Batch Size,” no. 2017, pp. 1–11, 2017. arXiv: `1711.00489`.

[76] L. Yin and P Ao, “Existence and construction of dynamical potential in nonequilibrium processes without detailed balance,” *Journal of Physics A: Mathematical and General*, vol. 39, no. 27, pp. 8593–8601, 2006.

[77] P. Chaudhari and S. Soatto, “Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks,” in *International Conference on Learning Representations*, 2018, pp. 1–20. arXiv: `1710.11029`.

[78] H. Risken and T. Frank, *The Fokker-Planck Equation: Methods of Solutions and Applications (Springer Series in Synergetics)*. 1996, ISBN: 354061530X.

[79] Y. Shechtman, Y. C. Eldar, O. Cohen, H. N. Chapman, J. Miao, and M. Segev, "Phase retrieval with application to optical imaging: A contemporary overview," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 87–109, 2015.

[80] K. Jaganathan, Y. C. Eldar, and B. Hassibi, "Phase retrieval: An overview of recent developments," *arXiv preprint arXiv:1510.07713*, 2015.

[81] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," in *Advances in Neural Information Processing Systems*, 2013, pp. 2796–2804.

[82] E. J. Candes, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Communications on Pure and Applied Mathematics*, vol. 66, no. 8, pp. 1241–1274, 2013.

[83] E. J. Candès and X. Li, "Solving quadratic equations via phaselift when there are about as many equations as unknowns," *Foundations of Computational Mathematics*, vol. 14, no. 5, pp. 1017–1026, 2014.

[84] B. Alexeev, A. S. Bandeira, M. Fickus, and D. G. Mixon, "Phase retrieval with polarization," *SIAM Journal on Imaging Sciences*, vol. 7, no. 1, pp. 35–66, 2014.

[85] Y. C. Eldar and S. Mendelson, "Phase retrieval: Stability and recovery guarantees," *Applied and Computational Harmonic Analysis*, vol. 36, no. 3, pp. 473–494, 2014.

[86] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, "Phase retrieval via matrix completion," *SIAM review*, vol. 57, no. 2, pp. 225–251, 2015.

[87] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval from coded diffraction patterns," *Applied and Computational Harmonic Analysis*, vol. 39, no. 2, pp. 277–299, 2015.

[88] ——, "Phase retrieval via wirtinger flow: Theory and algorithms," *IEEE Transactions on Information Theory*, vol. 61, no. 4, pp. 1985–2007, 2015.

[89] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Mathematical Programming*, vol. 149, no. 1-2, pp. 47–81, 2015.

[90] Y. Chen and E. Candes, "Solving random quadratic systems of equations is nearly as easy as solving linear systems," in *Advances in Neural Information Processing Systems*, 2015, pp. 739–747.

[91] S. Sanghavi, R. Ward, and C. D. White, "The local convexity of solving systems of quadratic equations," *Results in Mathematics*, pp. 1–40, 2016.

[92] H. Zhang and Y. Liang, "Reshaped wirtinger flow for solving quadratic system of equations," in *Advances in Neural Information Processing Systems*, 2016, pp. 2622–2630.

[93] G. Wang, G. B. Giannakis, and Y. C. Eldar, "Solving systems of random quadratic equations via truncated amplitude flow," *arXiv preprint arXiv:1605.08285*, 2016.

[94] R. Kolte and A. Özgür, "Phase retrieval via incremental truncated wirtinger flow," *arXiv preprint arXiv:1606.03196*, 2016.

[95] B. Gao and Z. Xu, "Gauss-newton method for phase retrieval," *arXiv preprint*, 2016.

[96] J. Sun, Q. Qu, and J. Wright, "A geometric analysis of phase retrieval," in *Information Theory (ISIT), 2016 IEEE International Symposium on*, IEEE, 2016, pp. 2379–2383.

[97] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, ACM, 1996, pp. 99–108.

[98] A. K. Lenstra, H. W. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, 1982.

[99] J. C. Lagarias and A. M. Odlyzko, "Solving low-density subset sum problems," *Journal of the ACM*, vol. 32, no. 1, pp. 229–246, 1985.

[100] M. R. Gary and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman and Company, New York, 1979.

[101] X. Yi, C. Caramanis, and S. Sanghavi, "Alternating minimization for mixed linear regression.," in *ICML*, 2014, pp. 613–621.

[102] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, "One sketch for all: Fast algorithms for compressed sensing," in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, ACM, 2007, pp. 237–246.

[103] R. Balan, P. Casazza, and D. Edidin, "On signal reconstruction without phase," *Applied and Computational Harmonic Analysis*, vol. 20, no. 3, pp. 345–356, 2006.

[104] X. Yi, C. Caramanis, and S. Sanghavi, "Solving a mixture of many random linear equations by tensor decomposition and alternating minimization," *arXiv preprint arXiv:1608.05749*, 2016.

[105] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models.," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2773–2832, 2014.

[106] R. Kueng, H. Rauhut, and U. Terstiege, "Low rank matrix recovery from rank one measurements," *Applied and Computational Harmonic Analysis*, vol. 42, no. 1, pp. 88–116, 2017.

[107] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, *Learning with rounding, revisited: New reduction, properties and applications*, Cryptology ePrint Archive, Report 2013/098, 2013.

[108] D. Stehlé, "Floating-point lll: Theoretical and practical aspects," in *The LLL Algorithm: Survey and Applications*, P. Q. Nguyen and B. Vallée, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 179–213, ISBN: 978-3-642-02295-1.

[109] A. Edelman, "Eigenvalues and condition numbers of random matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 9, no. 4, pp. 543–560, 1988.

[110] K. R. Davidson and S. J. Szarek, "Local operator theory, random matrices and banach spaces," *Handbook of the geometry of Banach spaces*, vol. 1, no. 317-366, p. 131, 2001.

[111] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge University Press, 1985.

[112] J. Unnikrishnan, S. Haghighatshoar, and M. Vetterli, "Unlabeled sensing with random linear measurements," *arXiv preprint arXiv:1512.00115*, 2015.

[113] A. Pananjady, M. J. Wainwright, and T. A. Courtade, "Linear regression with an unknown permutation: Statistical and computational limits," in *54th Annual Allerton Conference on Communication, Control, and Computing*, 2016, pp. 417–424.

[114] A. Abid, A. Poon, and J. Zou, "Linear regression with shuffled labels," *arXiv preprint arXiv:1705.01342*, 2017.

[115] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman and Company, New York, 1979.

[116] G. Elhami, A. J. Scholefield, B. Bejar Haro, and M. Vetterli, "Unlabeled sensing: Reconstruction algorithm and theoretical guarantees," in *Proceedings of the 42nd IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

[117] A. Pananjady, M. J. Wainwright, and T. A. Courtade, "Denoising linear models with permuted data," *arXiv preprint arXiv:1704.07461*, 2017.

[118] C. Boutsidis, P. Drineas, and M. Magdon-Ismail, "Near-optimal coresets for least-squares regression," *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6880–6892, 2013.

[119] M. Dereziński and M. K. Warmuth, "Unbiased estimates for linear regression via volume sampling," *arXiv preprint arXiv:1705.06908*, 2017.

[120] H. Avron and C. Boutsidis, “Faster subset selection for matrices and applications,” *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 4, pp. 1464–1499, 2013.

[121] A. M. Frieze, “On the lagarias-odlyzko algorithm for the subset sum problem,” *SIAM Journal on Computing*, vol. 15, no. 2, pp. 536–539, 1986.

[122] A. Andoni, D. Hsu, K. Shi, and X. Sun, “Correspondence retrieval,” in *Conference on Learning Theory*, 2017.

[123] T. S. Han and S. Verdú, “Generalizing the Fano inequality,” *IEEE Transactions on Information Theory*, vol. 40, no. 4, pp. 1247–1251, 1994.

[124] D. P. Woodruff, “Sketching as a tool for numerical linear algebra,” *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 1–2, pp. 1–157, 2014.

[125] M. Rudelson and R. Vershynin, “Non-asymptotic theory of random matrices: Extreme singular values,” *arXiv preprint arXiv:1003.2990*, 2010.

[126] B. Laurent and P. Massart, “Adaptive estimation of a quadratic functional by model selection,” *Annals of Statistics*, pp. 1302–1338, 2000.

[127] S. Bobkov and M. Ledoux, “One-dimensional empirical measures, order statistics and Kantorovich transport distances,” *preprint*, 2014.

[128] B. Yu, “Assouad, Fano, and Le Cam,” in *Festschrift for Lucien Le Cam*, Springer, 1997, pp. 423–435.

[129] P. Massart, *Concentration inequalities and model selection*. Springer, 2007, vol. 6.

[130] L. Le Cam, “Convergence of estimates under dimensionality restrictions,” *The Annals of Statistics*, pp. 38–53, 1973.

[131] M. Ledoux, *The Concentration of Measure Phenomenon*. American Mathematical Society, 2000.

[132] R.-D. Reiss, *Approximate distributions of order statistics: with applications to nonparametric statistics*. Springer Science & Business Media, 2012.