Fast Metric Decompositions in High Dimension

Robert Krauthgamer
The Harry Weinrebe Professorial Chair of Computer Science,
Weizmann Institute of Science, Rehovot, Israel

Asaf Petruschka
Weizmann Institute of Science, Rehovot, Israel

Nir Petruschka
Weizmann Institute of Science, Rehovot, Israel

Abstract

Metric decompositions are a fundamental tool in the design of algorithms involving distances. We study fast algorithms for sampling from probabilistic metric decompositions of n-point sets in ℓ_∞ and ℓ_2 spaces of high dimension d. For ℓ_∞, we design a padded-decomposition algorithm that runs in time $\tilde{O}(nd^2)$, which is near-linear in n, and achieves padding parameter $\tilde{O}(\log n)$. Our algorithm constructs a new sparse neighborhood cover that is based on geometric properties of ℓ_∞ [Indyk, JCSS'01], and utilizes recent reductions between covers and decompositions [Conroy and Filtser, STOC'25]. For ℓ_2, we design a separating-decomposition algorithm that achieves near optimal separation $\tilde{O}(\sqrt{\log n})$ in almost-linear time $n^{1+o(1)}$. Our bounds improve over known algorithms with similar running time by a factor $\Omega(\sqrt{\log n})$, and the techniques have additional applications to spanners and nearest-neighbor search.

2012 ACM Subject Classification Theory of computation → Computational geometry; Theory of computation → Sparsification and spanners

Keywords and phrases Metric Spaces, Separating Decomposition, Padded Decomposition

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.95

Funding *Robert Krauthgamer*: Work partially supported by the Israel Science Foundation grant #1336/23.
Asaf Petruschka: Supported by an Azrieli Foundation fellowship.

Acknowledgements We thank an anonymous reviewer for useful comments.

1 Introduction

Metric decompositions are fundamental algorithmic primitives when dealing with data that involves distances. Informally, a decomposition refers to a probabilistic partition of the points into low-diameter *clusters*, such that close-by points "tend" to be clustered together. Decompositions are a very useful tool in divide-and-conquer algorithms, which first handle each low-diameter cluster separately, and then combine the solutions into a global one. These decompositions have numerous applications, including metric embeddings [8, 50, 25, 35, 41, 22], approximation algorithms for cut problems [43, 24, 19], vertex sparsification [15, 18, 44, 32, 21], spanners, distance oracles and routing [1, 46, 17, 26, 23, 27], parallel algorithms for linear algebra [11, 48], fast algorithms for single-source shortest paths [9, 42], and spectral graph theory [33, 10].

Modern applications are scaling rapidly, and current datasets are often massive and high-dimensional. As exceeding linear time significantly is simply prohibitive, this creates a strong demand for *fast algorithms*. Prior work on fast algorithms for metric decompositions has addressed graph metrics, whereas we study the geometric setting of points in $\mathbb{R}^d$. This line of work aims to match the optimal quality bounds but using algorithms whose running time is close to linear (in the input size).

We consider two classical notions of probabilistic metric decompositions: *separating decompositions* (also known as *Lipschitz decompositions*) and *padded decompositions*, defined as follows. Let (X, d_X) be a metric space. For a partition $\mathcal{P}$ of X, we call each element of $\mathcal{P}$ (which is a subset of X) a *cluster*, and we denote by $\mathcal{P}(x)$ the cluster containing point $x \in X$. The partition $\mathcal{P}$ is called Δ-*bounded* if every cluster in it has diameter at most $\Delta > 0$, i.e., the distance between every two points in the same cluster is at most Δ.

▶ **Definition 1.1** (Separating Decomposition [8]). *A distribution $\mathcal{D}$ over Δ-bounded partitions of X is called an* (α, Δ)-separating decomposition *if*

$$\forall x, y \in X, \qquad \Pr_{\mathcal{P}\sim\mathcal{D}}[\mathcal{P}(x) \neq \mathcal{P}(y)] \leq \alpha \cdot \frac{d_X(x,y)}{\Delta}.$$

▶ **Definition 1.2** (Padded Decomposition [50, 34, 20]). *A distribution $\mathcal{D}$ over Δ-bounded partitions of X is called a* (β, δ, Δ)-padded decomposition *if*

$$\forall x \in X, \gamma \in [0, \delta], \qquad \Pr_{\mathcal{P}\sim\mathcal{D}}[B(x, \gamma\Delta) \subseteq \mathcal{P}(x)] \geq e^{-\beta\gamma},$$

where $B(x, r) := \{y \in X \mid d_X(x, y) \leq r\}$ is the closed ball of radius $r \geq 0$ around x.

We note that some literature uses variants of these definitions; for example, it is common to define padded decomposition using only the case $\gamma = \frac{\ln 2}{\beta}$, and then δ is not needed. Under our definitions above, every (β, δ, Δ)-padded decomposition is also a $(\max\{\beta, \delta^{-1}\}, \Delta)$-separating decomposition. To see this, consider $x, y \in X$; if $d_X(x, y) \geq \delta\Delta$ then $\Pr[\mathcal{P}(x) \neq \mathcal{P}(y)] \leq 1 \leq \frac{d_X(x,y)}{\delta\Delta}$; otherwise, $\Pr[\mathcal{P}(x) \neq \mathcal{P}(y)] \leq \Pr[B(x, d_X(x, y)) \not\subseteq \mathcal{P}(x)] \leq 1 - e^{-\beta\frac{d_X(x,y)}{\Delta}} \leq \beta\frac{d_X(x,y)}{\Delta}$. An earlier connection established in [40, Theorem 2.2] shows that even a (β, δ, Δ)-padded decomposition only for the case $\gamma = \frac{\ln 2}{\beta}$ implies a $(4\beta, 2\Delta)$-separating decomposition for the same metric.

We study finite datasets X of size $n = |X|$ that lie in $\mathbb{R}^d$ equipped with the ℓ_∞ or ℓ_2 norm (which we denote by ℓ_∞^d and ℓ_2^d). We focus on the high-dimensional regime, namely $d \geq \omega(\log n)$, in which case running time that is exponential in d is prohibitive, as it is super-polynomial in n. For ease of presentation, we use the following terminology. A *padded-decomposition algorithm of quality* $q \geq 1$ is an algorithm that, given a metric space (X, d_X) and a diameter bound $\Delta > 0$, outputs a partition sampled from a (β, δ, Δ)-padded decomposition with $\max\{\beta, \delta^{-1}\} \leq q$. The analogous definition regarding (α, Δ)-separating decomposition requires $\alpha \leq q$. Throughout, the notation $\tilde{O}(f)$ hides a polylogarithmic factor in f. The term *near-linear* in f refers to $\tilde{O}(f)$, whereas *almost-linear* refers to $f^{1+o(1)}$; they usually refer to running time, in our case mostly with respect to n, with polynomial dependence on d.

The spanner approach. Fast algorithms for metric decompositions have already been devised for *graphs*. Specifically, for input graphs G with m weighted edges, there are $\tilde{O}(m)$-time algorithms for padded (and separating) decomposition of optimal quality $O(\log n)$ [47, 7, 48, 16]. Thus, a natural approach is to approximate (X, d_X) by a *spanner* G_X, which is a sparse graph on the vertex set X, and find a decomposition of this graph G_X. State-of-the-art constructions of geometric spanners [26, 23, 5, 36, 38, 49] yield surprisingly strong results. While this approach serves as a good *baseline*, it has some limitations. First, the intermediate step (constructing a spanner) may be overly restrictive and thus preclude achieving a tight quality bound. Specifically, one factor going into the overall quality of the decomposition is the spanner's stretch, which is often $\log^c n$ for some fixed $c > 0$. Second, by passing to a

Table 1 Comparison of metric decompositions for n-point metrics, focusing on the quality achieved in time that is polynomial vs. almost-linear vs. near-linear in n (ignoring the dependence on d which is polynomial).

Metric	Decomposition	Time	Quality	Reference
general	padded	polynomial	$O(\log n)$	[8]
graph	padded	near-linear	$O(\log n)$	[47]
ℓ_∞^d	padded	polynomial	$O(\log n)$	[8]
	padded	almost-linear	$O(\log n \cdot \log^4 d)$	via spanners [38] + [47]
	padded	near-linear	$O(\log^{3/2} n \cdot \log^4 d)$	via spanners [38] + [47]
	padded	near-linear	$O(\log n \cdot \log\log d)$	Theorem 1.3
ℓ_2^d	separating	polynomial	$O(\sqrt{\log n})$	[14]
	separating	almost-linear	$\tilde{O}(\sqrt{\log n})$	Theorem 1.4
	separating	near-linear	$O(\log n)$	via randomly shifted grid

graph representation, we effectively "forget" the geometric properties of (X, d_X) and instead treat it as an arbitrary n-point metric, for which the optimal quality is $O(\log n)$ even for sparse graphs [8], and this factor propagates into the overall quality of the decomposition. To illustrate this issue, observe that for $X \subseteq \ell_2^d$, the spanner approach can achieve at best quality $O(\log n)$, even though a separating decomposition of quality $O(\sqrt{\log n})$ is known to exist [14]. We seek to circumvent these limitations by leveraging the dataset's geometry directly.

The shifted-grid approach. Another standard approach for fast construction of separating decompositions in ℓ_p^d spaces is to partition the dataset using a randomly shifted grid. This approach can be implemented in $O(dn)$ time and yields a decomposition of quality $O(d)$, which is suboptimal for the high-dimensional regime $d = \omega(\log n)$. This approach can nevertheless be used in ℓ_2^d, even when $d = \omega(\log n)$, because the dimension can be reduced to $O(\log n)$ by the JL lemma [31]. A technical issue that arises is that the JL map has a small failure probability, which might affect the correctness of the procedure (e.g., for very close-by points). We resolve it by verifying in near-linear-time that the decomposition has succeeded; see the proof of Theorem 1.4 for details. Perhaps a lesser-known fact is that the shifted-grid approach can be used in ℓ_1^d even when $d = \omega(\log n)$, despite the impossibility of dimension reduction in this space [12, 39]. To construct an $(O(\log n), \Delta)$-separating decomposition, set the grid sidelength to $\frac{\Delta}{3 \log n}$. For every pair of data points x, y, the probability they are separated by the grid is at most $O(\log n)\frac{\|x-y\|_1}{\Delta}$, and the probability they are clustered together is at most $n^{-\frac{3\|x-y\|_1}{\Delta}}$. The crux is that even though the ℓ_1-diameter of a grid cell is larger than Δ, the diameter of a cluster is actually determined by the data points it contains; each pair of data points whose distance exceeds Δ is clustered together with probability at most $\frac{1}{n^3}$, and thus by a union bound over all pairs, with high probability, no such pair is clustered together.

We proceed to describe our results, which are also summarized in Table 1 in comparison to known or baseline bounds.

ℓ_∞ metrics. Our main result is a near-linear-time padded-decomposition algorithm for ℓ_∞ with near-optimal quality $\tilde{O}(\log n)$. We prove the following in Section 2.

▶ **Theorem 1.3.** *There is a padded-decomposition algorithm for n-point metrics in ℓ_∞^d, that achieves quality $O(\log n \cdot \log\log d)$ in running time $\tilde{O}(nd^2)$.*

Note that by a classical result of Matousek [45], every n-point metric embeds into $\ell_\infty^{n^\epsilon}$ with distortion $O(1/\epsilon)$,[1] and this implies, by a result of [8], that some n-point metrics in $\ell_\infty^{n^\epsilon}$ require quality $\Omega(\epsilon \log n)$. For fixed $\epsilon > 0$, our Theorem 1.3 nearly matches this lower bound (with running time $\tilde{O}(n^{1+2\epsilon})$). For comparison with the baseline, current $o(n^2)$-time spanner constructions [38][2] have stretch roughly $O(\log^4 d) = O(\log^4 n)$, and thus a baseline padded-decomposition algorithm only achieves a far worse quality $O(\log^5 n)$.

The algorithm of Theorem 1.3 goes through the closely-related notion of *sparse neighborhood cover* [6, 7] (for formal definition see Definition 2.1), which was recently shown to yield a padded decomposition [16], and is also of independent interest. Indeed, we provide in Section 2.4 an application of our sparse neighborhood cover to fast spanner construction (with near-optimal stretch-size tradeoff) in ℓ_∞^d for $d = n^\epsilon$.

ℓ_2 metrics. Our result for ℓ_2 is an almost-linear-time algorithm that achieves near-optimal quality, namely $\tilde{O}(\sqrt{\log n})$. When the algorithm's parameters are tuned to run in near-linear time, the algorithm essentially amounts to applying the folklore shifted-grid approach.

▶ **Theorem 1.4.** *Fix $0 < \rho < 1$. There is a separating decomposition algorithm for n-point metrics in ℓ_2^d with quality $O(\sqrt{\rho^{-1} \log n \log\log n})$ and running time $\tilde{O}(n^{1+\rho} + d^2 n)$. The algorithm is Las Vegas, meaning that with high probability it reports a partition sampled from a separating decomposition, and with the remaining probability it reports "Fail".*

An optimal quality bound $O(\sqrt{\log n})$ was established in [14] without an explicit analysis of its algorithmic efficiency. A naive algorithmic implementation of their proof runs in time $\tilde{O}(n^3 + dn)$, which can be improved to $\tilde{O}(n^2 + dn)$ using standard methods, as explained in [51, Section 2]. However, their more sophisticated improvement of the running time to $\tilde{O}(n^{1.51} + dn)$ using Locality Sensitive Hashing (LSH), claimed in [51, Section 3], is unfortunately flawed.

We remark that Theorem 1.4 implies also a fast separating-decomposition algorithm for ℓ_p metrics, $p \in (2, \infty]$, by just applying known results of [36, 38, 49]. These results employ a sequence of mappings that essentially reduce the problem of constructing a separating decomposition in ℓ_p to the same problem in ℓ_2. We can simply replace the algorithm of [14] for ℓ_2 with ours, and thereby speed up the overall running time.

The proof of Theorem 1.4 is a surprisingly simple reduction to the low-dimensional setting for $\ell_2^{d'}$, for which we use a separating-decomposition algorithm of Andoni and Indyk [2]. In a nutshell, the reduction divides the coordinates into blocks of a carefully chosen length and works on each block separately. We additionally show in Section 3.1 that the same simple idea can be applied also to the *nearest-neighbor search* problem (NNS), yielding new deterministic and space-efficient data structures.

---

[1] A metric space (X, d_X) embeds into a metric space (Y, d_Y) with *distortion* $k \geq 1$ if there exist a function $f : X \to Y$ and $s > 0$ such that for all $x, y \in X$, $\frac{s}{k} \cdot d_X(x, y) \leq d_Y(f(x), f(y)) \leq s \cdot d_X(x, y)$.

[2] We note that [49] show the existence of spanners in ℓ_∞ with size $\tilde{O}(n)$ and stretch $O(\sqrt{\log n} \log^2 d)$, but their running time is not analyzed.

2 Padded Decomposition in ℓ_∞^d

This section proves Theorem 1.3 by presenting our padded-decompositions algorithm for a dataset $X \subseteq \ell_\infty^d$. We use d_X to denote the induced metric on a subset $X \subset \ell_\infty^d$. The closed ball of radius $r \geq 0$ around x in X is denoted $B_X(x,r) := \{y \in X : \|x-y\|_\infty \leq r\}$. We also abbreviate $B_{\ell_\infty^d}(x,r)$ as $B_\infty(x,r)$.

2.1 Sparse Neighborhood Covers

Our algorithm goes through a *sparse neighborhood cover* [6], which is a deterministic counterpart of padded decomposition, defined as follows.

▶ **Definition 2.1** (Neighborhood Cover). *A collection $\mathcal{C}$ of subsets of X (called* clusters*) is a* (β, s, Δ)-neighborhood cover *if the following hold:*

- *(Cluster Diameter) Each cluster $C \in \mathcal{C}$ has diameter at most Δ.*
- *(Covering) For each $x \in X$ there exists a cluster $C \in \mathcal{C}$ such that $B_X(x, \frac{\Delta}{\beta}) \subseteq C$.*
- *(Sparsity) Each $x \in X$ is contained in at most s clusters from $\mathcal{C}$.*

A common way to construct a neighborhood cover of sparsity s is to take s independent partitions drawn from a padded decomposition. Recently, Conroy and Filtser [16] showed that the other direction also holds: Given a (β, s, Δ)-neighborhood cover, one can generate a $(O(\beta \log s), \Omega(\frac{1}{\beta}), \Delta)$-padded decomposition. Using their reduction, our main technical contribution is a fast algorithm for sparse neighborhood covers in ℓ_∞^d, which, in fact, covers the dataset with "continuous" axis-aligned rectangles in $\mathbb{R}^d$.

▶ **Definition 2.2.** *An axis-aligned rectangle in $\mathbb{R}^d$ is a set of the from $R = I_1 \times \cdots \times I_d$, where each I_i is an interval in $\mathbb{R}$. (We allow intervals of all forms $[a,b], (a,b], [a,b), (a,b)$.) The* width *of R is the length of the longest interval among $I_1, \ldots, I_d$. Note that the intersection of two axis-aligned rectangles is another axis-aligned rectangle (or empty), and every ball $B_\infty(x,r)$ in ℓ_∞^d is an axis-aligned rectangle of width $2r$.*

▶ **Theorem 2.3.** *There is an algorithm that, given dataset $X \subseteq \ell_\infty^d$ of size $|X| = n$, diameter bound $\Delta > 0$ and parameter $0 < \rho < 1$, outputs a collection $\mathcal{R}$ of axis-aligned rectangles with associated cluster collection $\mathcal{C}(\mathcal{R}) = \{C(R) := X \cap R\}_{R \in \mathcal{R}}$ such that the following properties hold true. For $s = O(n^\rho \cdot \rho^{-1} \log n)$ and $\beta = O(\rho^{-1} \log\log d)$:*

- *(Diameter) Each rectangle $R \in \mathcal{R}$ has width at most Δ.*
- *(Covering) For each $x \in X$, there is covering rectangle $R \in \mathcal{R}$ such that $B_\infty(x, \frac{\Delta}{\beta}) \subseteq R$.*
- *(Sparsity) Each $x \in X$ is contained in at most s rectangles from $\mathcal{R}$.*

In particular, the clusters collection $\mathcal{C}(\mathcal{R})$ forms a (β, s, Δ)-neighborhood cover of X. The algorithm is deterministic and runs in time $O(n^{1+\rho} \cdot \rho^{-1} d^2 \log n)$.

We now focus on the proof of Theorem 2.3; later, in Section 2.3, we explain how it is used through the reduction of [16] to get the padded decomposition algorithm of Theorem 1.3.

Terminal reduction. The algorithm is based on a *terminal reduction* procedure. Given a set of terminals T within X, this procedure constructs a sparse collection of rectangles which covers a large fraction of the given terminals, leaving the uncovered ones as the new, smaller terminal set. The formal guarantees are stated in the following lemma:

▶ **Lemma 2.4** (Terminal Reduction). *There is an algorithm* Cover(D, X, T, ρ)*, whose input consists of axis-aligned rectangular domain $D = I_1 \times \cdots \times I_d \subseteq \mathbb{R}^d$, dataset X of n points from D, terminal set $T \subseteq X$ of points with ℓ_∞-distance more than 1 from the boundary of D, diameter bound $\Delta > 0$, and parameter $0 < \rho < 1$.*

It outputs a new terminal set $T^{\texttt{new}} \subseteq T$ such that $|T^{\texttt{new}}| \leq |T| - |T|^{1-\rho}$, along with a collection $\mathcal{R}$ of axis-aligned rectangles with associated clusters $\mathcal{C}(\mathcal{R}) = \{X \cap R\}_{R\in\mathcal{R}}$, such that each rectangle in $\mathcal{R}$ is contained within the domain D, and the following hold:

- *(Diameter) Each rectangle $R \in \mathcal{R}$ has width at most Δ.*
- *(Covering) For each $x \in T \setminus T^{\texttt{new}}$, there is some $R \in \mathcal{R}$ such that $B_\infty(x, \frac{\Delta}{\beta}) \subseteq R$, where $\beta = O(\frac{\log\log d}{\rho})$.*
- *(Sparsity) The collection $\mathcal{R}$ can be partitioned into at most $\log|T| + 1$ sub-collections, where each sub-collection consists of pairwise-disjoint rectangles.*

The algorithm is deterministic and runs in $O(n \cdot d^2 \log|T|)$ time.

Theorem 2.3 follows by repeated invocations of Lemma 2.4, as follows:

Proof of Theorem 2.3. By scaling the dataset, we may assume that $\Delta = \beta = O(\frac{\log\log d}{\rho})$. Define $M = \max_{x\in X}\{\|x\|_\infty\}$ and set the domain as $D = [-(M+2), M+2]^d$. The algorithm works in terminal reduction steps. Initially, $T_0 = X$. In step i, apply $\text{Cover}(D, X, T_{i-1}, \rho)$ of Lemma 2.4 to obtain new terminal set $T_i \subseteq T_{i-1}$ and rectangle collection $\mathcal{R}_i$. The process halts at the first step k where $T_k = \emptyset$. Since $|T_i| \leq |T_{i-1}| - |T_{i-1}|^{1-\rho}$ and $|T_0| = n$, we have $k = O(n^\rho \cdot \rho^{-1})$ (see [6, Claim 3.3]). We let $\mathcal{R} = \mathcal{R}_1 \cup \cdots \cup \mathcal{R}_k$.

- (Diameter) Lemma 2.4 ensures that every rectangle in every $\mathcal{R}_i$ has width at most Δ.
- (Covering) Let $x \in X$, and consider the last step i such that $x \in T_{i-1} \setminus T_i$. Then Lemma 2.4 implies that there is some $R \in \mathcal{R}_i$ such that $B_\infty(x, \frac{\Delta}{\beta}) = B_\infty(x, 1) \subseteq R$.
- (Sparsity) Lemma 2.4 ensures that each point $x \in X$ belongs to $O(\log n)$ rectangles from each $\mathcal{R}_i$, and hence only to $O(k \log n) = O(n^\rho \cdot \rho^{-1} \log n)$ rectangles overall.

Each of the $k = O(n^\rho \cdot \rho^{-1})$ invocations of Lemma 2.4 takes $O(n \cdot d^2 \log n)$ time, so the total time is $O(n^{1+\rho} \cdot \rho^{-1} d^2 \log n)$. ◀

2.2 Proof of Lemma 2.4

The main remaining part is thus proving Lemma 2.4. To this end, we utilize powerful tools for ℓ_∞^d that were presented by Indyk in his work on the Nearest-Neighbor Search problem [29].

Informally, Indyk proved that given a finite terminal set in ℓ_∞^d, one can either (a) find a dense box (i.e., ℓ_∞-ball) of small radius that contains many terminals, or (b) find an axis-aligned hyperplane that has only few terminals close to it, and bisects the rest of the terminals in a roughly balanced way.

▶ **Definition 2.5.** *Let $T \subseteq \ell_\infty^d$, and $\alpha, \beta, \gamma \in [0,1]$ such that $\alpha + \beta + \gamma = 1$. An (α, β, γ)-separator of T is an axis-aligned hyperplane $H_i(t) = \{x \in \mathbb{R}^d \mid x_i = t\}$ such that:*

- *$T_{\texttt{L}} := \{x \in T \mid x_i < t - 1\}$ is of size $|T_{\texttt{L}}| = \alpha|T|$,*
- *$T_{\texttt{M}} := \{x \in T \mid t - 1 \leq x_i \leq t + 1\}$ is of size $|T_{\texttt{M}}| = \beta|T|$,*
- *$T_{\texttt{R}} := \{x \in T \mid t + 1 < x_i\}$ is of size $|T_{\texttt{R}}| = \gamma|T|$.*

▶ **Lemma 2.6** (Restatement of [29, Lemma 1])**.** *Given a set $T \subseteq \ell_\infty^d$ and parameter $0 < \rho \leq 1$, there is an $O(d|T|)$-time algorithm whose output is one of the following options:*

1. *(Separator) An (α, β, γ)-separator of T such that $\alpha, \gamma \geq \frac{1}{4d}$ and $\alpha^{1-\rho} + \gamma^{1-\rho} \geq 1$.*
2. *(Dense Box) A point $x^* \in T$ such that, for $r = O(\frac{\log\log d}{\rho})$, we have $|B_T(x^*, r)| \geq \frac{|T|}{2}$.*

▶ Remark 2.7. In [29], the inequality in the separator case is $\log_{\frac{1}{\beta+\gamma}}\left(\frac{\beta+\gamma}{\gamma}\right) \leq \rho$, which is equivalent to $\beta \leq \gamma^{1/(1+\rho)} - \gamma$. This implies our inequality:

$$1 = \alpha + \beta + \gamma \leq \alpha + \gamma^{1/(1+\rho)} \leq \alpha + \gamma^{1-\rho} \leq \alpha^{1-\rho} + \gamma^{1-\rho}$$

(where the middle inequality utilizes the fact that $1/(1+\rho) \geq 1 - \rho$).

We are now ready to give the algorithm of Lemma 2.4. The basic idea is to apply Lemma 2.6 on the given terminal set, and recurse according to its output. If we get a dense box, we just add it (or, more precisely, its intersection with the domain) to the rectangle cover and recurse on the remaining, uncovered terminals. If we get a separator, we "give up" on covering the few terminals in the middle (namely, include them in $T^{\texttt{new}}$), and recurse on the left ones and on the right ones.

Algorithm 1 Algorithm **Cover**(D, X, T, ρ)

Input: rectangular domain D, dataset $X \subseteq D$, terminal set $T \subseteq X$, parameter $0 < \rho \leq 1$.

Output: new terminal set $T^{\texttt{new}}$, rectangle collection $\mathcal{R}$.

- Base Case: If $T = \emptyset$, return $(\emptyset, \emptyset)$.
- Recursive Step: Apply Lemma 2.6 on T with parameter ρ.
  1. Case 1: If found (α, β, γ)-separator $H_i(t)$ such that $\alpha, \gamma \geq \frac{1}{4d}$ and $\alpha^{1-\rho} + \gamma^{1-\rho} \geq 1$.
     - Compute the sets $T_{\texttt{L}}, T_{\texttt{M}}, T_{\texttt{R}}$ corresponding to $H_i(t)$ (see Definition 2.5).
     - Define $D_{\texttt{L}} = \{x \in D \mid x_i < t\}$ and $D_{\texttt{R}} = \{x \in D \mid x_i > t\}$.
     - Compute $X_{\texttt{L}} = D_{\texttt{L}} \cap X$ and $X_{\texttt{R}} = D_{\texttt{R}} \cap X$.
     - Let $(T^{\texttt{new}}_{\texttt{L}}, \mathcal{R}_{\texttt{L}}) \leftarrow \text{Cover}(D_{\texttt{L}}, X_{\texttt{L}}, T_{\texttt{L}}, \rho)$ and $(T^{\texttt{new}}_{\texttt{R}}, \mathcal{R}_{\texttt{R}}) \leftarrow \text{Cover}(D_{\texttt{R}}, X_{\texttt{R}}, T_{\texttt{R}}, \rho)$.
     - Return $(T^{\texttt{new}} := T^{\texttt{new}}_{\texttt{L}} \cup T_{\texttt{M}} \cup T^{\texttt{new}}_{\texttt{R}}, \mathcal{R} := \mathcal{R}_{\texttt{L}} \cup \mathcal{R}_{\texttt{R}})$.
  2. Case 2: Else, found $x^* \in T$ such that $|B_T(x^*, r)| \geq \frac{|T|}{2}$ for $r = O(\frac{\log\log d}{\rho})$.
     - Let $(T^{\texttt{new}}, \mathcal{R}') \leftarrow \text{Cover}(D, X, T \setminus B_T(x^*, r), \rho)$.
     - Define $R_{\texttt{new}} = B_\infty(x^*, r+1) \cap D$, with associated cluster $C(R_{\texttt{new}}) = X \cap R_{\texttt{new}}$.
     - Return $(T^{\texttt{new}}, \mathcal{R} := \mathcal{R}' \cup \{R_{\texttt{new}}\})$.

Correctness. We prove the correctness of $\text{Cover}(D, X, T, \rho)$ by induction on $|T|$. The base case $|T| = 0$ is trivial, so we assume $|T| > 0$ from now on, and denote the output of $\text{Cover}(D, X, T, \rho)$ by $(T^{\texttt{new}}, \mathcal{C})$. Note that the induction hypothesis applies to all recursive invocations of Cover. Indeed, in Case 1 we invoke it with terminal sets of size $(1-\alpha)|T|$ and $(1-\gamma)|T|$, and in Case 2 with a terminal set of size $\leq \frac{1}{2}|T|$. Furthermore, in both cases, the terminals in the recursive call all have ℓ_∞ distance of more than 1 to the boundary of the domain in this call.

Diameter. The fact that each rectangle in $\mathcal{R}$ has width $O(\frac{\log\log d}{\rho})$ is immediate by induction: In Case 1 we do not add any new rectangles other than the ones recursively constructed. In Case 2, we add one more rectangle of width $\leq 2(r+1) = O(\frac{\log\log d}{\rho})$.

Covering. Let $x \in T \setminus T^{\texttt{new}}$, and we should show there is some $R \in \mathcal{R}$ such that $B_\infty(x, 1) \subseteq R$.

We start with Case 1, and denote by $H_i(t)$ the (α, β, γ) separator of this case. We have that $T \setminus T^{\texttt{new}} = T \setminus (T^{\texttt{new}}_{\texttt{L}} \cup T_{\texttt{M}} \cup T^{\texttt{new}}_{\texttt{R}})$, so either $x \in T_{\texttt{L}} \setminus T^{\texttt{new}}_{\texttt{L}}$ or $x \in T_{\texttt{R}} \setminus T^{\texttt{new}}_{\texttt{R}}$. If the former (resp., latter) holds, then by induction, there is $R \in \mathcal{R}_{\texttt{L}}$ (resp. $R \in \mathcal{R}_{\texttt{R}}$) s.t. $B_\infty(x, 1) \subseteq R$.

Next, consider Case 2. Suppose first that $x \in B_T(x^*, r)$. In this subcase, we argue that the newly added rectangle $R_{\texttt{new}} = B_\infty(x^*, r+1) \cap D$ contains $B_\infty(x, 1)$. Indeed, since $x \in T$, its ℓ_∞-distance from the boundary of D is more than 1, so this follows by triangle inequality.

Finally, suppose $x \in (T \setminus B_T(x^*, r)) \setminus T^{\texttt{new}}$. Recall that $T^{\texttt{new}}$ is the new terminal set returned by the recursive call $\text{Cover}(D, X, T \setminus B_T(x^*, r), \rho)$, so by induction, there exists $R \in \mathcal{R}'$ that contains $B_\infty(x, 1)$.

Sparsity. We now show that $\mathcal{R}$ can be partitioned into at most $\log|T| + 1$ sub-collections, each consisting of pairwise-disjoint rectangles.

First consider Case 1. Note that rectangles in $\mathcal{R}_{\texttt{L}}$ (resp., $\mathcal{R}_{\texttt{R}}$) are contained within $D_{\texttt{L}}$ (resp., $D_{\texttt{R}}$) because they were returned by invoking Cover with domain $D_{\texttt{L}}$ (resp., $D_{\texttt{R}}$). By induction hypothesis, both $\mathcal{R}_{\texttt{L}}$ and $\mathcal{R}_{\texttt{R}}$ can be partitioned into at most $\log|T| + 1$ sub-collections of pairwise-disjoint rectangles, and we can concatenate the i-th sub-collection of $\mathcal{R}_{\texttt{L}}$ with the i-the sub-collection of $\mathcal{R}_{\texttt{R}}$ to obtain the i-th sub-collection of $\mathcal{R}$.

For Case 2, we add one more rectangle $R_{\texttt{new}}$ to the recursively constructed $\mathcal{R}'$. By induction, the latter can be partitioned into at most $\log\left|T \setminus B_T(x^*, r)\right| + 1 \le \log\frac{|T|}{2} + 1 = \log|T|$ sub-collections of pairwise-disjoint rectangles. So, we simply put $R_{\texttt{new}}$ in a new sub-collection $\{R_{\texttt{new}}\}$.

Terminal reduction. To conclude the correctness proof, we show that $|T^{\texttt{new}}| \le |T| - |T|^{1-\rho}$.

In Case 1, we have

$$\begin{aligned}
|T^{\texttt{new}}| &= |T_{\texttt{L}}^{\texttt{new}}| + |T_{\texttt{R}}^{\texttt{new}}| \\
&\le (|T_{\texttt{L}}| - |T_{\texttt{L}}|^{1-\rho}) + (|T_{\texttt{R}}| - |T_{\texttt{R}}|^{1-\rho}) && \text{by induction hypothesis,} \\
&= (\alpha+\gamma)|T| - (\alpha^{1-\rho} + \gamma^{1-\rho})|T|^{1-\rho} && \text{as } |T_{\texttt{L}}| = \alpha|T|,\ |T_{\texttt{R}}| = \gamma|T|, \\
&\le |T| - |T|^{1-\rho} && \text{as } \alpha+\gamma \le 1, \text{ and } \alpha^{1-\rho} + \gamma^{1-\rho} \ge 1.
\end{aligned}$$

In Case 2, denote $B := B_T(x^*, r)$.

$$\begin{aligned}
|T^{\texttt{new}}| &\le (|T| - |B|) - (|T| - |B|)^{1-\rho} && \text{by induction hypothesis,} \\
&\le |T| - (|B|^{1-\rho} + (|T| - |B|)^{1-\rho}) && \text{using } |B| \ge |B|^{1-\rho}, \\
&\le |T| - (|B| + |T| - |B|)^{1-\rho} && \text{by sub-additivity of } f(z) = z^{1-\rho} \\
&= |T| - |T|^{1-\rho}.
\end{aligned}$$

Running time. Finally, we analyze the running time of $\text{Cover}(D, X, T, \rho)$. Consider the recursion tree of the algorithm, where each node is associated with an instance of dataset X' and terminal set T'. The work done inside this node consists of applying Lemma 2.6, which takes $O(d|T'|) \le O(d|X'|)$ time, and spending another $O(d|X'|)$ time to partition X' into $X'_{\texttt{L}}, X'_{\texttt{R}}$ or to compute $B_{T'}(x^*, r)$ and the rectangle $R_{\texttt{new}}$ with its associated cluster $C(R_{\texttt{new}})$. Note that the datasets of instances from a given level i of the recursion tree are disjoint subsets of X, so the total time spent in each level i is $O(|X| \cdot d)$.

It thus remains to bound the depth of the recursion tree by $O(d \log|T|)$. For this, we assert that the instance of a node in depth i has terminal set of size at most $(1 - \frac{1}{4d})^i|T|$, which means that the leaves (with terminal sets of size 0) must have depth $O(d \log|T|)$. The assertion follows easily by induction on i, using the fact that $\alpha, \gamma \ge \frac{1}{4d}$ in Case 1, and that $B_T(x^*, r) \ge \frac{|T|}{2} \ge \frac{1}{4d}|T|$ in Case 2.

This concludes the proof of Lemma 2.4. ◀

2.3 From Neighborhood Cover to Padded Decomposition

We now explain the details of the reduction from our rectangle-based sparse neighborhood cover to padded decomposition, which gives the proof of Theorem 1.3.

The reduction is a slight modification of the one by [16] to ensure fast running time. The main point is the following: Given a (β, s, Δ)-neighborhood cover $\mathcal{C}$, the reduction of [16] uses the "boundary distances" $\partial_C(x) = \min_{y \in X \setminus C}\{d_X(x,y)\}$ defined for every point $x \in X$ and cluster $C \in \mathcal{C}$ to construct an $(O(\beta \log s), \frac{1}{4\beta}, \Delta)$-padded decomposition. However, this "discrete" definition of ∂_C is problematic for us to compute in less than $O(n^2)$ time. To overcome this, we utilize the fact that our clustering is in fact induced by intersections with "continuous" axis-aligned rectangles in $\mathbb{R}^d$, which allows to redefine the values $\partial_C(x)$ while preserving all their important properties.

From now on, assume Theorem 2.3 was applied with diameter bound Δ and parameter ρ to be chosen later, let $\mathcal{R}$ be the resulting rectangle collection, and $\mathcal{C}(\mathcal{R})$ the resulting cluster collection which forms a (β, s, Δ)-neighborhood cover with $\beta = O(\rho^{-1} \log\log d)$ and $s = O(n^\rho \rho^{-1} \log n)$. Let $x \in X$ and $R \in \mathcal{R}$ with corresponding cluster $C(R)$. We define $\partial_{C(R)}(x)$ as the ℓ_∞-distance between x and $\mathbb{R}^d \setminus R$. Formally, let $R = I_1 \times \cdots \times I_d$, where the interval $I_i \subseteq \mathbb{R}$ has boundary points $a_i \le b_i$. Then,

$$\partial_{C(R)}(x) = \inf\{\|x-y\|_\infty : y \in \mathbb{R}^d \setminus R\} = \max\left\{0, \min_{i=1,\dots,d}\left\{\min\{x_i - a_i, b_i - x_i\}\right\}\right\}. \quad (1)$$

Note that we can compute all of the nonzero values in $\{\partial_{C(R)}(x) : x \in X, C(R) \in \mathcal{C}(\mathcal{R})\}$ in just $O(n \cdot s \cdot d) = O(n^{1+\rho} \cdot \rho^{-1} d)$ time, since $\partial_{C(R)}(x)$ can be nonzero only if x belongs to the rectangle R, and there are at most s such rectangles in $\mathcal{R}$. This is the crucial point where we use the fact our neighborhood cover $\mathcal{C}(\mathcal{R})$ is induced by intersections with axis-aligned rectangles to allow fast computation.

We now use these values to define the padded decomposition exactly as in [16]. Let $\mathrm{Texp}(\lambda)$ be the truncated exponential distribution with parameter λ, with density function $g(z) = \frac{\lambda e^{-\lambda z}}{1-e^{-\lambda}}$ for $z \in [0,1]$. For every cluster $C \in \mathcal{C}(\mathcal{R})$, we sample (independently) $\delta_C \sim \mathrm{Texp}(2+2s)$, and define the function $f_C : X \to \mathbb{R}$ by

$$f_C(x) = \delta_C \cdot \frac{\Delta}{\beta} + \partial_C(x).$$

Define the partition $\mathcal{P} = \{P_C\}_{C \in \mathcal{C}(\mathcal{R})}$ where each point $x \in X$ joins the cluster P_C such that $f_C(x)$ is maximized.

First, we show that $\mathcal{P}$ can be computed in just $O(n \cdot s \cdot d)$ time. Fix $x \in X$, and let $\mathcal{R}_x$ be the rectangles in $\mathcal{R}$ that contain x, which are at most s. We claim that x can only join P_C if $C = C(R)$ for some $R \in \mathcal{R}_x$, so we only need to compute s values of $f_C(x)$ to determine which P_C contains x. This is seen by the observing the following:

- By the covering property, some $R \in \mathcal{R}_x$ contains $B_\infty(x, \frac{\Delta}{\beta})$, so $f_{C(R)}(x) \ge \partial_{C(R)}(x) \ge \frac{\Delta}{\beta}$.
- On the other hand, if $x \notin R$, then $\partial_{C(R)}(x) = 0$, and hence $f_{C(R)}(x) < \frac{\Delta}{\beta}$ w.p. 1.

Note that the above argument showed that $P_{C(R)} \subseteq C(R) \subseteq R$. Hence, the diameter of $P_{C(R)}$ is at most the width of R, which is at most Δ, so $\mathcal{P}$ is indeed Δ-bounded.

The proof of the padding property from [16] can be applied verbatim. This is because the only property of the values $\partial_C(x)$ used by [16] for this proof is the triangle inequality $|\partial_C(x) - \partial_C(y)| \le d_X(x,y) = \|x-y\|_\infty$, which also holds for our definition in Equation (1).

We thus obtained that $\mathcal{P}$ is indeed a partition sampled from a $(O(\beta \log s), \frac{1}{4\beta}, \Delta)$-padded decomposition. The running time is dominated by computing the rectangle neighborhood cover in Theorem 2.3. Choosing $\rho = \frac{\log\log n}{\log n}$ completes the proof of Theorem 1.3.

2.4 Application: Spanners in ℓ_∞^d

A *spanner with stretch* $t \geq 1$ for a finite metric (X, d_X) is a weighted graph $G = (X, E, w)$ with edge weights $w(u, v) = d_X(u, v)$ whose shortest-path distances d_G satisfy $d_X(x, y) \leq d_G(x, y) \leq t \cdot d_X(x, y)$ for all $x, y \in X$. The main quality measure for spanners is the tradeoff between the stretch and the *sparsity* (i.e., number of edges $|E|$). Neighborhood covers are well-known to yield spanner algorithms [6, 7, 26]. In our case, by applying the algorithm of Theorem 2.3 with $O(\log D_X)$ different diameter bounds, where $D_X = \frac{\max_{x,y \in X} \|x-y\|_\infty}{\min_{x \neq y \in X} \|x-y\|_\infty}$ is the *aspect ratio* of the metric (X, d_X), we obtain the following.

▶ **Theorem 2.8.** *There is a deterministic algorithm that, given dataset $X \subseteq \ell_\infty^d$ of size $|X| = n$ with aspect ratio D_X, and parameter $0 < \rho < 1$, outputs a spanner for (X, d_X) with stretch $O(\rho^{-1} \log \log d)$ and sparsity $O(n^{1+\rho} \rho^{-1} \log n \log D_X)$. The algorithm runs in $O(n^{1+\rho} \rho^{-1} d^2 \log n \log D_X)$ time.*

The proof is a standard argument, included in Appendix A for completeness. We remark that, while we mainly focused on neighborhood covers as a tool for padded decompositions, there is a relaxed notion of such covers with *average sparsity* guarantee which is enough to yield spanners. In the average sparsity variant, the property that each $x \in X$ is contained in at most s clusters is replaced by $\sum_{C \in \mathcal{C}} |C| \leq n \cdot s$. For this relaxed notion, our Theorem 2.3 can be slightly improved to achieve $s = O(n^\rho \log n)$, removing a ρ^{-1} factor, and this improvement propagates to the sparsity and running time of Theorem 2.8.

We omit the full details, and only sketch the argument as follows: Adapt the Cover algorithm so that instead of "giving up" on the middle $T_{\mathtt{M}}$, it includes it in both recursive calls on the left and on the right. In this case, just a single invocation of Cover produces all of the clusters $\mathcal{C}$. The analysis of the running time and the sparsity is by bounding the depth of the recursion tree and the total size of instances in each of its levels, similarly to [29].

3 Separating Decomposition in ℓ_2^d

This section proves Theorem 1.4 by providing a separating-decomposition algorithm with quality $\tilde{O}(\sqrt{\log n})$ that runs in almost-linear time. A central component in the proof is a powerful decomposition technique for the entire ℓ_2^d space from [2, Section 3], which runs in exponential time in the dimension of the space, as follows.

▶ **Theorem 3.1** ([2, Section 3])**.** *There is a separating decomposition algorithm for n-point metrics in ℓ_2^d with quality $O(\sqrt{d})$ and running time $\tilde{O}(n \cdot 2^{d \log d})$. The algorithm is Las Vegas, meaning that with high probability it reports a partition sampled from a separating decomposition, and with the remaining probability it reports "Fail".*

We are now ready to prove Theorem 1.4.

Proof of Theorem 1.4. Assume first that $d \leq O(\log n)$. Essentially, this assumption is valid due to the JL-transform [31], and we formally justify it at the end of the proof. Without loss of generality, we may assume that $\Delta = 1$ (by rescaling), that $\rho = \Omega(\frac{1}{\log n})$ (since a smaller ρ does not improve the running time), and that ρ^{-1} is an integer.

Arbitrarily partition the coordinates as $I_1 \cup \cdots \cup I_{\rho^{-1}} = [d]$, where each I_j is of size at most $O(\rho \cdot d)$. For every I_j and $x \in X$, let $x[I_j]$ denote the restriction of x to the coordinates in I_j, and let $X[I_j] = \{x[I_j] \mid x \in X\} \subset \ell_2^{O(\rho \cdot d)}$. For each I_j, apply Theorem 3.1 to sample a partition $\mathcal{P}_j$ from a $(O(\sqrt{\rho \cdot d}), \sqrt{\rho})$-separating decomposition of $X[I_j]$, which takes $\tilde{O}(n \cdot 2^{\rho \cdot d \log(\rho \cdot d)}) = \tilde{O}(n^{1+\rho \cdot \log \log n})$ time. If one of the ρ^{-1} applications of Theorem 3.1

fails, we halt the algorithm and report "Fail". Note that by a union bound, this step succeeds with high probability. Finally, the partition $\mathcal{P}$ of X is defined as the common refinement of all $\mathcal{P}_1, \ldots, \mathcal{P}_{\rho^{-1}}$. That is, two points $x, y \in X$ belong to the same cluster of $\mathcal{P}$ iff $\mathcal{P}_j(x[I_j]) = \mathcal{P}_j(y[I_j])$ for all $1 \le j \le \rho^{-1}$.

We move to analyze the correctness of the described algorithm.

Diameter. To show that $\mathcal{P}$ is 1-bounded, let $x, y \in X$ be points that were clustered together by $\mathcal{P}$; namely, for every $1 \le j \le \rho^{-1}$, $\mathcal{P}_j(x[I_j]) = \mathcal{P}_j(y[I_j])$. Then, since every $\mathcal{P}_j$ is ρ-bounded we have

$$\|x-y\|_2^2 = \sum_{j=1}^{d} \|x_j - y_j\|_2^2 = \sum_{j=1}^{t} \|x[I_j] - y[I_j]\|_2^2 \le \rho^{-1} \cdot (\sqrt{\rho})^2 = 1.$$

Separation. Let $x, y \in X$. Then, we have

$$\begin{aligned}
\Pr[\mathcal{P}(x) \ne \mathcal{P}(y)] &\le \sum_{j=1}^{\rho^{-1}} \Pr[\mathcal{P}_j(x[I_j]) \ne \mathcal{P}_j(y[I_j])] && \text{by a union bound,} \\
&\le \sum_{j=1}^{\rho^{-1}} O(\sqrt{\rho \cdot d}) \frac{\|x[I_j] - y[I_j]\|_2}{\sqrt{\rho}} && \text{by def. of } \mathcal{P}_j, \\
&= O(\sqrt{d}) \sum_{j=1}^{\rho^{-1}} \|x[I_j] - y[I_j]\|_2 \\
&\le O(\sqrt{d}) \left(\sqrt{\rho^{-1}} \sqrt{\sum_{j=1}^{\rho^{-1}} \|x[I_j] - y[I_j]\|_2^2} \right) && \text{by Cauchy-Schwarz,} \\
&= O(\sqrt{\rho^{-1} \log n}) \cdot \|x-y\|_2,
\end{aligned}$$

as required.

To conclude, for the case $d = O(\log n)$, we have shown a separating-decomposition algorithm with quality $O(\sqrt{\rho^{-1} \log n})$ that runs in time $\tilde{O}(\rho^{-1} \cdot n^{1+\rho \log\log n})$. So, Theorem 1.4 for this case follows by rescaling ρ.

General dimension. To resolve the case of general d, apply a JL transform [31] to embed the dataset X into $\ell_2^{O(\log n)}$, such that with probability at least $1 - \frac{1}{n^6}$, the distance between every two dataset points contracts or expands by at most factor 2. Then apply the preceding algorithm with diameter bound $\tilde{\Delta} = \Delta/4$, obtaining within $\tilde{O}(n^{1+\rho})$ time a partition sampled from a separating decomposition with separation parameter $\tilde{\alpha} = O(\sqrt{\rho^{-1} \log n \log\log n})$, which induces a partition $\mathcal{P}$ of the original dataset X.

Outputting the partition $\mathcal{P}$ would cause some issues, even though the failure probability of the JL transform is polynomially small. First, when it fails, clusters might not have diameter bounded by Δ, whereas the output must be a Δ-bounded partition (with probability 1). Second, for close-by points, say within distance $\ll \frac{\Delta}{n^6}$, the separation probability is dominated by that failure probability, which is not proportional to the distance. To solve these issues, execute the following verification steps before outputting $\mathcal{P}$ (and if one of them fails, we output "Fail"):

1. Diameter verification: For every cluster $C \in \mathcal{P}$, pick an arbitrary point $x \in C$, and verify that $\|x - y\|_2 \le \Delta/2$ for all $y \in C$. This can clearly be implemented in deterministic linear time $O(nd)$.
2. Close-pairs verification: This step outputs "Fail" or "Success", with the following guarantees, assuming that $d \le n$ without loss of generality:[3]
   - If some $x, y \in X$ with $\|x - y\|_2 \le \frac{\Delta}{n^6}$ are separated by $\mathcal{P}$, the output is "Fail".
   - If all $x, y \in X$ with $\|x - y\|_2 \le O(d^{3/2}) \cdot \frac{\Delta}{n^6} \le O(\frac{\Delta}{n^4})$ satisfy $\mathcal{P}(x) = \mathcal{P}(y)$, the output is "Success". (In the contrapositive: if the output is "Fail" then some $x, y \in X$ with $\|x - y\|_2 \le O(\frac{\Delta}{n^4})$ are separated by $\mathcal{P}$.)

   This step essentially scans all the pairs whose distance is at most $\frac{\Delta}{n^6}$, but relaxing this threshold is by factor $O(d^{3/2})$. It can be implemented in deterministic $\tilde{O}(d^2 n)$ time by applying hashing techniques of [13] for approximate near-neighbor search; see [51, Section 4] for a complete description.

We claim that both verification steps succeed with high probability. Indeed, the diameter verification succeeds whenever the JL transform succeeds, in which case every cluster of $\mathcal{P}$ has diameter at most $2\tilde{\Delta} = \Delta/2$. As for the close-pairs verification, when the JL transform succeeds, each pair $x, y \in X$ with $\|x - y\|_2 \le O(\frac{\Delta}{n^4})$ is separated with probability at most $\tilde{\alpha} \cdot \frac{2\|x-y\|_2}{\tilde{\Delta}} \le O(1/n^3)$, so by a union bound, with high probability no such pair is separated and the verification succeeds.

Finally, we show that when the algorithm reports $\mathcal{P}$, it is indeed a partition sampled from an $(O(\sqrt{\rho^{-1} \log n \log\log n}), \Delta)$-separating decomposition. Observe that $\mathcal{P}$ is Δ-bounded because by the first verification step every cluster has radius at most $\Delta/2$. To verify the separation probability, denote by V the event that the verification succeeds, and by J the event that the JL transform succeeds. Now consider $x, y \in X$. If $\|x - y\|_2 > \frac{\Delta}{n^6}$, then

$$\begin{aligned}
\Pr[\mathcal{P}(x) \ne \mathcal{P}(y) \mid V] &= \frac{\Pr[\mathcal{P}(x) \ne \mathcal{P}(y) \wedge V]}{\Pr[V]} \\
&\le \frac{\Pr[\mathcal{P}(x) \ne \mathcal{P}(y)]}{\Pr[V]} \\
&\le O(1) \cdot \Pr[\mathcal{P}(x) \ne \mathcal{P}(y)] && \text{as } V \text{ happens w.h.p.} \\
&\le O(1) \cdot \big(\Pr[\mathcal{P}(x) \ne \mathcal{P}(y) \mid J] + \Pr[\overline{J}]\big) && \text{by law of total probability,} \\
&\le O(1) \cdot \big(\tilde{\alpha} \cdot \frac{2\|x-y\|_2}{\tilde{\Delta}} + \frac{1}{n^6}\big) && \text{as } J \text{ happens w.h.p.} \\
&\le O(\tilde{\alpha}) \frac{\|x-y\|_2}{\Delta} && \text{as } \|x-y\|_2 > \frac{\Delta}{n^6};
\end{aligned}$$

and if $\|x - y\|_2 \le \frac{\Delta}{n^6}$ then clearly $\Pr[\mathcal{P}(x) \ne \mathcal{P}(y) \mid V] = 0$. This concludes the proof. ◀

3.1 Application: Deterministic Nearest-Neighbor Search

The c-approximate nearest-neighbor search (c-ANN) problem asks to preprocess an n-point dataset P in a metric space (X, d_X) into a data structure that can quickly answer the following queries: given $q \in X$, report $x \in P$ satisfying $d_X(q, x) \le c \cdot \min_{y \in P} d_X(q, y)$. In this section, we apply the simple "coordinate division" trick used in our proof of Theorem 1.4, together with several ingenious ANN algorithms of Indyk, to prove the following.

▶ **Theorem 3.2.** *For every $0 < \epsilon < 1$, there is a* deterministic $O(\epsilon^{-2} \log\log n)$*-ANN for n-point datasets in ℓ_2^d or in ℓ_1^d, with space $\tilde{O}(\epsilon^{-1} n^{1+\epsilon} \cdot \mathrm{poly}(d))$, query time* $\mathrm{poly}(d \log n)$*, and preprocessing time* $\mathrm{poly}(nd)$*.*

[3] If $d > n$, apply Gram-Schmidt to reduce to a subspace of dimension $\le n$ in time $O(n^2 d) \le O(nd^2)$.

To the best of our knowledge, no known deterministic ANN for ℓ_2^d or ℓ_1^d when $d \gg \log n$ achieves $o(\log n)$ approximation, $\mathrm{poly}(d \log n)$ query time, and $\tilde{O}(\epsilon^{-1} n^{1+\epsilon} \cdot \mathrm{poly}(d))$ space. Furthermore, a recent reduction from [37] shows that Theorem 3.2 can be extended to ℓ_p^d for every $1 \le p < \infty$, at the cost of an additional $\epsilon^{-1} p^{O(1)+\log\log p}$ factor in the approximation.

We now discuss the proof of Theorem 3.2. By employing a reduction from [28] (see also [3, Section 3]), for every $c > 1$, one can use $\mathrm{poly}(d \log n)$ deterministic c-ANNs for *Hamming space* $(\{0,1\}^{O(\log n)}, d_{\mathcal{H}})$ to compute (in polynomial time) a deterministic $O(c)$-ANN in ℓ_2^d or ℓ_1^d, with at most a $\mathrm{poly}(d \log n)$ overhead in query time and space. Thus Theorem 3.2 reduces to proving the following lemma.

▶ **Lemma 3.3.** *For every $0 < \epsilon < 1$, there is a* deterministic *$O(\epsilon^{-2} \log\log n)$-ANN for n-point datasets in Hamming space $(\{0,1\}^{O(\log n)}, d_{\mathcal{H}})$ with query time $O(\log n)$, space $\tilde{O}(\epsilon^{-1} n^{1+\epsilon})$ and preprocessing time $\mathrm{poly}(n)$.*

Proof. Let $P \subseteq \{0,1\}^{O(\log n)}$ be the given dataset. Divide the $O(\log n)$ coordinates of the space into $r = O(\epsilon^{-1})$ sets $I_1, \ldots, I_r$, each of size at most $\epsilon \log n$. Let $d_{\mathtt{prod}}$ denote the *product metric* over $\{0,1\}^{O(\log n)} = \{0,1\}^{I_1} \times \cdots \times \{0,1\}^{I_r}$, defined by

$$d_{\mathtt{prod}}(x,y) := \max_{j=1,\ldots,r} d_{\mathcal{H}}(x[I_j], y[I_j]).$$

That is, $d_{\mathtt{prod}}(x,y)$ is obtained by considering the Hamming distance restricted to each coordinate block I_j, and taking the maximum. Clearly, $\frac{1}{r} \cdot d_{\mathcal{H}}(x,y) \le d_{\mathtt{prod}}(x,y) \le d_{\mathcal{H}}(x,y)$, i.e., the identity mapping from $(\{0,1\}^{O(\log n)}, d_{\mathcal{H}})$ to $(\{0,1\}^{O(\log n)}, d_{\mathtt{prod}})$ has distortion at most $r = O(\epsilon^{-1})$.

For each space $(\{0,1\}^{I_j}, d_{\mathcal{H}})$, construct a trivial 1-ANN data structure of space $O(2^{|I_j|}) = O(n^{\epsilon})$ and query time $O(1)$ that explicitly stores the answers to all possible queries. Furthermore, this 1-ANN also works for the priority version of the problem, where each point in the dataset has a priority, and if there is more than one nearest neighbor, the one with smallest priority should be reported.

We now appeal to a known ANN construction for product metrics, from [30, Theorem 1], and its slight space refinement in [4, Appendix A] (which require the aforementioned priority version). These imply an $O(\epsilon^{-1} \log\log n)$-ANN for the dataset P, considered in $(\{0,1\}^{O(\log n)}, d_{\mathtt{prod}})$, with space $\tilde{O}(\epsilon^{-1} n^{1+\epsilon})$ and query time $\mathrm{polylog}\, n$. We use this ANN also for the original Hamming metric: Given a query q, the returned point $x \in P$ satisfies

$$d_{\mathcal{H}}(q,x) \le r \cdot d_{\mathtt{prod}}(q,x) \le r \cdot O(\epsilon^{-1} \log\log n) \cdot \min_{y \in P} d_{\mathtt{prod}}(q,y) \le O(\epsilon^{-2} \log\log n) \cdot \min_{y \in P} d_{\mathcal{H}}(q,y),$$

as needed. ◀

References

1. I. Abraham, C. Gavoille, A. V. Goldberg, and D. Malkhi. Routing in networks with low doubling dimension. In *26th IEEE International Conference on Distributed Computing Systems*, page 75. IEEE, 2006. doi:10.1109/ICDCS.2006.72.
2. Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *47th Annual IEEE Symposium on Foundations of Computer Science*, pages 459–468. IEEE, 2006. doi:10.1109/FOCS.2006.49.
3. Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. In *Proceedings of the International Congress of Mathematicians (ICM 2018)*, pages 3287–3318, 2019. doi:10.1142/9789813272880_0182.

4 Alexandr Andoni, Huy L. Nguyen, Aleksandar Nikolov, Ilya P. Razenshteyn, and Erik Waingarten. Approximate near neighbors for general symmetric norms. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 902–913. ACM, 2017. doi:10.1145/3055399.3055418.

5 Alexandr Andoni and Hengjie Zhang. Sub-quadratic $(1+\epsilon)$-approximate Euclidean spanners, with applications. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023*, pages 98–112. IEEE, 2023. doi:10.1109/FOCS57990.2023.00014.

6 B. Awerbuch and D. Peleg. Sparse partitions. In *31st Annual IEEE Symposium on Foundations of Computer Science*, pages 503–513, 1990. doi:10.1109/FSCS.1990.89571.

7 Baruch Awerbuch, Bonnie Berger, Lenore Cowen, and David Peleg. Near-linear time construction of sparse neighborhood covers. *SIAM Journal on Computing*, 28(1):263–277, 1998. doi:10.1137/S0097539794271898.

8 Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science*, pages 184–193. IEEE, 1996.

9 Aaron Bernstein, Danupon Nanongkai, and Christian Wulff-Nilsen. Negative-weight single-source shortest paths in near-linear time. *J. ACM*, 72(4):23:1–23:34, 2025. doi:10.1145/3742890.

10 Punyashloka Biswal, James R. Lee, and Satish Rao. Eigenvalue bounds, spectral partitioning, and metrical deformations via flows. *J. ACM*, 57(3):13:1–13:23, 2010. doi:10.1145/1706591.1706593.

11 Guy E. Blelloch, Anupam Gupta, Ioannis Koutis, Gary L. Miller, Richard Peng, and Kanat Tangwongsan. Nearly-linear work parallel SDD solvers, low-diameter decomposition, and low-stretch subgraphs. *Theory Comput. Syst.*, 55(3):521–554, 2014. doi:10.1007/S00224-013-9444-5.

12 B. Brinkman and M. Charikar. On the impossibility of dimension reduction in l1. *J. ACM*, 52(5):766–788, 2005. doi:10.1145/1089023.1089026.

13 Timothy M. Chan. Approximate nearest neighbor queries revisited. *Discret. Comput. Geom.*, 20(3):359–373, 1998. doi:10.1007/PL00009390.

14 M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin. Approximating a finite metric by a small number of tree metrics. In *39th Annual Symposium on Foundations of Computer Science*, pages 379–388, 1998. doi:10.1109/SFCS.1998.743488.

15 Moses Charikar, Tom Leighton, Shi Li, and Ankur Moitra. Vertex sparsifiers and abstract rounding algorithms. In *51st Annual Symposium on Foundations of Computer Science*, pages 265–274. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.32.

16 Jonathan Conroy and Arnold Filtser. How to protect yourself from threatening skeletons: Optimal padded decompositions for minor-free graphs. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC*, pages 2281–2292. ACM, 2025. doi:10.1145/3717823.3718252.

17 Michael Elkin, Ofer Neiman, and Christian Wulff-Nilsen. Space-efficient path-reporting approximate distance oracles. *Theor. Comput. Sci.*, 651:1–10, 2016. doi:10.1016/J.TCS.2016.07.038.

18 M. Englert, A. Gupta, R. Krauthgamer, H. Räcke, I. Talgam-Cohen, and K. Talwar. Vertex sparsifiers: New results from old techniques. *SIAM Journal on Computing*, 43(4):1239–1262, 2014. doi:10.1137/130908440.

19 Uriel Feige, MohammadTaghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput.*, 38:629–657, May 2008. doi:10.1137/05064299X.

20 Arnold Filtser. On strong diameter padded decompositions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019*, volume 145 of *LIPIcs*, pages 6:1–6:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.6.

21 Arnold Filtser. Steiner point removal with distortion $O(\log k)$ using the relaxed-voronoi algorithm. *SIAM J. Comput.*, 48(2):249–278, 2019. doi:10.1137/18M1184400.

22 Arnold Filtser and Hung Le. Clan embeddings into trees, and low treewidth graphs. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2021*, pages 342–355. ACM, 2021. doi:10.1145/3406325.3451043.

23 Arnold Filtser and Ofer Neiman. Light spanners for high dimensional norms via stochastic decompositions. *Algorithmica*, 84(10):2987–3007, 2022. doi:10.1007/s00453-022-00994-0.

24 N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996. doi:10.1137/S0097539793243016.

25 A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science*, pages 534–543, 2003. doi:10.1109/SFCS.2003.1238226.

26 Sariel Har-Peled, Piotr Indyk, and Anastasios Sidiropoulos. Euclidean spanners in high dimensions. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013*, pages 804–809. SIAM, 2013. doi:10.1137/1.9781611973105.57.

27 Sariel Har-Peled, Manor Mendel, and Dániel Oláh. Reliable spanners for metric spaces. In *37th International Symposium on Computational Geometry (SoCG 2021)*, LIPIcs, pages 43:1–43:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.SOCG.2021.43.

28 Piotr Indyk. Dimensionality reduction techniques for proximity problems. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 371–378. ACM/SIAM, 2000. URL: http://dl.acm.org/citation.cfm?id=338219.338582.

29 Piotr Indyk. On approximate nearest neighbors under ℓ_∞ norm. *J. Comput. Syst. Sci.*, 63(4):627–638, 2001. doi:10.1006/JCSS.2001.1781.

30 Piotr Indyk. Approximate nearest neighbor algorithms for frechet distance via product metrics. In *Proceedings of the 18th Annual Symposium on Computational Geometry (SoCG 2002)*, pages 102–106. ACM, 2002. doi:10.1145/513400.513414.

31 William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into Hilbert space. *Contemporary mathematics*, 26:189–206, 1984. doi:10.1090/conm/026/737400.

32 Lior Kamma, Robert Krauthgamer, and Huy L. Nguyen. Cutting corners cheaply, or how to remove steiner points. *SIAM Journal on Computing*, 44(4):975–995, 2015. doi:10.1137/140951382.

33 Jonathan A. Kelner, James R. Lee, Gregory N. Price, and Shang-Hua Teng. Higher eigenvalues of graphs. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009*, pages 735–744. IEEE Computer Society, 2009. doi:10.1109/FOCS.2009.69.

34 R. Krauthgamer and J. R. Lee. The intrinsic dimensionality of graphs. *Combinatorica*, 27(5):551–585, 2007. doi:10.1007/s00493-007-2183-y.

35 R. Krauthgamer, J. R. Lee, M. Mendel, and A. Naor. Measured descent: A new embedding method for finite metrics. *Geometric And Functional Analysis*, 15(4):839–858, 2005. doi:10.1007/s00039-005-0527-6.

36 Robert Krauthgamer and Nir Petruschka. Lipschitz Decompositions of Finite ℓ_p Metrics. In *41st International Symposium on Computational Geometry (SoCG 2025)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 66:1–66:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.SoCG.2025.66.

37 Robert Krauthgamer and Nir Petruschka. Fast nearest neighbor search for ℓ_p metrics. *arXiv preprint*, 2026. Accepted to SoCG 2026. arXiv:2603.21148.

38 Robert Krauthgamer, Nir Petruschka, and Shay Sapir. The power of recursive embeddings for ℓ_p metrics. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 2546–2554. IEEE, 2025. doi:10.1109/FOCS63196.2025.00132.

39 J. R. Lee and A. Naor. Embedding the diamond graph in L_p and dimension reduction in L_1. *Geom. Funct. Anal.*, 14(4):745–747, 2004.

40 J. R. Lee and A. Naor. Metric decomposition, smooth measures, and clustering. Unpublished manuscript, 2004. URL: `https://web.math.princeton.edu/~naor/homepagefiles/cluster.pdf`.

41 James R. Lee and Assaf Naor. Extending Lipschitz functions via random metric partitions. *Inventiones Mathematicae*, 160(1):59–95, 2005. `doi:10.1007/s00222-004-0400-5`.

42 Jason Li, Connor Mowry, and Satish Rao. Faster negative-weight shortest paths and directed low-diameter decompositions. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2026*, pages 2824–2845. SIAM, 2026. `doi:10.1137/1.9781611978971.104`.

43 N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995. `doi:10.1007/BF01200757`.

44 Konstantin Makarychev and Yury Makarychev. Metric extension operators, vertex sparsifiers and Lipschitz extendability. *Israel Journal of Mathematics*, 212(2):913–959, 2016. `doi:10.1007/s11856-016-1315-8`.

45 J. Matoušek. On the distortion required for embedding finite metric spaces into normed spaces. *Israel J. Math.*, 93:333–344, 1996.

46 M. Mendel and A. Naor. Ramsey partitions and proximity data structures. *J. Eur. Math. Soc.*, 9(2):253–275, 2007.

47 Manor Mendel and Chaya Schwob. Fast C-K-R partitions of sparse graphs. *Chic. J. Theor. Comput. Sci.*, 2009, 2009. URL: `http://cjtcs.cs.uchicago.edu/articles/2009/2/contents.html`.

48 Gary L. Miller, Richard Peng, and Shen Chen Xu. Parallel graph decompositions using random shifts. In *25th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2013*, pages 196–203. ACM, 2013. `doi:10.1145/2486159.2486180`.

49 Assaf Naor and Kevin Ren. Optimal randomized clustering for subsets of L_p when $p > 2$. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, SODA 2026*, pages 984–995. SIAM, 2026. `doi:10.1137/1.9781611978971.40`.

50 S. Rao. Small distortion and volume preserving embeddings for planar and Euclidean metrics. In *Proceedings of the 15th Annual Symposium on Computational Geometry (SoCG 1999)*, pages 300–306. ACM, 1999. `doi:10.1145/304893.304983`.

51 Sharon Stein. Fast algorithms for separating decompositions in high-dimensional Euclidean spaces. Master's thesis, Weizmann Institute of Science, 2025. URL: `https://www.wisdom.weizmann.ac.il/~robi/files/Sharon.Stein-MScThesis-2025_03.pdf`.

A Spanner Construction in ℓ_∞

Proof of Theorem 2.8. Without loss of generality, assume $\min_{x,y\in X}\|x-y\|_\infty = 1$ and $\max_{x,y\in X}\|x-y\|_\infty = D_X$, and let $\beta = O(\rho^{-1}\log\log d)$. For every $i \in \{1,\dots,\log D_X + 1\}$, apply Theorem 2.3 with ρ and $\Delta_i = 2^i\beta$ to obtain a (β, s, Δ_i)-neighborhood cover $\mathcal{C}_i$, where $s = O(n^\rho\rho^{-1}\log n)$. We construct the spanner by selecting an arbitrary point s_C from every cluster $C \in \mathcal{C}_i$ and adding the edges $\big\{\{x, s_C\} \mid x \in C\big\}$.

The spanner's sparsity follows immediately from that of the covers $\mathcal{C}_1,\dots,\mathcal{C}_{O(\log D_X)}$: for each i, we add a linear number of edges per cluster $C \in \mathcal{C}_i$, and each point belongs to at most $s = O(n^\rho\rho^{-1}\log n)$ clusters. To verify the stretch, let $x, y \in X$ and let i such that $\|x-y\|_\infty \in [2^{i-1}, 2^i)$. Let $C \in \mathcal{C}_i$ be the cluster covering x, meaning $B(x, 2^i) \subseteq C$. Since $\|x-y\|_\infty < 2^i$, $y \in B(x,2^i) \subseteq C$, so the spanner contains the path $x \to s_C \to y$. The diameter of C is at most $2^i\beta$, hence, the weight of this path is at most $2^{i+1}\beta \le 4\beta\|x-y\|_\infty = O(\rho^{-1}\log\log d)\|x-y\|_\infty$, concluding the proof. ◀