Abstract—The turnpike problem of recovering a set of points in \( \mathbb{R}^D \) from the set of their pairwise differences is a classic inverse problem in combinatorics. This problem or variants arise in a number of applications, prominently including bioinformatics and imaging, where it is intimately related to the phase retrieval problem of recovering a signal from magnitude-only measurements of its Fourier transform. We propose the Multidimensional Intersection Sparse Phase Retrieval (MISTR) algorithm for solving the turnpike problem in dimension \( D \geq 2 \). By taking advantage of the structure of multiple dimensions, we are able to achieve the same accuracy as the best one-dimensional algorithms in less time. We prove theoretically that MISTR correctly recovers most signals distributed as a Gaussian

Multidimensional Intersection Sparse Phase Retrieval (MISTR) only measurements of its Fourier transform. We propose the formatics and imaging, where it is intimately related to the turnpike problem

We consider a classic problem from combinatorics called the turnpike problem, in which one aims to recover an unknown set \( V \subseteq \mathbb{R}^D \) from its set of pairwise differences \( W \). We call the set \( W = \{ n_1 - n_2 \mid n_1, n_2 \in V \} \) the difference set of \( V \).

Definition 1 (Turnpike Problem).

\[
\text{Find } V \text{ subject to } W = \{ n_1 - n_2 \mid n_1, n_2 \in V \} \quad (1)
\]

This problem is fundamentally ill-posed: given any set \( V \) the sets \( V + c \) and \( -V \) will also produce the same difference set \( W \). Thus, in the above definition, we aim to recover an equivalent solution to \( V \), which differs only by a constant shift and possible multiplication by \( -1 \). We emphasize that under this definition, we do not know multiplicity information; that is, for each difference in \( W \), we do not know how many pairs of points in \( V \) generate this difference. The computational complexity of the turnpike problem is not known, but some variations on it have been shown to be NP-complete [1] [2]. Solutions to the turnpike problem are not unique in general, though some partial uniqueness results are known [3].

The turnpike problem and its variants have been applied fruitfully in numerous fields. In bioinformatics, a version called the “partial digest” problem that includes multiplicity data is used to reconstruct a complete strand of DNA from numerous cut pieces of different lengths [2]. In higher dimensions, the (unassigned) distance geometry problem arises when trying to understand the geometry of molecules and nanostructures [4]. In this version, the data consists of distances between points in \( V \) instead of directed differences; multiplicity information is usually assumed as well. The case when the difference set is corrupted with noise has also been considered [5] [6], as data in many of the above use cases is often noisy.

We are particularly motivated by applications in imaging. In many imaging setups, technological or physical constraints prevent sensors from detecting phases, and so the phase retrieval problem of recovering this missing phase information is an important problem for a number of fields, including biological imaging [7] and X-ray crystallography [8], among many others [9]. Magnitude-only knowledge of a signal \( x \) is equivalent to knowledge of its autocorrelation function, and under very mild conditions, the support of the autocorrelation of \( x \) is exactly the difference set of the support of \( x \). Thus, solving the turnpike problem on this difference set returns the support of the unknown signal \( x \) itself. Once the support is known, it is possible to recover the signal itself using convex relaxation techniques\(^1\). Importantly, unlike many other applications of turnpike-like problems, the phase retrieval problem does not afford the researcher any multiplicity information about the difference set \( W \).

A. Existing algorithms for the turnpike problem

In [10], Lemke and Werman proposed a factorization algorithm that recovers subsets \( V \subseteq \{0, 1, \ldots, N - 1\} \) in \( \mathcal{O}(k^m) \) time, where \( m \) is the largest difference in \( W \) and \( k \) is the number of elements in \( V \); however, this quickly becomes impractical as the set \( V \) grows large. Lemke et al. later proposed a backtracking algorithm for solving the turnpike problem [2], but this method has exponential worst-case complexity in \( k \). Juayong and Albores [11] outlined a parallel computing solution which can implement a similar \( \mathcal{O}(2^k k \log(k)) \) backtracking algorithm in linear time, but this method requires exponential space to make up the difference.

In [5], Jaganathan et al. showed that dramatically faster recovery times are possible when the set \( V \) is sparse. In this work, the authors developed a \( \mathcal{O}(k^4) \) algorithm for recovering most \( \mathcal{O}(N^{1/2 - \delta}) \)-sparse subsets of \( \{0, 1, \ldots, N - 1\} \) from their difference sets. Their primary tool is an intersection step

---

A. Novikov is with the Mathematics Department, Penn State University, University Park, PA 16802, USA e-mail: novikov@psu.edu
S. White is with the Mathematics Department, Penn State University, University Park, PA 16802, USA e-mail: sew347@psu.edu

\(^1\)For a detailed derivation of the relationship between the turnpike problem and phase retrieval, see [5].
that deletes many differences in \( W \) not in the support of \( V \) without deleting any support elements. Together with a method for phase retrieval with known support, the authors call this algorithm Two-Stage Phase Retrieval (TSPR).

Though TSPR is designed for the one-dimensional setting, one can immediately apply the results proven in [5] to the D-dimensional case by projecting a difference set \( W \subseteq \mathbb{R}^D \) on a random unit vector in \( \mathbb{R}^D \). The primary contribution of our present work is to capitalize on the difference in structure between one and multiple dimensions to achieve even better algorithmic performance.

B. Our contribution and organization of paper

The vast majority of work on the turnpike problem is set in only a single dimension. Motivated by phase retrieval applications in imaging, where data is typically two- or three-dimensional, our contribution is an algorithm that takes advantage of multidimensional structure to solve the D-dimensional turnpike problem more efficiently and accurately than leading one-dimensional alternatives. By exploiting the geometry of multiple dimensions, we develop an algorithm for solving (1) when \( D > 1 \) which runs in \( \mathcal{O}(k^2 \log(k)) \) average-case time. It is worth emphasizing that in the sparse case under consideration, the difference set \( W \) has approximately \( k^2 \) elements order-wise, and so our algorithm’s runtime is nearly linear in the size of the input. We provide accuracy guarantees for sparsity up to \( \mathcal{O}(N^{D/2-\delta}) \), though numerical simulations show that MISTR overperforms this bound in practice: for \( D = 3, k = N^{3D/5} \), no failures were observed out of 500 test runs for \( k \) as high as 1000. Lastly, we provide comprehensive numerical simulations verifying these findings.

The outline of the paper is as follows. In section II, we introduce the MISTR (multidimensional intersection sparse turnpike recovery) algorithm for efficiently solving (1) when \( D \geq 2 \). In section III, we prove accuracy guarantees for this algorithm that guarantee recovery as long as the set \( V \) is sufficiently sparse. Lastly, in section IV, we provide a number of reproducible numerical simulations which verify the effectiveness of MISTR in practice. All code used in the numerical experiments can be found at https://github.com/sew347/multi-dimensional-turnpike. The appendix contains proofs of technical lemmas as well as a few notes on our particular implementation of MISTR.

II. Algorithm

A. Notation and Setting

We begin by establishing the following notation. Let \( \mathbb{Z}^D \) be the grid of integer-valued vectors in \( \mathbb{R}^D \), and let \( H_{N,D} = H_N = \frac{1}{N^D} \mathbb{Z}^D \). \( N^D \) can be understood as the resolution of a discrete image window, up to a constant. We will write \( \mathcal{N}(A) \) for the number of points in a finite set \( A \). Throughout the paper we use \( \ln(x) \) to refer to the natural logarithm of \( x \), while \( \log(x) \) refers to the logarithm of base 2.

Let \( V \) denote our unknown support set \( V \subseteq H_N \). As we are approaching this problem from the perspective of imaging, we will refer to \( V \) as our set of scatterers. Let \( k \) be the \( ( \text{unknown) number of scatterers in } V \). We write \( W = \{v_1 - v_2 \mid v_1, v_2 \in V\} \) for the set of pairwise differences of \( V \), and denote \( K = \mathcal{N}(W) \). Though the exact number of scatterers is unknown a priori, one can compute the minimum number \( \hat{k}(K) \) of scatterers required to create \( K \) differences.

Since \( W \) has at most \( k(k-1) + 1 \) elements if all differences are distinct, \( k \geq \hat{k}(K) = \left\lceil \frac{1}{2} \left(1 + \sqrt{4K - 3}\right) \right\rceil \).

We will often want to order these sets by their inner product with different random vectors \( X_i \). We say that for two vectors \( x \) and \( y \), \( x < y \) with respect to \( X_i \) if and only if \( \langle x, X_i \rangle < \langle y, X_i \rangle \). We then index the set \( V \) according to \( X_i \) using the following notation: if \( v_j^i \) and \( v_m^i \) are vectors in \( V \), then \( j < m \iff v_j^i < v_m^i \) with respect to \( X_i \). We will refer to this ordering as the ordering induced by the vector \( X_i \). We will index starting from 0. We model sparsity as a power of \( N^D \); that is, \( s = N^{D\theta} \) for a fixed sparsity parameter \( \theta \). The exact equality is not important; it will be clear how one could generalize our results to apply if \( s = s(N) = \mathcal{O}(N^{D\theta}) \).

B. The MISTR Algorithm

We are now ready to introduce our algorithm, Multidimensional Intersection Sparse Turnpike Recovery (MISTR), which solves (1) with high probability. As inputs, MISTR takes an integer \( \tau \) and difference set \( W \) without multiplicity. The algorithm outputs an equivalent solution to \( V \) if one is found, or “false” if no solution is found.

First, MISTR generates \( \tau \) random vectors \( X_1, \ldots, X_\tau \) from the half-sphere \( S^D_+ = \{X \in S^D \mid X_1 > 0\} \). MISTR consists of a projection step followed by intersection step for each \( X_i \). After these \( \tau \) intersection steps, if no solution has been found, MISTR employs a collaboration search to find a solution. The idea for the intersection step is directly taken from Jaganathan et al. [5], while the use of random projections and the collaboration search are our contributions.

For each \( i \), MISTR attempts to recover a set \( \tilde{V}^i \) which is equivalent to the true support set \( V \). Specifically, for each \( i \), the intersection step will return a set \( U^i \) which is guaranteed to contain a set \( \tilde{V}^i \) that is equivalent to \( V \), defined by:

\[
\tilde{V}^i = \begin{cases} 
V - v_0^i, & \langle v_1^i - v_0^i, X_i \rangle \leq \langle v_{k-1}^i - v_{k-2}^i, X_i \rangle \\
v_{k-1}^i - V, & \text{otherwise}
\end{cases}
\]  \hspace{1cm} (2)

Essentially, \( \tilde{V}^i \) changes coordinates of \( V \) by choosing either \( v_0^i \) or \( v_{k-1}^i \) as the origin, based on the condition in (2), flipping by \(-1\) in the latter case. We will denote elements in \( \tilde{V}^i \) as \( \tilde{v}_j^i \), where the index order is induced by the inner product with \( X_i \). By this definition, we always have \( v_0^i = 0 \). Further, from (2) one can see that any \( \tilde{v} \in \tilde{V}^i \) is the difference between some point in \( V \) and either \( v_0^i \) or \( v_{k-1}^i \). It follows that \( \tilde{V}^i \subseteq W \). Thus our algorithm only needs to remove false positives: vectors \( \ell \)
such that \( \ell \in W \) but \( \ell \notin \vec{V}^i \). The goal of the intersection step is to efficiently eliminate as many of these false positive vectors as possible without removing any elements of \( \vec{V}^i \).

Algorithm 1 MISTR

**Input:** A set \( W \) that is the set of pairwise differences between points in an unknown set \( V \), without multiplicities.

**Output:** A set \( U \) with difference set \( W \), or FALSE if no such solution can be found.

1: Select \( \tau \) random vectors \( X_1, \ldots, X_\tau \) uniformly at random from \( \mathbb{R}^{D-1} \).
2: for \( i = 1 : \tau \) do
3: \hspace{1em} **Projection Step:** Compute \( W^i \) by projecting \( W \) onto \( X_i \), sorting, and selecting differences with positive inner product with \( X_i \).
4: \hspace{1em} Deduce \( \vec{v}^i_1 = w^i_k - w^i_k - 2 \).
5: \hspace{1em} **Intersection Step:** Compute \( U^i = \{0\} \cup [W^i \cap (W^i + \vec{v}^i_1)] \).
6: end for

7: **Collaboration Search:** Perform a breadth-first search for a solution on a particular (virtual) tree \( T \), using known properties of the sets \( \vec{V}^i \) to prune the vast majority of non-solutions.

1) Projection and intersection steps: **Projection Step:** The difference set \( W \) is ordered according to projection with \( X_i \), and \( W^i \) is taken to be the subset of vectors in \( W \) with nonnegative projections, indexed by the ordering induced by \( X_i \). Since \( W = -W \), we do not sacrifice any information about \( V \) by restricting \( W \) to this half-space. As noted above, we know \( \vec{v}^i_0 = 0 \). We can also infer that \( \vec{v}^i_1 = w^i_k - w^i_k - 2 \) by the following reasoning. It is immediate that the vector with the largest projection must be included in \( \vec{V}^i \); that is, \( w^i_k = \vec{v}^i_1 \). Further, by definition of \( \vec{V}^i \), \( \langle \vec{v}^i_1, -\vec{v}^i_0, X_i \rangle \leq \langle \vec{v}^i_1, -\vec{v}^i_1, X_i \rangle \). Thus, \( w^i_k - w^i_k = \vec{v}^i_1 - \vec{v}^i_1 \), so it follows that \( w^i_k = \vec{v}^i_1 \).

**Intersection Step:** This step is exactly the same as the intersection step employed by Jagathan et al. in [5]. This step finds:

\[
U^i = \{0\} \cup [W^i \cap (W^i + \vec{v}^i_1)]
\]

Here \( W^i + \vec{v}^i_1 = \{w + \vec{v}^i_1 \mid w \in W^i\} \). We quote directly\(^4\) the description of this step given in [5]:

The key idea of this step can be summarized as follows: suppose we know the value of \( \vec{v}^i_m \) for some \( m \), then

\[
\{ \vec{v}^i_j \mid m \leq j \leq k - 1 \} \subseteq W^i \cap (W^i + \vec{v}^i_m)
\]

... This can be seen as follows: \( \vec{v}^i_j(i) \in W^i \) by construction for \( 0 \leq j \leq k - 1 \), \( \vec{v}^i_m\) by construction for \( 0 \leq j \leq k - 1 \), which when added by \( \vec{v}^i_m \) gives \( \vec{v}^i_j \) and hence \( \vec{v}^i_j \in W^i \cap (W^i + \vec{v}^i_m) \) for \( m \leq j \leq k - 1 \).

We will write elements in \( U^i \) as \( u^i_j \), where the ordering \( j \) is the ordering induced by \( X_i \). Since we do not know how many elements are in \( U^i \) a priori, we will refer to the largest element in \( U^i \) in terms of this ordering as \( u^i_{\text{max}} \). We note that by construction, \( u^i_0 = \vec{v}^i_0 = 0 \), \( u^i_1 = \vec{v}^i_1 \), and \( u^i_{\text{max}} = \vec{v}^i_{k-1} \).

After \( U^i \) is found, if we have \( N(U^i) = k(K) \) as defined in (II-A), we know that it must be the true solution \( \vec{V}^i \). This is because \( N(U^i) = k(K) \) means \( U^i \) contains the minimum number of elements required to produce a difference set with \( K \) elements, and since \( V^i \subseteq U^i \) it follows that \( V^i = U^i \). Otherwise, we repeat the projection and intersection steps for \( i + 1 \) up to \( \tau \).

A single projection step takes \( O(k^2 \log(k)) \) time to compute and sort \( W^i \). The intersection step likewise takes \( O(k^2 \log(k)) \) time to compute \( W^i \cap (W^i + \vec{v}^i_0) \) [12]. Thus performing \( \tau \) intersection steps takes at most \( O(\tau k^2 \log(k)) \) time.

When the number of scatterers is small relative to \( N^D \) (\( s \approx N^D/4 \) or smaller), the projection and intersection steps are usually sufficient to find a solution. Indeed, it is proved in [5] that in one dimension a single intersection step can recover \( O(N^{1/4}) \) signals for large enough \( N \). However, for sufficiently high sparsity the probability that a single intersection step gives a solution becomes close to 0. Thus for larger \( s \), the projection and intersection steps are not sufficient to ensure recovery with high probability.

2) Collaboration search: Though the individual sets \( U^i \) are rarely solutions when the number of scatterers grows large, we do know that each \( U^i \) contains an equivalent solution. It thus stands to reason that taking the intersection would improve the chances of removing all false positive vectors. Care is needed, however: while the recovered solution always contains an equivalent solution, these solutions may differ by a possible global shift and negative flip as per (2). Thus we need to reorient the solutions \( U^i \) before we can take their intersection in a meaningful way.

From (2) we see that for each \( i \) there exists a transformation \( \Theta^i \) consisting of offset by a particular vector and possible multiplication by \( -1 \), such that \( \Theta^i(V^i) = \vec{V}^i \). (For notational convenience we take \( \Theta^i \) to be the identity \( \Theta^i(v) = v \).) For example, if \( \vec{V}^i = V - v^i_0 \) and \( V^2 = v^i_{k-1} - V \), then \( \vec{V}^i = v^i_{k-1} - v^i_0 - \vec{v}^2 \), in which case \( \Theta^2(v) = v^2_{k-1} - v^i_0 - v \). The collaboration search attempts to find the transformed intersection defined by:

\[
U = \bigcap_{i=1}^{\tau} \Theta^i(U^i)
\]

consisting of the intersections of the reoriented intersections \( U^i \). The difficulty lies in the fact that the sets \( \vec{V}^i \) are unknown a priori, as are the vectors \( v^i_{k-1} \) and \( v^i_0 \), so we must infer the transformations \( \Theta^i \) from the retrieved sets \( U^i \).

In order to formalize the above intuition, we introduce the following definitions. Consider a sequence \( j \) of \( d \) pairs \( (j_i, \omega_i) \) where \( j_i \in \{1, 2, \ldots, N(U^i) - 1\} \) and \( \omega_i \in \{1, -1\} \). We will consider only sequences \( j \) such that \( j_1 = 0 \) and \( \omega_1 = 1 \). Given such a sequence \( j \), we define its \( i \)-th orientation \( O^j_i \) as the function on \( U^i \) such that for any \( u \in U^i \):

\[
O^j_i(u) = \omega_i(u - u^i_{j_i})
\]

Thus the orientation \( O^j_i \) is a shift of the coordinates of \( U^i \) so that \( u^i_{j_i} \) is the origin, along with a possible flip by a factor of

\(^4\)Notation has been adapted to our setting.
We then define the collaboration (of $U^1, U^2, \ldots, U^d$) with respect to $j$, denoted $C_j$, as the intersection of the orientations:

$$C_j = U^1 \cap_{i=2}^{d} \omega_i (U^i - u^i_j) = \cap_{i=1}^{d} \omega_i (U^i - u^i_j) = \cap_{i=1}^{d} O_{j}^i(U^i)$$

We call $d$ the depth of the collaboration. It is clear that $U = \cap_{i=1}^{d} \Theta^i(U^i)$ is a collaboration of depth $\tau$. Denote $j^*$ the index such that $U = C_{j^*}$, and denote by $C_j$ the first $d$ terms of $j^*$.

Correspondingly, let $U_{j} = C_{j} = \cap_{i=1}^{d} \omega_i (U^i - u^i_j)$ be the partial collaboration with respect to $j$ up to depth $d$. We also note that $O_{j}^i = \Theta^i$.

It follows that, in principle, one could find $j^*$ by computing $C_j$ for every possible $j$ and checking whether $\text{Diff}(C_j) = W$. To do this would be totally impractical, of course, since the number of collaborations to compute would be enormous.

However, this informs the intuition that we can cast the problem of finding $U$ as a search algorithm. By treating the problem of finding $j^*$ as a tree-based search algorithm, we could leverage known information about $U_d$ to prune branches at each depth to dramatically reduce search time. We call this iterative pruning and search process the collaboration search.

Formally, we consider the tree $T$ consisting of all sequences $j$ as above with length between 1 and $\tau$, where the parent-child structure is the following. The root of the tree is the node consisting of the single-element sequence $(u^1_0, 1)$. Given any sequence $j_0 = (0,1), (j_1, \omega_2), \ldots, (j_d, \omega_d)$ will have as its children all the sequences $j_{d+1}, (j_{d+1}, \omega_1)$ for every possible pair $(j_{d+1}, \omega)$ with $j_{d+1} \in \{0,1,2,\ldots, N(U^{d+1}) - 1\}$ and $\omega \in \{1, -1\}$. It is clear that the leaves of this tree will be all possible indices $j$ of length $\tau$. An example of the initial setup of this tree can be found in section II-C, figure 1a.

The collaboration search is a combination of breadth-first search and pruning on this tree that uses the known structure of $U_d$ to avoid searching the vast majority of the tree’s nodes. (We note that while the collaboration search step is best described in terms of this search tree, it is never necessary in practice to actually construct the tree, which would take an impractical amount of time and space.)

To facilitate this search, the key observation is this: from the intersection step, we know that for each $i$, $u^i_1$ and $u^i_{\text{max}}$ must be elements of the equivalent solution $\bar{V}^i$. Thus, at each depth $d$, $\{\Theta^1(u^1_1, \Theta^i(u^i_{\text{max}})) \subseteq U_d$ for all $i = 1, 2, \ldots, d$. This means that $U_d$ must satisfy the following condition:

$$\{u^1_1, u^i_{\text{max}}, \Theta^2(u^2_1), \Theta^2(u^2_{\text{max}}), \ldots, \Theta^d(u^d_1), \Theta^d(u^d_{\text{max}})) \subseteq U_d$$

Thus for any node $j_d$ in $T$, if that node or one of its children are $j^*$, it must be that:

$$\{u^1_1, u^i_{\text{max}}, O_{j}^2(u^2_1), O_{j}^2(u^2_{\text{max}}), \ldots, O_{j}^d(u^d_1), O_{j}^d(u^d_{\text{max}})) \subseteq C_j$$

We find that (3) is most conveniently expressed as two different conditions, which are best checked in different ways:

$$\{u^1_1, u^i_{\text{max}} \subseteq O_{j}^i(U^i) \quad (4)$$

Condition 4 depends only on $j_d$, the $d$-th term of the sequence $j_d$. Thus 4 can be checked for the entire set $U^d$ at once due to the following observation: some rearrangings gives that 4 is equivalent to $u^i_{\text{max}} \subseteq (U^d - u^i_1) \cap (U^d - u^i_{\text{max}})$ if $\omega = 1$ and $u^i_{\text{max}} \subseteq (U^d + u^i_1) \cap (U^d + u^i_{\text{max}})$ if $\omega = -1$. Thus all possible values of $(j_d, \omega)$ can be found by taking the intersections $u^i_{\text{max}} \subseteq (U^d - u^i_1) \cap (U^d - u^i_{\text{max}})$ for $\omega = 1$ and $(U^d + u^i_1) \cap (U^d + u^i_{\text{max}})$ for $\omega = -1$. On the other hand, 5 depends on the parent sequence $j_{d-1}$ and so needs to be checked separately for each $j_{d-1}$ remaining in the tree.

If a node $j$ passes conditions (4) and (5), we “search” the node by computing the collaboration $C_j$. After computing the collaboration, we check two further conditions. These conditions both emerge from the fact that if a collaboration $C_j$ contains our target solution $\bar{V}^1$, then its difference set $\text{Diff}(C_j)$ must contain $W$. First, it must be that $N(C_j) \geq k$, as $C_j$ must contain enough elements to generate a difference set the same size as $W$. Once this is confirmed, we could check explicitly if $W \subseteq \text{Diff}(C_j)$; however, computing $\text{Diff}(C_j)$ and checking this condition takes $O(N(C_j)^2)$ operations. Since $N(C_j)$ can in theory be as large as $O(k^2)$, we only perform this check if $N(C_j) < c k^2$ where $c$ is a predetermined constant. One advantage of this check is that once $\text{Diff}(C_j)$ is known, it can be checked whether it is a solution (Diff($C_j$) = $W$) at virtually no additional cost. This gives MISTR the advantage that it knows at the time it returns whether it has actually found a solution, without any additional computation necessary to check.

The collaboration search can be summarized as follows: for each $d$, eliminate any nodes and their children that fail any of the above conditions. If any difference sets are computed as part of the third pruning condition, check if they equal $W$. If yes, return the collaboration indexed by that node as a solution; otherwise, proceed to depth $d + 1$. Pseudocode outlining our implementation of the collaboration search is provided in figure 2.

When $d = \tau$, one of the remaining nodes will always be $U$; it follows that the only case in which the collaboration search fails to eventually find a solution is the case that $U$ is not itself a solution because it contains a false positive vector. We prove in the results section that for $\theta < 1/2$, this probability is small for sufficiently high resolution $N$.

We now consider the time complexity of the collaboration search. Since $U^1 \subseteq W$, $N(U^1) < k^2$ for each $i$. Thus condition (3) can be checked by a sort and binary search in $O(k^2 \log(k))$ time. Computing $C_j$ involves a single intersection for each node passing (3), which takes at most $O(k^2 \log(k))$ time for each node, and the restriction on computing difference sets only if $N(C_j) < c k^2$ guarantees that this check takes at most $O(k^2 \log(k))$ time to compute the difference set and compare it with $W$. Thus as long as the number of indices searched for each depth $d$ is not too large, for each $d$ the search process can be completed in $O(k^2 \log(k))$ time. If this holds, it follows that the entire collaboration step can be performed in $O(\tau k^2 \log(k))$ time.
the same as the time complexity of the projection step. Empirical evidence that this holds in practice is provided in section IV.

C. An Example

We illustrate the application of the collaboration search step with an example. For simplicity of exposition, we pretend that $k$ is known, i.e. set $\tilde{k} = k$ for this example. We set the constant $c = 2$. Let $D = 2$, $\tau = 3$ and let $V$ be the set:

$$V = \left\{ \left( \frac{-2}{0} \right), \left( \frac{-1}{-1} \right), \left( \frac{-1}{0} \right), \left( \frac{0}{2} \right), \left( \frac{1}{-1} \right), \left( \frac{1}{0} \right), \left( \frac{2}{1} \right) \right\}$$

It follows that $k = 7$. Set $X_X = (0.911, 0.413)$, $X_2 = (0.974, 0.228)$, and $X_3 = (0.0266, 0.9996)$. After the intersection step, we are left with (in order according to projection with $X_1$, $X_2$, $X_3$ respectively):

$$U^1 = \left\{ \left( \frac{0}{0} \right), \left( \frac{1}{2} \right), \left( \frac{1}{1} \right), \left( \frac{0}{1} \right), \left( \frac{2}{0} \right), \left( \frac{1}{2} \right), \left( \frac{2}{3} \right), \left( \frac{3}{0} \right), \left( \frac{1}{1} \right) \right\}$$

$$U^2 = \left\{ \left( \frac{0}{0} \right), \left( \frac{1}{2} \right), \left( \frac{1}{1} \right), \left( \frac{0}{1} \right), \left( \frac{2}{0} \right), \left( \frac{2}{1} \right), \left( \frac{2}{3} \right), \left( \frac{3}{1} \right), \left( \frac{4}{1} \right) \right\}$$

$$U^3 = \left\{ \left( \frac{0}{0} \right), \left( \frac{-2}{1} \right), \left( \frac{0}{1} \right), \left( \frac{1}{2} \right), \left( \frac{-1}{2} \right), \left( \frac{2}{1} \right), \left( \frac{2}{3} \right), \left( \frac{-1}{3} \right), \left( \frac{1}{1} \right) \right\}$$

It is clear that none of these are equivalent solutions to $V$, so the collaboration search step is necessary. The initial setup of the search tree $\mathcal{T}$ can be seen in figure 1a.

We begin the search by checking condition 4 for $U^2$. We find that:

$$(U^2 - u_1^1) \cap (U^2 - u_{\text{max}}^1) = \emptyset$$

$$(U^2 + u_1^1) \cap (U^2 + u_{\text{max}}^1) = \left\{ \left( \frac{4}{1} \right) \right\}$$

It follows that the only valid index at this depth is $(9, -1)$, so we eliminate all other branches of $\mathcal{T}$. As condition (5) is empty for depth $d = 2$, it trivially holds. We now compute the collaboration with respect to $j_2 = (0, 1), (9, -1)$:

$$\mathcal{C}_{j_2} = U^1 \cap -\left( U^2 - \left( \frac{4}{1} \right) \right) = \left\{ \left( \frac{0}{0} \right), \left( \frac{1}{-2} \right), \left( \frac{1}{0} \right), \left( \frac{2}{-1} \right), \left( \frac{3}{-1} \right), \left( \frac{2}{2} \right), \left( \frac{3}{0} \right), \left( \frac{4}{1} \right) \right\}$$

This set has $k < 8 < 2k$ elements, so we compute Diff($\mathcal{C}_{j_2}$), and find that $W \subset $ Diff($\mathcal{C}_{j_2}$). Thus this node remains, but is not itself a solution. The configuration of $\mathcal{T}$ after this step can be seen in figure 1b.

Thus we need to proceed to depth $d = 3$, which has initial setup shown in figure 2a. We begin by checking condition 4:

$$(U^3 - u_1^1) \cap (U^3 - u_{\text{max}}^1) = \emptyset$$

$$(U^3 + u_1^1) \cap (U^3 + u_{\text{max}}^1) = \left\{ \left( \frac{2}{1} \right) \right\}$$
IEEE TRANS. ON SIGNAL PROCESSING

(a) Initial setup of search tree $T$. Each node at depth $d$ has twice as many children as $U^d + 1$ has elements.

(b) The configuration of $T$ after the collaboration search up to depth $d = 2$. All but the node $(0, 1), (9, -1)$ have been eliminated by condition 4.

Fig. 1: Collaboration search at depth $d = 2$

(a) Configuration of $T$ at depth $d = 3$.

(b) Final configuration of the search tree after collaboration search to depth $d = 3$. Only the two nodes in bold ever need to be searched.

Fig. 2: Collaboration search at depth $d = 3$.

(a) Original scatterers $V$

(b) Retrieved set $C_{j_3}$

Fig. 3: Original set of scatterers $V$ alongside recovered scatterer locations $C_{j_3}$. The two differ only by a constant shift.

This means that the only valid index at this depth is $(7, -1)$. We then confirm that condition 5 holds, i.e. that $(3, -1)$ is in $- \left( U^3 - \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right)$. Having passed conditions 4 and 5, we compute the collaboration $C_{j_3}$ for $j_3 = (0, 1), (9, -1), (7, -1)$:

$$C_{j_3} = C_{j_2} \cap - \left( U^3 - \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right) = \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 0 \end{pmatrix}, \begin{pmatrix} 4 \\ 1 \end{pmatrix} \right\}$$

The size of this set is exactly 7, so we explicitly check the difference set $\text{Diff}(C_{j_3})$ and confirm that $\text{Diff}(C_{j_3}) = W$, and thus $C_{j_3}$ is a solution. The final configuration of $T$ is shown in figure 2b, while figure 3 shows the original set $V$ mapped alongside the recovered set $C_{j_3}$.

D. A Note on Implementation

Though MISTR is best explained as above, where the three steps are performed in sequence, in practice it is faster to have MISTR run in a “breadth-first” order: for each $i$, perform a projection step, an intersection step, and a collaboration search for depth $d = i$, before proceeding to $i + 1$ if no solution is found. This avoids performing a large number of unnecessary projection steps when only a small number are required for the collaboration search to find a solution. This is the implementation used in the numerical simulations in section IV.

III. Theoretical Results

This section is dedicated to an overview of the proof of our main result. The major steps will be covered, though proofs of technical lemmas are reserved for the appendix.

A. Probabilistic Setting

We now introduce the probabilistic setting for which we will prove our theoretical guarantees. We consider an underlying continuous process for distributing points, and show how it can be reduced to a discrete setting in a natural way.

Let $s = N^{D\theta}$ where $0 < \theta < 1/2$. Let $\Phi_s$ denote the probability measure associated with a Gaussian random variable in $\mathbb{R}^D$ with mean 0 and covariance matrix $\frac{1}{2\ln(s)}I$ (this is the distribution of a standard Gaussian normalized by $1/\sqrt{2\ln s}$). We consider a Poisson point process $\Pi_s$ with intensity $s$ and underlying measure $\Phi_s$. Specifically, this means that for any set $A \subseteq \mathbb{R}^D$, the number of points $\mathcal{N}(A \cap \Pi_s)$ is distributed as a Poisson random variable with mean $s\Phi_s(A)$, and this distribution for any collection of disjoint sets will be
to guarantee that all points in \( \Pi_s \) will lie inside the ball of radius 1 with high probability as \( N \to \infty \).

From an imaging point of view, this normalization makes sure the points in \( \Pi_s \) lie inside the same image window for all values of \( N \) (with high probability). Our proofs of these theorems are such that they still hold if points were truncated outside of some fixed image window, so long as that window contains the ball \( B(0,2) \). It is possible to get at least slightly better error bounds with a growing image window, but this does not accurately model the imaging setting we wish to understand, and the main theorem on asymptotic recovery holds regardless.

Now, denote \( P_N(c) \) be the open \( \ell^\infty \) ball centered at \( c \) with radius \( 1/2N \). In keeping with our imaging analogy we will call these balls “pixels”. Our goal is to recover a discrete process \( V_s \) on \( H_N \) from its pairwise difference set, where \( V_s \) is distributed as follows:

\[
V_s = \{ z \in H_N \mid \mathcal{N}(P_N(z) \cap \Pi_s) \geq 1 \}
\]

\( V_s \) should be understood as the natural discretization of \( \Pi_s \) into cubic pixels.

With probability 1, all points in \( \Pi_s \) will lie in an \( \ell^\infty \) ball centered at a point in \( H_N \), so no points from \( \Pi_s \) are missed by this discretization process. It is clear that the pixels are disjoint, so the probabilities that distinct points \( z_1, z_2 \in H_N \) are in \( V_s \) are independent. By this independence property, we have that each \( z \in H_N \) is in \( V_s \) independently with probability \( s^2 \mathcal{P}(P_N(z)) \). With our first lemma, we remark that for sufficiently high \( N \) (which can be thought of as sufficiently high resolution), no two points in \( \Pi_s \) will fall in the same pixel:

**Lemma 1.** If \( 0 < \theta < 1/2 \), then the probability that there exists a pixel \( P_N(z^*) \) such that \( \mathcal{N}(P_N(z^*) \cap \Pi_s) \geq 2 \) is bounded by \( \frac{2}{s^2} + O\left(\frac{s^2}{\ln N} \right) \).

This lemma means that if the resolution \( N^D \) is greater than \( s^2 \) order-wise, the discretized process \( V_s \) contains the same number of points as the underlying process \( \Pi_s \) with high probability as \( N \to \infty \).

**B. Main Results**

We are now ready to state our main result, which states that the probability that MISTR fails for the discrete process \( V_s \) becomes arbitrarily small for large \( N \) as long as \( s \) grows slower than \( N^{D/2} \).

**Theorem 1.** Let \( s = N^{D\theta} \) with \( \theta = 1/2 - \delta \), \( 0 < \delta < 1/2 \). Then for any \( \tau > \frac{2}{s^2} \), \( V_s \) distributed as above, MISTR with \( \tau \) random projections recovers a set equivalent to \( V \) with probability tending to 1 as \( N \) goes to infinity.

It should be noted that this theorem is asymptotic in the resolution \( N^D \). Our numerical results in section IV show that this method achieves excellent results in practice with a relatively low (\( \approx 10-20 \)) number of random projections. Fewer projections are needed the higher the dimension, as demonstrated in figure 4b.
the convex hulls of i.i.d. copies of Gaussian processes. Let $C^k$ be the convex hull of the process consisting of $k$ i.i.d. Gaussian random vectors with mean 0 and covariance $I$. Then:

**Corollary 2** (Davydov). As $k \to \infty$, with probability 1,

\[
\frac{1}{\sqrt{2 \ln(k)}} C^k \to B(0, 1)
\]

in the Hausdorff metric. In particular, almost surely,

\[
\mathcal{H}\left(\frac{1}{\sqrt{2 \ln(k)}} C^k, B(0, 1)\right) = o\left(\frac{1}{\sqrt{\ln(k)}}\right)
\]

It follows from the corollary that $\text{Vert}(V_s)$ converges to the unit sphere $S^{D-1}$. Thus for large enough $N$, $\text{Vert}(V_s)$ is nearly uniformly distributed on $S^{D-1}$, from which we can conclude our last lemma:

**Lemma 3.** Let $s = N^{D\theta}$, $0 < \theta < 1/2$. Then as $N \to \infty$, $\text{Dist}(V_s) \to \tau$ in probability.

Theorem 1 then follows from this and lemma 2.

C. Variations on 1

We conclude this section with a remark on certain variations of the results proved above. First, similar results can be proven when the process consists of $k$ i.i.d. points instead of a Poisson point process, but the independence property of the Poisson point process makes the proof of lemma 2 much cleaner.

We consider this Gaussian process a natural way to model scatterers encountered in imaging problems, but the choice of a Gaussian measure is not essential. Our proofs of this result do not fundamentally rely on our choice of a Gaussian distribution for our underlying Poisson point process. Indeed, our results hold essentially unchanged were $\Pi_n$ replaced by any point process $\tilde{\Pi}_n$ such that the convex hull of $\tilde{\Pi}_n$ approaches a compact set with everywhere positive curvature. It is known that this holds, at least in expectation, for $k$ points uniformly distributed in a convex set with smooth boundary and everywhere positive curvature: Bárány proved in [14] that for a convex body $A$ with smooth boundary in $\mathbb{R}^D$, the expected Hausdorff distance between $A$ and the convex hull of $k$ points distributed uniformly in $A$ is approximately $(\ln(k)/k)^{1/(D+1)}$.

By contrast, technical complications arise when the limit shape of the convex hull of $V_s$ has zero curvature or has corners. In this situation, it is no longer the true that the vertices of the convex hull of $V_s$ will collect roughly uniformly along the boundary; rather, they tend to collect closer and closer to the corners of the cube as $N$ grows to infinity. This makes it difficult to prove an analogous result to lemma 3, which relies on the fact that vertices on the convex hull spread out across the entire sphere $S^{D-1}$.

IV. Numerical Simulations

The numerical simulations displayed in figures 4 and 5 were implemented using MATLAB® 2020a on a 1.2GHz Intel Core m3 processor with 8 GB RAM, 1867 MHz LPDDR3. All of the code used to generate these simulations is publicly available at https://github.com/lew347/multi-dimensional-turnpike. We set most of our simulations in three dimensions, with $D = 2$ and $D = 4$ sometimes considered to show differences in dimension.

Here, data were simulated by the following process: $s$ was fixed as a parameter. $s$ points were generated from a $D$-dimensional normal distribution with mean 0 and covariance matrix $I$. These points were multiplied by $N/\sqrt{2 \ln(s)}$, then discretized to $\mathbb{Z}^D$ by rounding coordinate-wise to the nearest integer vector. It is clear that this distribution is essentially the same as a normal distribution with mean 0 and covariance $I$, normalized by $\sqrt{2 \ln(s)}$ and discretized to $\mathbb{Z}^D$. Simulations for recovery probability were run for 1000 iterations, while those for time and collaboration depth were run for 200 iterations.

Our implementation includes an additional optimization detailed in appendix VI-D to encourage the random vectors chosen to be relatively decorrelated, which improves the chances of selecting different points in the convex hull as minima. This shortcut has the same effect as selecting additional random vectors, but the resulting algorithm spends less time performing redundant projection and intersection steps.

A. Simulation Details

1) Our first simulation, displayed in figure 4a, shows the recovery probability of MISTR when $\tau = 30$, $D = 3$ and $N^D = 10000$. Notably, this graph reflects high accuracy well above the theoretical accuracy cutoff of $\theta = 1/2$.

2) The second simulation, documented in figure 4b, shows the performance of MISTR with $s = 200$ in dimensions 2 through 4. Here $N$ is chosen such that $N^D = 40000$ for each $D$, so $\theta = 1/2$. We see that essentially perfect recovery is obtained with sufficiently high $\tau$ in every dimension, but that fewer random projections are required to achieve the same recovery percentage. This likely corresponds to the greater number of vertices in the convex hull of $V$ in higher dimensions.

3) The simulation documented in figures 5a and 5b compares how the time complexity of MISTR evolves when sparsity grows as a fixed power of $N^D$ for $D = 3$. In this simulation, $\tau = 30$ while $N$ is chosen such that $s = N^{D\theta}$ for various values of $\theta$. These figures provide empirical evidence that the average-case time complexity of MISTR is $O(\tau k^2 \log(s)) \approx O(\tau k^2 \log k)$. This can be seen most clearly in 5b, which shows average runtime plotted against $s^2$; we can see that the resulting curves are nearly linear in $s^2$. We note that no recovery failures were observed in this test for any of the plotted sparsity levels $\theta$.

4) Figure 5c, which shows the average “collaboration depth” for the same values of $\theta$ when $D = 3$. Collaboration depth refers to the depth of the collaboration search tree $T$ at which the first solution is found, including the root. A collaboration depth of 1 implies that a solution was found during the first intersection step, so no collaboration search was required. Collaboration depth is a rough measure of how hard it is to find a solution for a given set of parameters, in the sense
Fig. 4: **Recovery probability.** (a) shows how recovery probability varies with \( s \) when \( \tau = 30, D = 3 \) and \( N^D = 10000 \). (b) shows recovery probability against \( \tau \) for \( s = 200, N^D = 40000 \) in dimensions 2, 3, and 4.

Fig. 5: **Recovery time.** Figures 5a show how average runtime increases with \( s \) for different values of \( \theta \). Figure 5b shows a nearly linear relationship between average runtime and \( s^2 \). Figure 5c shows the average “collaboration depth” (how deep in the tree \( T \) one must search before a solution is found) against \( s \) for various values of \( \theta \). Figure 5d shows the average size \( K \) of the difference set \( W \) against \( s^2 \), revealing a nearly linear relationship. All figures display simulations for \( D = 3 \).
that when collaboration depth is higher, more information is needed before a solution can be found.

5) Figure 5d compares the average size of the difference set \( W \) to the squared number of scatterers \( s^2 \) over 100 samples. The graph shows a linear relationship for \( \theta \leq 1/2 \), though a slight nonlinearity for \( \theta > 1/2 \). Since \( W \) is the input to our algorithm, the simulations displayed in figures 5d and 5b corroborate our earlier assertion that MISTR runs in average-case time nearly linear in the size of the input. This means that for \( \theta \leq 1/2 \), MISTR has close to the best asymptotic time complexity one can hope for in solving the turnpike problem.

B. Discussion

Our numerical experiments verify the accuracy and showcase the speed of MISTR in practice. We observe by this experimental evidence that the average runtime of MISTR is approximately \( O\left(\tau k^2 \log(k)\right) \) as predicted in section II. Runtime increases especially slowly for lower sparsities \( \theta \); this encourages our opinion that MISTR is especially well-suited for problems when sparsity is small, as is the case in many imaging setups. Secondly, our results confirm the predicted recovery properties, and even show successful recovery for values of \( \theta \) substantially above the theoretically predicted cutoff of \( \theta < 1/2 \), even for large values of \( s \). For instance, with parameters \( s = 1000 \), \( \theta = 0.55 \), and \( \tau = 30 \), MISTR correctly recovered 1000 of 1000 test sets, despite \( \theta = 0.55 \) being significantly above the theoretical sparsity cutoff for guaranteed recovery.

We are thus unsure whether the bounds in 1 are sharp. It is possible that an analysis that better harnesses the properties of the Gaussian distribution might yield even better recovery guarantees. On the other hand, it is possible that the \( \theta \) bound is actually sharp, and recovery may not be possible for \( \theta > 1/2 \) for higher number of scatterers \( s \). The number of meaningfully different collaborations that MISTR can generate is approximately bounded by the number of points in the convex hull of \( V \), and it is shown by Bárány and Vu in [15] that the number of points in the convex hull of \( s \) \( D \)-dimensional Gaussian random variables concentrates around a mean of order \( O\left(\ln D/2(s)\right) \) as \( s \) grows large. Thus if the number of collaborations required for successful recovery grows faster than \( \ln D/2(s) \), this number will eventually exceed the number of points in the convex hull of \( V \) and thus recovery by MISTR would be impossible.

C. Comparison with one-dimensional TSPR

It is worth pausing to compare our results to the TSPR support recovery algorithm. The two algorithms employ essentially the same strategy: they attempt to gather information on the support by using multiple intersection steps. The key difference lies in how the offset vectors used in these intersection steps can be found in different dimensions. Because the structure of higher dimensions allows for nontrivial convex hulls, by taking different random projections MISTR can compute multiple different intersection step sets \( U^i \) in only \( O\left(\tau k^2 \log(k)\right) \) time. We have seen that the collaboration step can then combine the information from these initial intersection steps with the same time complexity.

By contrast, one-dimensional TSPR must resort to a “graph step” which involves computing the difference set of the result of a single intersection step, which can take up to \( O\left(k^3\right) \) time. Further, TSPR is limited in how many intersection steps it can take while avoiding false negatives: TSPR’s graph step attempts to recover the first \( p \) elements of a solution to use in \( p \) additional intersection steps. However, there is a nonzero chance that the graph step skips a member of the true support\(^4\), causing a false negative. To mitigate this risk, the number of intersection steps \( p \) must be kept low—Jaganathan et al. prove asymptotic recovery for \( 1 + \sqrt{\log(s)} \) steps. By contrast, in 3 dimensions the number of points in the convex hull of \( k \) Gaussian points grows as \( O\left(\ln^3/2(k)\right) \) [15], so MISTR can access a significantly larger number of different intersection steps, without this risk of false negatives. Thus by exploiting the geometry of multiple dimensions, MISTR achieves better accuracy, with the same guaranteed accuracy, in substantially less time than does TSPR.

V. Conclusion

In conclusion, we introduced MISTR to efficiently solve the turnpike problem in multiple dimensions, which among other applications can immediately be combined with existing convex relaxation approaches to solve the phase retrieval problem in its entirety. Further, we proved that MISTR recovers most \( O\left(N^{D(1/2-\delta)}\right) \)-sparse signals. Finally, we provided numerical evidence that verifies our theoretical results, and shows empirically that MISTR runs in \( O\left(\tau k^2 \log(k)\right) \) expected runtime. Remarkably, our empirical evidence shows that when data is distributed as Gaussian, MISTR is capable of excellent recovery well above the theoretically predicted threshold.

VI. Appendix

A. Proof of lemma 2

We begin with the proof of lemma 2. Recall that we denote \( \mathcal{G}_\tau \) the event that \( \{\ell - v_0^i, \ell - v_1^i\} \subseteq W \) for each \( i = 1, 2, \ldots, \tau \), with \( v_\tau = \{v_0^1, v_1^1, \ldots, v_\tau^1, v_0^\tau, v_1^\tau\} \).

Lemma 2. Let \( \tau \) be a finite positive integer, let \( \theta = 1/2 - \delta \) where \( 0 < \delta < 1/2 \), and let \( s = N^{D\theta} \). Suppose that \( \text{Dist}(v_\tau) = \tau > 0 \). Then the probability that there exists \( \ell \in H_N \), such that \( \ell \notin V_s \) and \( \mathcal{G}_\tau \) occurs, is bounded by:

\[
\frac{2}{N^{D\theta}} + O\left(\frac{\ln^{D\sqrt{1/4}(N)} N^{D\left(2\delta^2 - 1\right)}}{N^{D\left(2\delta^2 - 1\right)}}\right)
\]

The proof of this lemma is long, and so it makes sense to outline the general steps and approach involved. Step 1 is truncation, where we bound the probability that points in \( I_s \) lie outside the ball of radius 2 using standard techniques from the study of Gaussian processes; see e.g. [16] or [17] for comprehensive treatment of these techniques. Step 2 is coupling, where we relate \( I_s \) and thereby \( V_s \) to a homogeneous process

\(^4\)For example, it might recover \( \tilde{v}_1, \tilde{v}_2 \), and \( \tilde{v}_4 \), skipping \( \tilde{v}_3 \) and causing a false negative.
on a discrete, bounded domain. Step 3 is a combinatorial bound, which is proven in a pair of sub-lemmas 4 and 5. These two lemmas are essentially the same as lemmas VIII.4 and VIII.5 from [5], but adapted to our multidimensional setting; proofs are available in the supplementary materials for completeness.

**Proof.** Recall that $\mathcal{F}_\ell$ is the probability that $\ell \in U \mid \ell \notin V_s$; this is the event that MISTR returns $\ell$ as a false positive inclusion. Denote the event:

$$\mathcal{F} = \bigcup_{\ell \in H_N} \mathcal{F}_\ell \cap \{\ell \notin V_s\}$$

In other words, $\mathcal{F}$ is the probability that there exists an $\ell \in H_N$ which is returned by MISTR as a false positive. As previously noted, this is exactly the event in which MISTR fails, so our goal in the following is to bound $P(\mathcal{F})$.

Step 1: truncation. We begin by relating the process $V_s$ to a process on a bounded domain. First, consider the conditional distribution $\Pi^k_s = \Pi_s \mid (N'(\Pi_s) = k)$. It is clear that this is distributed as $k$ independent Gaussian random variables distributed by $\Phi_s$, while $k = N'(\Pi_s)$ is a Poisson random variable with mean $s$. It thus follows that the distribution $\frac{\sqrt{2 \ln(s)}}{\sqrt{2 \ln(\ell)}} \Pi^k_s$ is distributed as $k$ i.i.d. Gaussian random variables with mean 0 and covariance matrix $\frac{1}{\sqrt{\ln(\ell)}} I$. Denote this distribution $\Pi^k$.

It is known that for $k$ independent, identically distributed Gaussian random vectors $\{Z_i\}^k_{i=1}$ with mean 0 and covariance matrix $I$, $E \max_{i=1,...,k} \|Z_i\|^2 = \sqrt{2 \ln(k)}$. The Borell-TIS inequality, see for instance [17], gives that this maximum has sub-Gaussian tails. In particular, we have that:

$$P\left( \max_{i=1,...,k} \|Z_i\|^2 - E \max_{i=1,...,k} \|Z_i\|^2 > \sqrt{2 \ln(k)} \right) = \rho_N = 1 - \exp\left( -\frac{s \ln^{D/2}(s)}{N^D} \right)$$

Since $k$ is distributed as a Poisson random variable with mean $s$, $\sqrt{2 \ln(s)} \to 1$ almost surely by the law of large numbers. It follows that $\Pi^k \to \Pi^k_s$ almost surely in the Hausdorff metric. This plus the subexponential concentration of Poisson random variables guarantees that for $N$ large, $P(\max_{\rho \in \Pi^k_s} \|\rho\|^2 > 2) \leq \frac{2}{s}$.

Let $\Omega_s$ be the event that all points in $\Pi_s$ fall inside the ball of radius 2. By straightforward conditional probability calculations, we have that:

$$P(\mathcal{F}) = P(\mathcal{F} | \Omega_s)P(\Omega_s) + P(\mathcal{F} | \Omega^c_s)P(\Omega^c_s) \leq P(\mathcal{F} | \Omega_s) + P(\Omega^c_s) \leq P(\mathcal{F} | \Omega_s) + \frac{2}{s}$$

Thus it suffices to show that $P(\mathcal{F} | \Omega_s)$ satisfies the desired order bounds.

Step 2: Coupling. By the independence property of the Poisson process, the conditional process $\Pi_s \mid \Omega_s$ will be distributed as $\Pi_s$ on the ball of radius 2, and contain no points on or outside of this ball. We now note that since the normalized density $\Phi_s$ is bounded by $\ln^{D/2}(s)$, we can bound $\Pi_s \mid \Omega_s$ by an homogeneous Poisson point process $\pi_s$ on $B(0, 2)$ with uniform intensity $s \ln^{D/2}(s)$. Here the bound is in the sense that for any set $A \subseteq B(0, 2)$ and any $j$,

$$P(\mathcal{F}(A \cap \pi_s) \geq j \mid \Omega_s) \leq P(\mathcal{F}(A \cap \pi_s) \geq j)$$

Denote $\tilde H_N$ the set of all $z \in H_N$ such that that $B(0, 2) \cap \Pi(N(z)) = \emptyset$. We can then define $Y_s$ be the discretization of $\pi_s$ on $\tilde H_N$, where a point $z \in P_R$ is in $Y_s$ iff $\mathcal{F}(\Pi(N(z) \cap \pi_s)) \geq 1$. Since $\pi_s$ is homogeneous with intensity $s$, it follows that every point $z \in P_R$ in $\pi_s$ with independent probability at most $1 - \exp\left( -\frac{s \ln^{D/2}(s)}{N^D} \right)$.

Denote this by:

$$\rho_N = 1 - \exp\left( -\frac{s \ln^{D/2}(s)}{N^D} \right)$$

It follows that $\rho_N \to 0$ as $N \to \infty$ as long as $s < 1$.

By the above definitions, we see that the discrete, homogeneous process $Y_s$ bounds the discretized Poisson process $V_s$ in the sense of (10). Thus we know that for any $z \in P_R$

$$P(z \in V_s) \leq P(z \in Y_s) = P(\mathcal{F}(\pi_s \cap \Pi(N(z)) \geq 1) \leq \rho_N$$

In other words, each $z \in \tilde H_N$ is in $V_s$ independently with probability not exceeding $\rho_N$.

The following lemmas comprise step 3: the combinatorial bound. Due to their length, full proofs are relegated to our supplementary materials. Specifically, we bound the probability that a pixel $\ell$ appears in $\bigcap_{i=1}^k \Theta_i(U(i)) \subseteq \mathcal{I}(\tau)$ given $\ell \notin V_s$:

**Lemma 4.** Let $\tau, \theta, s$, and Dist $(\nu_s)$ be as in lemma 2. Then for any $\ell \in \tilde H_N$, the probability that $\ell \notin V_s$ and

$$\{\ell - v_0(i), \ell - v_1(i)\} \subseteq W \text{ for each } i = 1, 2, \ldots, D$$

is bounded by $O\left( \left( \frac{s \sqrt{\tau}}{N^\tau} \right) \right)$ as $N \to \infty$.

The proof of this result is a lengthy application of combinatorial reasoning, counting the various ways that a false positive $\ell$ could persist through the intersection and collaboration steps. The next lemma expands this result to apply to all $\ell$ simultaneously:

**Lemma 5.** Under the same assumptions as in lemma 4, the probability that there exists an $\ell \in \tilde H_N$, such that $\ell \notin V_s$, and $\mathcal{F}_\ell$ occurs is of order at most:

$$O\left( \frac{s \sqrt{\tau}}{N^\tau} \ln^{D/4}(N) \right) = O\left( \ln^{D/4}(\frac{4N^\tau}{\sqrt{s}}) \right)$$

The proof of this lemma involves a straightforward application of the Markov inequality. This lemmas proves the order bound for the conditional Poisson process $\Pi_s \mid \Omega_s$ on $\tilde H_N$, which with (9) completes the proof of lemma 2. $\blacksquare$
B. Proof of lemma 1

We can use the same truncation and coupling steps from the previous section to prove lemma 1:

**Lemma 1.** If $0 < \theta < 1/2$, then the probability that there exists a pixel $P_N(z^*)$ such that $\mathcal{N}(P_N(z^*) \cap \Pi_s) \geq 2$ is bounded by $\frac{2}{s} + \mathcal{O}\left(\frac{s^2 \ln^2(s)}{N^2}\right)$.

**Proof.** Consider the same homogeneous process $\pi_s$ on $B(0,2)$ with intensity $s$ as in the proof of the previous lemma. By the same arguments, it suffices to prove the order bound for $\pi_s$. For each $z \in H_N$, the probability that there are two points in $P_N(z)$ is at most:

$$\mathbb{P}(\mathcal{N}(P_N(z) \cap \Pi_s) \geq 2) \leq 1 - \exp\left(-\frac{s \ln^{D/2}(s)}{N^D}\right) - \frac{s \ln^{D/2}(s)}{N^D} \exp\left(-\frac{s \ln^{D/2}(s)}{N^D}\right) = \mathcal{O}\left(\frac{s^2 \ln^2(s)}{N^2D}\right)$$

As there are $K_N = \mathcal{O}(N^D)$ pixels in $\tilde{H}_N$, we conclude from a union bound that:

$$\mathbb{P}(\exists z \in \tilde{H}_N \text{ s.t. } \mathcal{N}(P_N(z) \cap \Pi_s) \geq 2) \leq \mathcal{O}\left(N^D \times \frac{s^2 \ln^2(s)}{N^{2D}}\right) = \mathcal{O}\left(\frac{s^2 \ln^2(s)}{N^D}\right)$$

which converges to 0 as $N$ goes to infinity since $\theta < 1/2$. The result follows.

We conclude with the proof of lemma 3, which completes the proof of theorem 1.

C. Proof of lemma 3

**Lemma 3.** Let $s = N^{D/2}$, $0 < \theta < 1/2$. Then as $N \to \infty$, $\text{Dist}(v_{\tau}) \to \tau$ in probability.

The idea of the proof is as follows. First, we recall the relationship between $\Pi_s$ and the process $\Pi^k$ consisting of a fixed number of independent, normalized Gaussian random vectors. We then apply corollary 2 to conclude that $\Pi^k$ converges almost surely to the unit ball in the Hausdorff metric. It follows that vertices of the convex hull of $\Pi^k$ converge in probability to $\mathbb{S}^{D-1}$ in the Hausdorff metric; thus these vertices collect roughly uniformly on the sphere $\mathbb{S}^{D-1}$. Therefore, the probability that the same vertex has the maximal inner product with more than one of $k$ unit vectors tends to zero as $N \to \infty$. By the relationship between $\Pi_s$ and $\Pi^k$, this also holds for vertices of the convex hull of $\Pi_s$, from which the result follows.

**Proof.** Recall the renormalized Poisson process $\Pi^k_s = (\Pi_s \mid \mathcal{N}(\Pi_s) = k)$, and let $V^k_s = (V_s \mid \mathcal{N}(\Pi_s) = k)$ and $\mathcal{D}^k_s = (\text{Dist}(v_{\tau}) \mid \mathcal{N}(\Pi_s) = k)$. By definition of the process $\Pi_s$ and $V_s$, $k$ is distributed as a Poisson random variable with mean $s = N^{D/2}$, so $k \to \infty$ as $s \to \infty$. Thus if $\mathcal{D}^k_s \to \tau$ in probability as $k \to \infty$, it follows that $\text{Dist}(v_{\tau}) \to \tau$ in probability.

We recall further that $\Pi^k = \frac{\sqrt{2 \ln(s)}}{\sqrt{2 \ln(k)}} \Pi^k_s$ is distributed as $k$ i.i.d. Gaussian random variables with covariance $\frac{1}{\sqrt{2 \ln(k)}} I$, so we can apply corollary 2 to conclude that almost surely, the convex hull of $\Pi^k$ converges to $B(0,1)$. We already saw that $\Pi^k$ converges to $\Pi^k_s$ almost surely in the Hausdorff metric, so it follows that the convex hull of $\Pi^k_s$ also converges to $B(0,1)$ almost surely.

As $V_s$ is the discretization of the Poisson process $\Pi_s$ on the grid $H_N$, as $N \to \infty$, $V_s$ becomes arbitrarily close to $\Pi_s$ in the Hausdorff metric. It follows that as $k \to \infty$, $V^k_s$ also converges to $B(0,1)$ almost surely in this metric. In particular, since the boundary of $B(0,1)$ has positive curvature at every point, we can conclude that the vertices $\text{Vert}(V^k_s) \to \mathbb{S}^{D-1}$ almost surely as $k \to \infty$.

Now, for any $\eta$ we can choose a set $S_\eta$ of finitely many points from $\mathbb{S}^{D-1}$ such that for a random vector $X$ chosen uniformly from $\mathbb{S}^{D-1}$,

$$\max_{s^* \in S_\eta} \mathbb{P}\left(\text{arg min}_{s \in S} \langle X, s^* \rangle = s^\star\right) \leq \frac{1}{\eta}$$

Since $\text{Vert}(V^k_s)$ converges to $\mathbb{S}^{D-1}$, by continuity of projections there exists $k$ large enough that we can guarantee that given two unit vectors $X_1 \neq X_2$, $\arg\min_{s \in S_\eta}(X_1, s) \neq \arg\min_{s \in S_\eta}(X_2, s)$ implies $\arg\min_{s \in V^k_s}(X_1, s) \neq \arg\min_{s \in V^k_s}(X_2, s)$. It follows that for any $\eta$, there exists $k_0$ such that for all $k > k_0$, for any $X$ distributed uniformly at random on $\mathbb{S}_+^{D-1}$:

$$\max_{a \in \text{Vert}(V^k_s)} \mathbb{P}\left(\text{arg min}_{s \in V^k_s} \langle X, v \rangle = a\right) \leq \frac{1}{\eta(k)}$$

where $\eta(k) \to \infty$ as $k \to \infty$.

We now show that this is sufficient to conclude that $\mathcal{D}^k_s \to \tau$ in probability. In order for $\mathcal{D}^k_s$ to be less than $\tau$, there must be two random vectors $X_i$ and $X_j$, $i \neq j$, which select the same minimal point. We can thus write:

$$\mathbb{P}(\mathcal{D}^k_s < \tau) \leq \sum_{i \neq j} \mathbb{P}\left(\arg\min_{v \in V^k_s} \langle X_i, v \rangle = \arg\min_{v \in V^k_s} \langle X_j, v \rangle\right)$$

Since the $X_i$’s are independent and identically distributed, by a union bound:

$$\mathbb{P}(\mathcal{D}^k_s < \tau) \leq \binom{\tau}{2} \mathbb{P}\left(\arg\min_{v \in V^k_s} \langle X_1, v \rangle = \arg\min_{v \in V^k_s} \langle X_2, v \rangle\right)$$

Thus it suffices to show that

$$\mathbb{P}\left(\arg\min_{v \in V^k_s} \langle X_1, v \rangle = \arg\min_{v \in V^k_s} \langle X_2, v \rangle\right) \to 0 \text{ as } k \to \infty$$
Denote $A_a$ the event that $\arg\min_{v \in V_k^\tau} \langle X_1, v \rangle = a$ for $a \in V_s$. Conditioning on $A_a$ we have:

$$
\mathbb{P} \left( \arg\min_{v \in V_k^\tau} \langle X_1, v \rangle = \arg\min_{v \in V_k^\tau} \langle X_2, v \rangle \right) = \sum_{a \in \text{Vert}(V_k^\tau)} \mathbb{P} \left( \arg\min_{v \in V_k^\tau} \langle X_2, v \rangle = a \ \bigg| A_a \right) \mathbb{P} \left( A_a \right) \leq \max_{a \in \text{Vert}(V_k^\tau)} \mathbb{P} \left( \arg\min_{v \in V_k^\tau} \langle X_2, v \rangle = a \right) \leq \frac{1}{\eta(k)}
$$

by (14). Since $\eta(k) \to \infty$ as $k \to \infty$, this converges to 0. This proves that $\mathbb{D}_k^\tau \to \tau$ in probability, which as previously noted is sufficient to conclude that $\text{Dist} \left( \mathbf{v}_\tau \right) \to \tau$ in probability as $N \to \infty$.

We conclude with a comment on our simplifying assumption $\overline{V}_i = V_s - v_0$. As $N$ gets large, the probability that this occurs becomes quite low, so we pause to justify this assumption in the above lemma. Without this assumption, for each $X_i$, $\mathbf{v}_\tau$ contains either $v_0 = \arg\min_{v \in V_s} \langle X_i, v \rangle$ or $v_{k-1} = \arg\max_{v \in V_s} \langle X_i, v \rangle$. Using the same argument used in the above lemma, it follows that as $N \to \infty$, the 2$\tau$ elements $\{v_0, v_{k-1}, \ldots, v_0, v_{k-1}\}$ will all be distinct with high probability. Thus, regardless of whether the max or the min is included in $\mathbf{v}_\tau$ for each $i$, it is guaranteed that $\text{Dist} \left( \mathbf{v}_\tau \right) \to \tau$ in probability as $N \to \infty$.

### D. Random vector modification

In the implementations described in section IV, we mentioned that to reduce the number of random vectors used in the algorithm, we do not draw random vectors uniformly at random from $\mathbb{S}^{D-1}$ but rather modify the distribution to select vectors that are decorrelated. This makes it more likely that each vector selects different points out of the convex hull of $V_s$, and so reduces the odds of “wasted” random vectors that do not add any new elements to $\mathbf{v}_\tau$. We describe the exact process below:

### REFERENCES


VII. SUPPLEMENTARY APPENDIX

This section contains proofs of lemmas 4 and 5, which are adaptations of lemmas VIII.4 and VIII.5 from [5].

Lemma 4. Let $\tau, \theta, s$, and $\text{Dist}(v_\tau)$ be as in lemma 2. Then for any $\ell \in \mathcal{H}_N$, the probability that $\ell \notin \mathcal{V}_s$ and
\[
\{\ell - v_0(i), \ell - v_1(i)\} \subseteq \mathcal{W} \text{ for each } i = 1, 2, \ldots, D
\]
is bounded by $O\left(\left(\frac{S^{\sqrt{D}}}{N^{\sqrt{D} + 2}}\right)\right)$ as $N \to \infty$.

Proof. Since all probabilities in this lemma are conditioned on $\text{Dist}(v_\tau) = t$, we do not write this explicitly.

We begin by noting that for fixed dimension $D$, there exists a constant depending only on dimension $C_D$ such that $\mathcal{H}_N$ contains $K_N \leq C_D 2^D N^D = O(N^D)$ points. Recall that from our earlier reasoning, each point in $\mathcal{H}_N$ is in $\mathcal{V}_s$ with probability at most $\rho_N = 1 - \exp\left(\frac{-s}{\sqrt{n}}\right) = O\left(\frac{s}{\sqrt{n}}\right)$.

Denote by $\mathcal{V}_t$ the event $\{\ell - v_0(i), \ell - v_1(i)\} \subseteq \mathcal{W}$ for each $i$. We begin by conditioning on $v_\tau = d$, where $d = (d_1, \ldots, d_t)$ ranges over all ordered sets of $t$ distinct elements in $\mathcal{H}_N$:
\[
P(\mathcal{V}_t) = \sum_d \sum_{d \in \mathcal{D}} P(\mathcal{V}_t | v_\tau = d) P(v_\tau = d)
\]

Our goal is to show that the probability is small that all the events $\{\ell - d_i\} \in \mathcal{W}$ occur simultaneously. Our strategy follows three steps: first, we prove that interpoint differences among points in $d$ will be unique with high probability as $N \to \infty$. Then, by a counting argument, we show that when differences between elements in $d$ are unique, $P(\mathcal{V}_t | v_\tau = d)$ is small with high probability as $N \to \infty$. We then conclude the result with some conditional probability computations.

First, we note that interpoint differences among points in $d$ will be unique with probability arbitrarily close to 1 for large enough $N$. We bound this probability as follows. Denote the event $\mathcal{U}_\eta$ the event in which the points $d_1, \ldots, d_\eta$ have unique pairwise differences. Then:
\[
P(\mathcal{U}_\eta) = \prod_{\eta=1}^{t} P(\mathcal{U}_\eta | \mathcal{U}_{\eta-1})
\]

We now bound $P(\mathcal{U}_\eta | \mathcal{U}_{\eta-1})$. By conditioning on $\mathcal{U}_{\eta-1}$, we know that pairwise differences not involving $d_\eta$ will be unique. For a fixed choice of $d_1, \ldots, d_{\eta-1}, d_\eta$ with each index less than $\eta$, we have that $d_\eta - d_i = d_h - d_i \iff d_\eta = d_i + d_h - d_i$. This probability is upper bounded by $\rho_N$ as in (11). Unfixing $i, j, h, \eta$, there are at most $(\eta - 1)^3$ choices for $i, j, h$, and $\eta$, so the probability that $\mathcal{U}_\eta | \mathcal{U}_{\eta-1}$ fails is upper bounded by $(\eta - 1)^3 \rho_N$. It follows that:
\[
P(\mathcal{U}_\eta) = \prod_{\eta=1}^{t} P(\mathcal{U}_\eta | \mathcal{U}_{\eta-1}) \geq \prod_{\eta=3}^{t} \left(1 - (\eta - 1)^3 \rho_N\right)
\]

Since $t$ is finite and $\rho_N \to 0$, $P(\mathcal{U}_\eta)$ converges to 1 as $N \to \infty$. Thus we conclude that the pairwise differences in $d$ are unique with high probability for $N$ large. Denote by $\mathcal{D}$ the set of all possible $d$ that have unique pairwise differences. We can now bound (16) by:
\[
P(\mathcal{V}_t) \leq O(\rho_N) + \sum_{d \in \mathcal{D}} P(\mathcal{V}_t | v_\tau = d) P(v_\tau = d)
\]

From this it is clear that it suffices to prove the result for $d \in \mathcal{D}$. We now proceed with the counting argument to bound $P(\mathcal{V}_t | v_\tau = d)$ for $d \in \mathcal{D}$. We say that a pair of points $v_1, v_2 \in \mathcal{V}_s$ is an explaining pair for a difference $\ell - d_i$ if $v_1 - v_2 = \ell - d_i$. We will also write that a point $v$ explains a difference to mean that $v$ is a member of an explaining pair for that difference.

When $d \in \mathcal{D}$, the points in $d$ can explain at most $t/2$ differences by the interpoint differences among themselves. To see this, suppose that $d_i - d_j = \ell - d_i - d_k$ and $d_i - d_j = \ell - d_i - d_k$. By adding these, $d_j - d_k = d_j - d_k$, which happens with arbitrarily low probability for large enough $n$.

Since there are at most $t/2$ disjoint pairs of elements in $d$, it follows that pairs of points in $d$ can be explaining pairs for at most $t/2$ differences.

Thus at least $t/2$ differences must be explained by other points in $\mathcal{V}_s$. This can only happen due to one of the following cases:

(i) There exists $g \in \mathcal{V}_s$ that forms at least two explaining pairs with different elements in $d$. Suppose that $g - d_1 = \ell - d_1$, and $g - d_2 = \ell - d_2$. Adding these, this implies $d_1 - d_2 = d_1 - d_2$, which contradicts $d \in \mathcal{D}$ unless $\ell = g$. Thus the probability of this case is $P(\ell \in \mathcal{V}_s)$.

Let $\mathcal{G}$ be the set of points in $\mathcal{H}_N$ that can form an explaining pair of a difference $\ell - d_i$ using only points from $d$. Each $g$ in $\mathcal{G}$ can be form an explaining pair with at most one point in $d$, as otherwise this would put us in case (i). For a point $g$ to be in this set, it must satisfy $g - d_i = \ell - d_i$ for some $i, j$. As there are at most $t$ choices each for $d_i$ and $d_j$, $\mathcal{G}$ is composed of at most $t^2$ points.

(ii) We consider the case when $\eta \geq t/4$ differences are explained by explaining pairs with one point in $\mathcal{G}$ and one point in $d$. For each such difference, a different $g \in \mathcal{G}$ must be chosen. Since there are at most $t^2$ points in $\mathcal{G}$, there are at most $t^2 \rho_N$ ways to choose $\eta$ points in $\mathcal{G}$, each of which will be in $\mathcal{V}_s$ with probability at most $\rho_N$. Thus the probability of this case is bounded by:
\[
\sum_{\eta=t/4} t^2 \rho_N^{\eta} \leq t^2 \rho_N^{t/4}
\]

(iii) At least $t/4$ differences are explained by pairs of points not involving $d$. There are two possibilities:

a) There exist at least $t/4$ points $\{g_1, g_2, \ldots, g_{t/4}\}$ in $\mathcal{V}_s$ such that $\{g_1, g_1 + \ell - d_{p_1}, \ldots, g_{t/4} + \ell - d_{p_{t/4}}\}$ are all in $\mathcal{V}_s$ and are distinct. Given $g_1 \in \mathcal{V}_s$, the probability that $g_1 + \ell - d_{p_i}$ is in $\mathcal{V}_s$ is bounded by $\rho_N$. Unfixing $g_1$ by a union bound, as there are $K_N$ possible choices for $g_1$ in $\mathcal{H}_N$, the probability that there exists $g_1 + \ell - d_{p_i}$ both in $\mathcal{V}_s$ is upper bounded by $K_N \rho_N$. By independence, the probability that this case occurs is at most $(K_N \rho_N)^{t/4}$.

b) There exist at least $t/4$ points $\{g_1, g_2, \ldots, g_{t/4}\}$ in $\mathcal{V}_s$ such that $\{g_1, g_1 + \ell - d_{p_1}, \ldots, g_{t/4}, g + \ell - d_{p_{t/4}}\}$ are all in $\mathcal{V}_s$, but are not distinct. Let $\eta$ be the number of distinct points in the latter
that case (iii)b gives the weakest asymptotic bound in terms of $\ell/T$. Jaganathan et al. compute in [5] that there are at most $5^{t/2}$ ways these points can overlap, and at most $\eta/2$ points that are not determined by their overlap with others. Since these can each be chosen each in $K_N$ ways, we have that the probability of this case is at most:

$$\sum_{n=\sqrt{T/2}}^{2t} 5^{t/2}(K_N)^{n/2}(\rho_N)^n \leq 2t(5^{t/2})(K_N)^{1/2}\rho_N\sqrt{T/2}$$

Recalling that $K_N = O(N^D)$ and $\rho_N = O(\sqrt{T/2})$, we see that case (iii)b gives the weakest asymptotic bound in terms of $s, N$, and $t$. It follows that the probability that $\mathcal{F}_\ell$ occurs, given $v_\ell = d$, is bounded by:

$$\Pr(\mathcal{F}_\ell \mid v_\ell = d) \leq \Pr(\ell \in V_s) + O\left(\frac{s \ln D/2(s)}{N^D/2}\sqrt{T/2}\right) = \Pr(\ell \in V_s) + O\left(\frac{s^2}{N^D}\sqrt{T/4}\ln D\sqrt{T/4}(s)\right)$$

Then from (17), we have:

$$\Pr(\mathcal{F}_\ell) \leq O(\rho_N) + \sum_{d \in D} \Pr(v_\ell = d) \times \left(\Pr(\ell \in V_s) + O\left(\frac{s^2}{N^D}\sqrt{T/4}\ln D\sqrt{T/4}(s)\right)\right) \leq O\left(\frac{s \ln D/2(s)}{N^D}\right) + \Pr(\ell \in V_s) + O\left(\frac{s^2}{N^D}\sqrt{T/4}\ln D\sqrt{T/4}(s)\right) \sum_{d \in D} \Pr(v_\ell = d) \leq \Pr(\ell \in V_s) + O\left(\frac{s^2}{N^D}\sqrt{T/4}\ln D\sqrt{T/4}(s)\right)$$

since $\sum_{d \in D} \Pr(v_\ell = d) \leq 1$.

We are almost finished. Now, note that we can write

$$\Pr(\mathcal{F}_\ell) = \Pr(\mathcal{F}_\ell \mid \ell \in V_s)\Pr(\ell \in V_s) + \Pr(\mathcal{F}_\ell \mid \ell \notin V_s)\Pr(\ell \notin V_s)$$

Since $\Pr(\mathcal{F}_\ell \mid \ell \in V_s) = 1$, we have:

$$\Pr(\mathcal{F}_\ell \mid \ell \notin V_s) = \frac{\Pr(\mathcal{F}_\ell) - \Pr(\ell \in V_s)}{\Pr(\ell \notin V_s)}$$

Thus by plugging in (18), and noting that $\Pr(\ell \notin V_s) \to 1$ as $N \to \infty$, we conclude:

$$\Pr(\mathcal{F}_\ell \mid \ell \notin V_s) \leq O\left(\frac{s^2}{N^D}\sqrt{T/4}\ln D\sqrt{T/4}(s)\right)$$

This completes the proof of lemma 4.

**Lemma 5.** Under the same assumptions as in lemma 4, the probability that there exists an $\ell \in \mathcal{F}_\ell$, such that $\ell \notin V_s$ and $\mathcal{F}_\ell$ occurs is of order at most:

$$O\left(\frac{s^{\sqrt{T/2}}\ln D\sqrt{T/4}(s)}{N^D(\sqrt{T/4}-1)}\right) = O\left(\frac{\ln D\sqrt{T/4}(N)}{N^D(\sqrt{T/4}-1)}\right)$$

**Proof.** Let $L$ be a random variable defined as the number of false positives; that is, the number of $\ell$ such that $\ell \notin V_s$ but $\mathcal{F}_\ell$ occurs. Under this definition, we need to bound $\Pr(L \geq 1)$, which we do as follows:

$$E[L] \leq \sum_{\ell \in \mathcal{F}_\ell} \Pr(\mathcal{F}_\ell \mid \ell \notin V_s) \leq K_N \times O\left(\frac{s^{\sqrt{T/2}}}{N^D\sqrt{T/4}}\ln D\sqrt{T/4}(s)\right)$$

By the Markov inequality and substituting $K_N = O(N^D)$ it follows that:

$$\Pr(L \geq 1) \leq O\left(K_N \times \frac{s^{\sqrt{T/2}}}{N^D\sqrt{T/4}}\ln D\sqrt{T/4}(s)\right) = O\left(\frac{s^{\sqrt{T/2}}}{N^D(\sqrt{T/4}-1)}\ln D\sqrt{T/4}(s)\right)$$

Expanding $s$ in terms of $N$ gives the result. \(\square\)