Online Euclidean Spanners

Sujoy Bhore
Indian Institute of Science Education and Research, Bhopal, India.

Csaba D. Tóth
California State University Northridge, Los Angeles, CA; and Tufts University, Medford, MA, USA.

Abstract

In this paper, we study the online Euclidean spanners problem for points in $\mathbb{R}^d$. Given a set $S$ of $n$ points in $\mathbb{R}^d$, a $t$-spanner on $S$ is a subgraph of the underlying complete graph $G = (S, (\frac{1}{2}))$, that preserves the pairwise Euclidean distances between points in $S$ to within a factor of $t$, that is the stretch factor. Suppose we are given a sequence of $n$ points $(s_1, s_2, \ldots, s_n)$ in $\mathbb{R}^d$, where point $s_i$ is presented in step $i$ for $i = 1, \ldots, n$. The objective of an online algorithm is to maintain a geometric $t$-spanner on $S_i = \{s_1, \ldots, s_i\}$ for each step $i$. The algorithm is allowed to add new edges to the spanner when a new point is presented, but cannot remove any edge from the spanner. The performance of an online algorithm is measured by its competitive ratio, which is the supremum, over all sequences of points, of the ratio between the weight of the spanner constructed by the algorithm and the weight of an optimum spanner. Here the weight of a spanner is the sum of all edge weights.

First, we establish a lower bound of $\Omega(\varepsilon^{-1} \log n / \log \varepsilon^{-1})$ for the competitive ratio of any online $(1 + \varepsilon)$-spanner algorithm, for a sequence of $n$ points in 1-dimension. We show that this bound is tight, and there is an online algorithm that can maintain a $(1 + \varepsilon)$-spanner with competitive ratio $O(\varepsilon^{-1} \log n / \log \varepsilon^{-1})$. Next, we design online algorithms for sequences of points in $\mathbb{R}^d$, for any constant $d \geq 2$, under the $L_2$ norm. We show that previously known incremental algorithms achieve a competitive ratio $O(\varepsilon^{-(d+1)} \log n)$. However, if the algorithm is allowed to use additional points (Steiner points), then it is possible to substantially improve the competitive ratio in terms of $\varepsilon$. We describe an online Steiner $(1 + \varepsilon)$-spanner algorithm with competitive ratio $O(\varepsilon^{(1-d)/2} \log n)$. As a counterpart, we show that the dependence on $n$ cannot be eliminated in dimensions $d \geq 2$. In particular, we prove that any online spanner algorithm for a sequence of $n$ points in $\mathbb{R}^d$ under the $L_2$ norm has competitive ratio $\Omega(f(n))$, where $\lim_{n \to \infty} \frac{f(n)}{n} = \infty$. Finally, we provide improved lower bounds under the $L_1$ norm: $\Omega(\varepsilon^{-d} / \log \varepsilon^{-1})$ in the plane and $\Omega(\varepsilon^{-d})$ in $\mathbb{R}^d$ for $d \geq 3$.

2012 ACM Subject Classification
Mathematics of computing → Approximation algorithms; Mathematics of computing → Paths and connectivity problems; Theory of computation → Computational geometry

Keywords and phrases
Geometric spanner, $(1 + \varepsilon)$-spanner, minimum weight, online algorithm

1 Introduction

We study the online Euclidean spanners problem for a set of points in $\mathbb{R}^d$. Let $S$ be a set of $n$ points in $\mathbb{R}^d$. A $t$-spanner for a finite set $S$ of points in $\mathbb{R}^d$ is a subgraph of the underlying complete graph $G = (S, (\frac{1}{2}))$, that preserves the pairwise Euclidean distances between points in $S$ to within a factor of $t$, that is the stretch factor. The edge weights of $G$ are the Euclidean distances between the vertices. Chew [22, 23] initiated the study of Euclidean spanners in 1986, and showed that for a set of $n$ points in $\mathbb{R}^2$, there exists a spanner with $O(n)$ edges and constant stretch factor. Since then a large body of research has been devoted to Euclidean spanners due to its vast applications across domains, such as, topology control in wireless networks [50], efficient regression in metric spaces [31], approximate distance oracles [36], and many others. Moreover, Rao and Smith [48] showed the relevance of Euclidean spanners in the context of other fundamental geometric NP-hard problems, e.g., Euclidean traveling salesman problem and Euclidean minimum Steiner tree problem. Many
Online Euclidean Spanners

different spanner construction approaches have been developed for Euclidean spanners over the years, that each found further applications in geometric optimization, such as spanners based on well-separated pair decomposition (WSPD) [17, 35], skip-lists [4], path-greedy and gap-greedy approaches [3, 5], locality-sensitive orderings [21], and more. We refer to the book by Narasimhan and Smid [47] and the survey of Bose and Smid [16] for a summary of results and techniques on Euclidean spanners up to 2013.

Online Spanners. We are given a sequence of \( n \) points \( (s_1, s_2, \ldots, s_n) \), where the points are presented one-by-one, i.e., point \( s_i \) is revealed at the step \( i \), and \( S_i = \{ s_1, \ldots, s_i \} \) for \( i = 1, \ldots, n \). The objective of an online algorithm is to maintain a geometric \( t \)-spanner \( G_i \) for \( S_i \) for all \( i \). Importantly, the algorithm is allowed to add edges to the spanner when a new point arrives, however is not allowed to remove any edge from the spanner.

The performance of an online algorithm \( \text{ALG} \) is measured by comparing it to the offline optimum \( \text{OPT} \) using the standard notion of competitive ratio [14, Ch. 1]. The competitive ratio of an online \( t \)-spanner algorithm \( \text{ALG} \) is defined as \( \sup_{\sigma} \frac{\text{ALG}(\sigma)}{\text{OPT}(\sigma)} \), where the supremum is taken over all input sequences \( \sigma \), \( \text{OPT}(\sigma) \) is the minimum weight of a \( t \)-spanner for \( \sigma \), and \( \text{ALG}(\sigma) \) denotes the weight of the \( t \)-spanner produced by \( \text{ALG} \) for this input.

Computing a \((1 + \varepsilon)\)-spanner of minimum weight for a set \( S \) in Euclidean plane is known to be \( \text{NP} \)-hard [20]. However, there exists a plethora of constant-factor approximation algorithms for this problem in the offline model; see [3, 25, 26, 48]. Most of these algorithms approximate the parameter lightness (the ratio of the spanner weight to the weight of the Euclidean minimum spanning tree \( \text{MST}(S) \)) of Euclidean spanners, which in turn also approximates the optimum weight of the spanner. We refer to Section 1.1 for a more detailed overview of the parameter lightness.

Minimum spanning trees (MST) on \( n \) points in a metric space, which have no guarantee on the stretch factor, have been studied in the online model. It is not difficult to show that a greedy algorithm achieves a competitive ratio \( \Theta(\log n) \). The online Steiner tree problem was studied by Imase and Waxman [39], who proved \( \Theta(\log n) \)-competitiveness for the problem. Later, Alon and Azar [2] studied minimum Steiner trees for points in the Euclidean plane, and proved a lower bound \( \Omega(\log n / \log \log n) \) for the competitive ratio. Their result was the first to analyse the impact of Steiner points on a geometric network problem in the online setting. Several algorithms were proposed over the years for the online Steiner Tree and Steiner forest problems, on graphs in both weighted and unweighted settings; see [1, 6, 10, 37, 46].

Online Steiner Spanners. An important variant of online spanners is when it is allowed to use auxiliary points (Steiner points) which are not part of input sequence of points. It turns out that Steiner points allow for substantial improvements over the bounds on the sparsity and lightness of Euclidean spanners in the offline settings; see [12, 13, 42, 43]. In the geometric setting, an online algorithm is allowed to add Steiner points and subdivide existing edges with Steiner points at each time step. (This modeling decision has twofold justification: It accurately models physical networks such as roads, canals, or power lines, and from the theoretical perspective, it is hard to tell whether an online algorithm introduced a large number of Steiner points when it created an edge/path in the first place). However, the spanner must achieve the given stretch factor only for the input point pairs.

It is easy to see that in this model the online spanners in 1-dimension could attain optimum competitive ratio. However, it is unclear how it extends to higher dimensions as it has been observed in the offline settings that it tends to be more difficult to achieve tight bounds for Steiner spanners than their non-Steiner counterparts.

When the optimal Steiner spanner is lighter than \( \text{OPT}(S_i) \) without Steiner points, the adversary may decrease \( \text{OPT}(S_i) \) by adding suitable Steiner vertices to \( S_i \); see Fig. 1. In
Figure 1 (a) An optimum $\frac{3}{2}$-spanner on three points with all edges of unit length. (b) After inserting a fourth point at the center, the weight of the optimum $\frac{3}{2}$-spanner decreases.

In particular, $\text{OPT}(S_i)$ may or may not increase with $i$ in the model without Steiner points, but $\text{OPT}(S_i)$ monotonically increases in $i$ when Steiner points are allowed.

1.1 Related Work

Dynamic Spanners. In applications, the data (modeled as points in $\mathbb{R}^d$) changes over time, as new cities emerge, new wireless antennas are built, and users turn their wireless devices on or off. Dynamic models aim to maintain a geometric $t$-spanners for a dynamically changing point set $S$; in a restricted insert-only model, the input consists of a sequence of point insertions. In the dynamic model, the objective is design algorithms and data structures that minimize the worst-case update time needed to maintain a $t$-spanner for $S$ over all steps, regardless of its weight, sparsity, or lightness. Notice that dynamic algorithms are allowed to add or delete edges in each step, while online algorithms cannot delete edges. However, if a dynamic (or dynamic insert-only) algorithm always adds edges for a sequence of points insertions, it is also an online algorithm, and one can analyze its competitive ratio.

Arya et al. [4] designed a randomized incremental algorithm for $n$ points in $\mathbb{R}^d$, where the points are inserted in a random order, and maintains a $t$-spanner of $O(n)$ size and $O(\log n)$ diameter. Their algorithm can also handle random insertions and deletions in $O(\log^d n \log \log n)$ expected amortized update time. Later, Bose et al. [15] presented an insert-only algorithm to maintain a $t$-spanner of $O(n)$ size and $O(\log n)$ diameter in $\mathbb{R}^d$. Fischer and Har-Peled [29] used dynamic compressed quadtrees to maintain a WSPD-based $(1+\epsilon)$-spanner for $n$ points in $\mathbb{R}^d$ in expected $O((\log n + \log \epsilon^{-1}) \epsilon^{-d} \log n)$ update time. Their algorithm works under the online model, too, however, they have not analyzed the weight of the resulting spanner. Gao et al. [30] used hierarchical clustering for dynamic spanners in $\mathbb{R}^d$. Their DefSpanner algorithm is fully dynamic with $O(\log \Delta)$ update time, where $\Delta$ is the spread$^1$ of the set $S$. They maintain a $(1+\epsilon)$-spanner of weight $O(\epsilon^{-(d+1)} \|MST(S)\| \log \Delta)$, and for a sequence of point insertions, DefSpanner only adds edges. As $\text{OPT} \geq \|MST(S)\|$, DefSpanner can serve as an online algorithm with competitive ratio $O(\epsilon^{-(d+1)} \log \Delta)$.

Gottlieb and Roditty [32] studied dynamic spanners in more general settings. For every set of $n$ points in a metric space of bounded doubling dimension$^2$, they constructed a $(1+\epsilon)$-spanner whose maximum degree is $O(1)$ and that can be maintained under insertions and deletions in $O(\log n)$ amortized update time per operation. Later, Roditty [49] designed fully dynamic geometric $t$-spanners with optimal $O(\log n)$ update time for $n$ points in $\mathbb{R}^d$. Very

---

$^1$ The *spread* of a finite set $S$ in a metric space is the ratio of the maximum pairwise distance to the minimum pairwise distance of points in $S$; and $\log \Delta \geq \Omega(\log n)$ in doubling dimensions.

$^2$ A metric is said to be of a constant doubling dimension if a ball with radius $r$ can be covered by at most a constant number of balls of radius $r/2$. Very
Online Euclidean Spanners

recently, Chan et al. [21] introduced locality sensitive orderings in $\mathbb{R}^d$, which has applications in several proximity problems, including spanners. They obtained a fully dynamic data structure for maintaining a $(1 + \varepsilon)$-spanners in Euclidean space with logarithmic update time and linearly many edges. However, the spanner weight has not been analyzed for any of these constructions. Dynamic spanners have been subject to investigation in abstract graphs, as well. See [8, 9, 11] for some recent progress on dynamic graph spanners.

**Lightness and sparsity** are two natural parameters for Euclidean spanners. For a set $S$ of points in $\mathbb{R}^d$, the lightness is the ratio of the spanner weight (i.e., the sum of all edge weights) to the weight of the Euclidean minimum spanning tree $\text{MST}(S)$. It is known that greedy-spanner ([3]) has constant lightness; see [25, 26]. Later, Rao and Smith [48] in their seminal work, showed that the greedy spanner has lightness $\varepsilon^{-O(d)}$ in $\mathbb{R}^d$ for every constant $d$, and asked what is the best possible constant in the exponent. Then, the sparsity of a spanner on $S$ is the ratio of its size to the size of a spanning tree. Classical results [23, 24, 40, 53] show that when the dimension $d \in \mathbb{N}$ and $\varepsilon > 0$ are constant, every set $S$ of $n$ points in $d$-space admits an $(1 + \varepsilon)$-spanners with $O(n)$ edges and weight proportional to that of the Euclidean MST of $S$.

**Dependence on $\varepsilon > 0$ for constant dimension $d$.** The dependence of the lightness and sparsity on $\varepsilon > 0$ for constant $d \in \mathbb{N}$ has been studied only recently. Le and Solomon [42] constructed, for every $\varepsilon > 0$ and constant $d \in \mathbb{N}$, a set $S$ of $n$ points in $\mathbb{R}^d$ for which any $(1 + \varepsilon)$-spanner must have lightness $\Omega(\varepsilon^{-d})$ and sparsity $\Omega(\varepsilon^{-d+1})$, whenever $\varepsilon = \Omega(n^{-1/(d-1)})$. Moreover, they showed that the greedy $(1 + \varepsilon)$-spanner in $\mathbb{R}^d$ has lightness $O(\varepsilon^{-d} \log \varepsilon^{-1})$. In fact, Le and Solomon [42] noticed that Steiner points can substantially improve the bound on the lightness and sparsity of an $(1 + \varepsilon)$-spanner. For minimum sparsity, they gave an upper bound of $O(\varepsilon^{(1-d)/2})$ for $d$-space and a lower bound of $\Omega(\varepsilon^{-1/2} \log \varepsilon^{-1})$. For minimum lightness, they gave a lower bound of $\Omega(\varepsilon^{-1} / \log \varepsilon^{-1})$, for points in the plane ($d = 2$) [42].

More recently, Bhore and Tóth [13] established a lower bound of $\Omega(\varepsilon^{-d/2})$ for the lightness of Steiner $(1 + \varepsilon)$-spanners in Euclidean $d$-space for all $d \geq 2$. Moreover, for points in the plane, they established an upper bound of $O(\varepsilon^{-1})$ [12].

### 1.2 Our Contributions

We present the main contributions of this paper, and sketch the key technical and conceptual ideas used for establishing these results. (Refer to the technical sections for precise definitions, complete proofs, and additional remarks.)

**Points on a line.** In Section 2 (Theorem 3), we establish a lower bound $\Omega(\varepsilon^{-1} \log n / \log \varepsilon^{-1})$ for the competitive ratio of any online algorithm for a sequence of points on the real line. Moreover, we show that this bound is tight. We present an online algorithm that maintains a $(1 + \varepsilon)$-spanner with competitive ratio $O(\varepsilon^{-1} \log n / \log \varepsilon^{-1})$.

Our online algorithm is a 1-dimensional instantiation of hierarchical clustering, which was used by Roditty [49] for dynamical spanners in doubling metrics. When a new point $s_i$ is “close” to a previous point $s_j$, we add $s_i$ to the “cluster” of $s_j$, otherwise we open a new cluster. The key question is to define when $s_i$ is “close” to a previous point. Instead of the closest points on the line, we find the shortest edge $pq$ that contains $s_i$ in the current spanner, and say that $s_i$ is “close” to $p$ (resp., $q$) if $\|ps_i\| \leq \frac{1}{4}\|pq\|$ (resp., $\|qs_i\| \leq \frac{1}{4}\|pq\|$). The algorithm (and its analysis), does not explicitly maintain “clusters,” though. It is easy to show, by induction, that ALG maintains a $(1 + \varepsilon)$-spanner. The main contribution is a tight analysis of the competitive ratio. We partition the edges into buckets by weight, where bucket $E_\ell$ contains edges $e$ of weight $\varepsilon^{-(\ell+1)} < \|e\| \leq \varepsilon^\ell$. The edges of the spanner will
form a laminar family (any edges are interior-disjoint or one contains the other); and the edge weight decay by factors of at most \((1 - \frac{1}{4})^{4/\varepsilon} < \frac{1}{4}\), we can show that the total weight of edges in a level decreases by a factor of \(\frac{1}{3}\) after every \([5/\varepsilon]\) levels. Thus, the sum of edge weights in a block of \([5/\varepsilon]\) consecutive levels is \(O(\varepsilon^{-1}\text{OPT})\). This bound, applied to \(O(\log n, n) = O(\log n / \log \varepsilon^{-1})\) buckets, proves the upper bound. The lower bound construction matches the upper bound for each block of levels and for each bucket.

**Euclidean \(d\)-space without Steiner points.** In Section 3, we study the online Euclidean spanners for a sequence of points in \(\mathbb{R}^d\). For constant \(d \geq 2\) and parameter \(\varepsilon > 0\), we show that the dynamic algorithm by Fischer and Har-Peled achieves, in the online model, competitive ratio \(O(\varepsilon^{-d+1})\) for \(n\) points in \(\mathbb{R}^d\) (Theorem 4 in Section 3.1), matching the competitive ratio of \(\text{DefSpanner}\) by Gao et al. [30, Lemma 3.8].

The new competitive analysis of this algorithm is instrumental for extending the algorithm and its analysis to online Steiner \((1 + \varepsilon)\)-spanners (see below). We briefly describe a key geometric insight. It is well known that for \(a, b \in \mathbb{R}^d\), any \(ab\)-path of weight at most \((1 + \varepsilon)\|ab\|\) lies in an ellipsoid \(B_{ab}\) with foci \(a\) and great axes \((1 + \varepsilon)\|ab\|\). Summation over disjoint ellipsoids gives a lower bound for OPT. Unfortunately, ellipsoids \(B_{ab}\) for all pairs \(ab \in S\) may heavily overlap. Recently, Bhore and Tóth [13, Lemma 3] proved that any \(ab\)-path of weight at most \((1 + \varepsilon)\|ab\|\) must contain an edge of total weight at least \(\frac{1}{2}\|ab\|\) that are “near-parallel” to \(ab\) (technically, they make an angle at most \(\varepsilon^{1/2}\) with \(ab\)); see Fig. 4(right). By partitioning the edges of the unknown OPT spanner by both directions and disjoint ellipsoids, we obtain a bound of \(\frac{\text{ALG}}{\text{OPT}} \leq O(\varepsilon^{-d+1}\log n)\).

**Euclidean \(d\)-space with Steiner points.** When we are allowed to use Steiner points, we can substantially improve the competitive ratio in terms of \(\varepsilon\): We describe an algorithm with competitive ratio \(O(\varepsilon^{(1-d)/2})\log n\) (Theorem 5 in Section 3.2).

The online Steiner algorithm adds a secondary layer to the non-Steiner algorithm: For each edge \(ab\) of the non-Steiner spanner \(G_1\), we maintain a path of weight \((1 + \varepsilon)\|ab\|\) with Steiner points; the stretch factor of the resulting Steiner spanner \(G_2\) is \((1 + \varepsilon)^2 < (1 + 3\varepsilon)\). The key idea is to reduce the weight to maintain buckets of edges of \(G_1\) that have roughly the same direction and weight, and are nearby locations; and we construct a common Steiner network \(N\) for them. Importantly, we can construct a “backbone” of the network \(N\) when the first edge \(ab\) in a bucket arrives, and we have \(\|N\| \leq O(\varepsilon^{(1-d)/2}\|ab\|)\). When subsequent edges \(a'b'\) in the same bucket arrive, then we can add relatively short “connectors” to \(N\) so that it also contains an \(a'b'\)-path of weight at most \((1 + \varepsilon)\|a'b'\|\). Thus \(N\) can easily accommodate new paths in the online model. The key technical tool for constructing Steiner networks \(N\) (one for each bucket) is the so-called shallow-light trees, introduced by Awerbuch et al. [7] and Khuller et al. [41], and optimized in the geometric setting by Elkin and Solomon [28, 52].

As a counterpart, we show (Theorem 7 in Section 4) that the dependence on \(n\) cannot be eliminated in dimensions \(d \geq 2\). In particular, we prove that any \((1 + \varepsilon)\)-spanner for a sequence of \(n\) points in \(\mathbb{R}^d\), has competitive ratio \(\Omega(f(n))\) for some function \(f(n)\) with \(\lim_{n \to \infty} f(n) = 0\). The lower bound construction consists of an adaptive strategy for the adversary in the plane: The adversary recursively maintains a space partition and places points in rounds so that the spanner constructed so far is disjoint from most of the ellipses \(B_{ab}\) that will contains the \(ab\)-paths for pairs of new points \(a, b\). In order to control OPT, the adversary maintains the property that \(\text{OPT}_i\) is an \(x\)-monotone path \(\gamma_i\) after round \(i\). However, this requirement means that any new point must be very close to \(\gamma_i\), and \(S\) will be a set of almost collinear points. The core challenge of the Steiner spanner problem seems to lie in the case of almost collinear points.
Higher dimensions under the $L_1$-norm. Finally, in Section 5 we provide improved lower bounds for points in $\mathbb{R}^d$ under the $L_1$ norm (without Steiner points). We show that for every $\varepsilon > 0$, under the $L_1$ norm, the competitive ratio of any online $(1 + \varepsilon)$-spanner algorithm $O(\varepsilon^{-2}/\log \varepsilon^{-1})$ in $\mathbb{R}^2$ and is $\Omega(\varepsilon^{-d})$ in $\mathbb{R}^d$ for $d \geq 3$.

The adversary takes advantage of the non-monotonicity of $\text{OPT}$, mentioned above. In round 1, it presents a point set $S_1 \cup S_2$ for which any $(1 + \varepsilon)$-spanner (without Steiner points) must contain a complete bipartite graph between $S_1$ and $S_2$; however the optimal Steiner $(1 + \varepsilon)$-spanner for $S_1 \cup S_2$ has much smaller weight. Then in round 2, the adversary presents all Steiner points $S_1 \cup S_2$ of an optimal Steiner $(1 + \varepsilon)$-spanner for $S_1 \cup S_2$. The key insight is that under the $L_1$-norm (and for this particular point set), the optimal Steiner spanner for $S_1 \cup S_2$ already contains Manhattan paths between any two points in $S = (S_1 \cup S_2) \cup (S_1 \cup S_2)$, and so it remains the optimum solution (without Steiner points) for the point set $S$.

We were unable to replicate this phenomenon under the $L_2$-norm, where the current best lower bound in $\mathbb{R}^d$, for all $d \geq 1$, derives from the 1-dimensional construction. In particular, it is not sufficient to consider the Steiner ratio for $(1 + \varepsilon)$-spanners, defined as the supremum ratio between the weight of the minimum $(1 + \varepsilon)$-spanner and the minimum Steiner $(1 + \varepsilon)$-spanner of a finite point set in $\mathbb{R}^d$. Under the $L_2$-norm, this ratio is $\Theta(\varepsilon^{-1})$ in the plane and $\tilde{\Theta}(\varepsilon^{(1-d)/2})$ in $\mathbb{R}^d$ for $d \geq 3$ [12, 42, 44]. However, an optimal Steiner $(1 + \varepsilon)$-spanner, need not achieve the desired $1 + \varepsilon$ stretch factor for the Steiner points.

## 2 Lower and Upper Bounds for Points on a Line

It is easy to analyze the one-dimensional case as the offline optimum network (OPT) for any set of points in a line is a path from the leftmost point to the rightmost point; the stretch factor of this path is always 1. (In contrast, in 2- and higher dimensions, the optimum $(1 + \varepsilon)$-spanner is highly dependent on the distribution of points, which in turn may change over time in the online model.)

![Figure 2 Left: A sequence of $n$ points $(q_0, p_0, p_1, p_2, \ldots, p_{n-2})$, for $n = \lfloor \varepsilon^{-1} \rfloor$, for which any online $(1 + \varepsilon)$-spanner has weight $\Omega(\varepsilon^{-1}\text{OPT})$. For clarity, the edges are drawn as circular arcs, but the weight of an edge $p_i p_j$ is $\|p_i p_j\| = |p_i - p_j|$. Right: Iteration in each subinterval.](image)

**Lower bound.** The following adversarial strategy establishes a lower bound $L(n) = \Omega(\varepsilon^{-1})$ for the competitive ratio; refer to Fig. 2 (left). Start with two points $p_0 = 0$ and $q_0 = 1$. For the first two points, $\text{ALG}$ must add a direct edge $p_0 q_0$. Then the adversary successively places points $p_i = i \cdot \frac{\varepsilon}{2}$, for $i = 1, \ldots, n$ so that all points remain in the interval $[0, \frac{1}{2}]$. Thus the number of points is $n = 2 + \lceil \varepsilon^{-1} \rceil$. In each round, $\text{ALG}$ must add the edge $p_i q_0$, otherwise any path between $p_i$ and $q_0$ would have to make a detour via a point in $\{p_0, \ldots, p_{i-1}\}$, and so it would be longer than $(1 + \varepsilon)\|p_i q_0\|$. Since $\|p_i q_0\| \geq \frac{1}{2}$, the weight of the network after
n − 2 iterations is at least \( \text{ALG} \geq 1 + \frac{1}{2}(n - 2) \geq 1 + \frac{1}{2} \varepsilon^{-1} \). Combined with \( \text{OPT} = 1 \), this yields a lower bound of \( \Omega(\varepsilon^{-1}) \) for the competitive ratio.

The adversary has placed only \( O(\varepsilon^{-1}) \) points so far; this is the first stage of the strategy. In subsequent stages, the adversary repeats the same strategy in every subinterval \( ab \) of previous stage, as indicated in Fig. 2 (right). After stage \( j \geq 1 \), we have \( \text{ALG} \geq 1 + \frac{1}{2} \varepsilon^{-1} \) and \( \text{OPT} = 1 \). The number of points placed in each stage increases by a factor of \( \Omega(\varepsilon^{-1}) \), hence \( j = \Theta(\log_{\varepsilon^{-1}} n) = \Theta(\log n / \log \varepsilon^{-1}) \). Overall, the competitive ratio is at least \( \text{ALG}/\text{OPT} \geq \Omega(j \varepsilon^{-1}) = \Omega(\varepsilon^{-1} \log n / \log \varepsilon^{-1}) \).

**Upper bound.** For proving a matching upper bound in one-dimension, we use the following online algorithm: For all \( i = 1, \ldots, n \), we maintain a spanning graph \( G_i \) on \( S_i = \{s_1, \ldots, s_i\} \) and the \( x \)-monotone path \( P_i \) between the leftmost and the rightmost points in \( S_i = \{s_1, \ldots, s_i\} \). When point \( s_i \), \( i \geq 2 \), arrives, we proceed as follows (see Fig. 3).

If \( s_i \) is left (resp., right) of all previous points, we add an edge from \( s_i \) to the closest point in \( S_{i-1} \) to both \( P_{i-1} \) and \( G_{i-1} \). Otherwise, let \( ab \) be the (unique) edge of \( P_{i-1} \) which contains \( s_i \), and \( pq \) a shortest edge of \( G_{i-1} \) that contains \( s_i \). Clearly, we have \( P_i = P_{i-1} - ab + as_i + sb \).

If \( \min\{\|ps_i\|, \|s_iq\|\} > \frac{1}{2} \|pq\| \), we add both \( as_i \) and \( sb \) to \( G_i \), that is, \( G_i = G_{i-1} + as_i + sb \).

Otherwise, let \( G_i = G_{i-1} + as_i \) if \( \|ps_i\| \leq \|s_iq\| \), or else \( G_i = G_{i-1} + sb \).

We observe a few properties of \( G_i \) that are immediate from the construction: (P1) At the time when edge \( e \) is added to \( G_i \), then the interior of \( e \) does not contain any vertices. (P2) The edges in \( G_i \) form a laminar set of intervals (i.e., any two edges are interior-disjoint, or one contains the other). (P3) If \( e_1, e_2 \) are edges in \( G_i \) and \( e_2 \subset e_1 \), then \( \|e_2\| \leq (1 - \frac{1}{4})\|e_1\| \).

We note that properties (P1)-(P3) are inherently 1-dimensional, as the edges are intervals in \( \mathbb{R} \), and they do not seem to generalize to higher dimensions.

**Lemma 1.** For \( i = 1, \ldots, n \), the graph \( G_i \) is a \((1 + \varepsilon)\)-spanner for \( S_i \).

**Proof.** We proceed by induction on \( i \). The base case \( i = 1 \) is trivial. Assume that \( i > 1 \) and \( G_{i-1} \) is a \((1 + \varepsilon)\)-spanner for \( S_{i-1} \). It is enough to show that for every \( j = 1, \ldots, i - 1 \), \( G_i \) contains an \( s_is_j \)-path of weight at most \( (1 + \varepsilon)\|s_is_j\| \). We consider all cases step \( i \) of the algorithm.

**Case 1:** Assume that \( s_i \) is left of (resp., right of) \( S_{i-1} \), and the closest previous point is \( a \in S_{i-1} \). By induction, \( G_{i-1} \) contains a path \( \pi(a, s_j) \) of weight at most \( (1 + \varepsilon)\|s_ia\| \). Hence the weight of the path \( sia + \pi(a, s_j) \) is at most \( \|s_ia\| + (1 + \varepsilon)\|s_ia\| < (1 + \varepsilon)\|s_ia\| + \|as_j\| = (1 + \varepsilon)\|s_ia\| \).

**Case 2:** Assume that \( pq \) is the shortest edge of \( G_{i-1} \) that contains \( s_i \), and \( a, b \in S_{i-1} \) are the
closest previous points to $s_i$ on the left and right, respectively. If $\min\{||p_s||, ||s,q||\} > \frac{3}{4} ||pq||$, then $G_i = G_{i-1} + a_s + s,b$, and we can argue similarly to Case 1.

Otherwise, we have $\min\{||p_s||, ||s,q||\} \leq \frac{3}{4} ||pq||$. Assume w.l.o.g. that $||p_s|| \leq ||s,q||$, consequently $G_i = G_{i-1} + a_s$. If $s_j$ is to the left of $s_i$, we can argue similarly to Case 1. Hence we may assume that $s_j$ is to the right of $s_i$.

Property (P2) implies that $G_{i-1}$ contains an $x$-monotone $ap$-path $\pi(a,p)$, which has optimal weight $\|ap\|$; and an $x$-monotone $qb$-path $\pi(q,b)$ of weight $\|bq\|$.

Property (P1) implies that $a = p$ or $a$ was inserted after $p$ and $q$; and since $a$ is not the left endpoint of any edge in $G_i$, then $\|ap\| \leq \frac{3}{4} \|pq\|$. Analogously, we have $\|bq\| \leq \frac{3}{4} \|pq\|$.

Consequently,

$$||s_i,b|| = ||pq|| - ||ps_i|| - \|bq\| \geq \left(1 - 2 \cdot \frac{3}{4}\right) ||pq|| = \left(1 - \frac{3}{2}\right) \|pq\|.$$ 

Now we can construct an $s_ib$-path $\pi(s_i,b) = s_i a + \pi(a,p) + pq + \pi(q,b)$ of weight at most

$$||\pi(s_i,b)|| \leq ||s,p|| + ||pq|| + \|bq\| \leq \left(1 + \frac{\varepsilon}{2}\right) \|pq\| \leq \frac{1 + \varepsilon}{1 - \varepsilon/2} \|s_i,b\| < (1 + \varepsilon)\|s_i,b\|.$$ 

We can complete the proof now. By induction, $G_{i-1}$ contains a $bs_j$-path $\pi(b,s_j)$ of weight at most $(1 + \varepsilon)\|bs_j\|$. Hence the weight of the path $\pi(s_i,b) + \pi(b,s_j)$ is at most $(1 + \varepsilon)\|s_i,b\| + (1 + \varepsilon)\|bs_j\| = (1 + \varepsilon)(\|s_i,b\| + \|bs_j\|) = (1 + \varepsilon)||s_i,s_j||$, as required. \hfill \blacktriangleleft

**Lemma 2.** For $i = 1, \ldots, n$, we have $\|G_i\| \leq O(\varepsilon^{-1} \OPT; \log i / \log \varepsilon^{-1})$.

**Proof.** We may assume w.l.o.g. that $i = n$, and let $\OPT = \OPT_n$ for brevity. Let $E$ be the edge set of $G_n$. The order in which ALG adds edges to $E$ defines a (precedence) poset on $E$. We partition $E$ by weight as follows: Let $\beta = \varepsilon^{-1}$; and for all $\ell \in \mathbb{Z}$, let $E_\ell$ be the set of edges $e \in E$ with $\beta^\ell < ||e|| \leq \beta^{\ell+1}$. Since $||e|| \leq \OPT$ for all $e \in E$, every edge is in $E_\ell$ for some $\ell \leq \log_\beta \OPT$. Furthermore, for all $\ell \leq \log_\beta (\OPT/n^2)$, the edges $e \in E_\ell$ have weight $||e|| \leq \OPT/n^2$, and so the total weight of these edges is less than $\OPT$. It remains to consider $E_\ell$ for $\ell \leq \log_\beta (\OPT/n^2) = \ell \leq \log_\beta \OPT$, that is, for $O(\log n/\log \varepsilon^{-1})$ values of $\ell$.

Let $pq$ be an edge in $E_\ell$ that is not contained in any previous edge in $E_\ell$. By property (P2), the edges in $E_\ell$ form a laminar family, and so $pq$ does not overlap with any previous edge in $E_\ell$; and $pq$ contains any subsequent edge that overlaps with it. Let $E_\ell(pq)$ be the set of all edges in $E_\ell$ that are contained in $pq$ (including $pq$). We claim that

$$\|E_\ell(pq)\| \leq O(\varepsilon^{-1} \|pq\|).$$

(1)

Summation over all edges $pq \in E_\ell$ that are not contained in previous edges in $E_\ell$ implies $\|E_\ell\| \leq O(\varepsilon^{-1} \OPT)$. Summation over all $\ell \in \mathbb{Z}$ then yields

$$\|E\| = \sum_{\ell \in \mathbb{Z}} \|E_\ell\| \leq \sum_{\ell = \lceil \log_\beta (\OPT/n^2) \rceil}^{\log_\beta \OPT} \|E_\ell\| + O(\OPT) = O(\varepsilon^{-1} \OPT \log_\beta n).$$

To prove (1), consider the containment poset of $E_\ell(pq)$. In fact, we represent the poset as a rooted binary tree $T$: The root corresponds to $pq$, and edges $e_1, e_2 \in E_\ell(pq)$ are in parent-child relation if $e_2 \subset e_1$, and there is no edge $e' \in E_\ell(pq)$ with $e_2 \subset e' \subset e_1$. Each level of $T$ corresponds to interior-disjoint edges contained in $\|pq\|$, so the sum of weight on each level is at most $\|pq\|$. The total weight of the first $k = \lceil 5 \varepsilon^{-1} \rceil$ levels is $O(\varepsilon^{-1} \|pq\|)$.

We claim that the total weight on level $k = \lceil 5 \varepsilon^{-1} \rceil$ is at most $\frac{4}{3} \|pq\|$. We distinguish between three types of nodes in the subtree of $T$ between levels 0 and $k$: A **branching node** has two children, a **single-child node** has one child, and a **leaf** has no children (in particular...
all nodes in level \( k \) are considered leaves in this subtree). The nodes (leaves) at level \( k \) correspond to interior-disjoint edges \( e \subset pq \) with \( ||e|| \geq \varepsilon ||pq|| \) by the definition of \( E_\varepsilon(pq) \). Thus there are at most \( \lfloor \varepsilon^{-1} \rfloor \) nodes at level \( k \), hence there are less than \( \lfloor \varepsilon^{-1} \rfloor \) branching nodes. This implies that for any node \( e \) on level \( k \), the descending path from the root \( pq \) to \( e \) contains at least \( k - \lfloor \varepsilon^{-1} \rfloor \geq \lfloor 4\varepsilon^{-1} \rfloor \) single-child nodes.

For the purpose of bounding the total weight at level \( k \), we can modify \( T \), by incrementally moving all single-child nodes below all branching nodes as follows. While there is an edge \( uv \) in \( T \), such that \( u \) is a branching node, and its parent \( v \) is a single-child node, we suppress \( u \) and subdivide the two edges of \( T \) below \( u \) with new nodes \( v_1 \) and \( v_2 \). The weight along the edge \( uv \) goes down by a factor of at most \( (1 - \frac{\varepsilon}{4}) \) by property (P3); we set the weights in the modified tree such that the same decrease occurs along the edges \( uv_1 \) and \( v_2 \). Then each operation maintains property (P3), and the total weight at level \( k \) does not change. When the while loop terminates, we obtain a full binary tree with a chain attached to each leaf.

As we argued above, each chain has length \( \lfloor 4/\varepsilon \rfloor \) or more. The full binary tree does not necessarily decrease the weight. Along each chain of \( \lfloor 4/\varepsilon \rfloor \) or more single-child nodes, the weight is cumulatively multiplied by a factor of at most \( (1 - \frac{\varepsilon}{4})^{\lfloor 4/\varepsilon \rfloor} < \frac{1}{\varepsilon} \). Overall, the total weight at level \( k = \lceil 5\varepsilon^{-1} \rceil \) is at most \( \frac{1}{\varepsilon} ||pq|| \), as claimed.

By induction, for every integer \( j \geq 0 \), the total weight at level \( jk = j\lfloor 5\varepsilon^{-1} \rfloor \) is at most \( \frac{1}{\varepsilon} ||pq||/2^j \). Consequently, the total weight of a block of \( k \) consecutive levels \( \{jk + 1, \ldots, (j + 1)k\} \) is at most \( k||pq||/2^j \). Overall, \( ||E_\varepsilon(pq)|| = \sum_{j \geq 0} k||pq||/2^j = O(k||pq||) = O(\varepsilon^{-1} ||pq||) \), which completes the proof of (1).

We can summarize the discussion above in the following theorem.

\textbf{Theorem 3.} For every \( \varepsilon > 0 \), the competitive ratio of any online algorithm for \((1 + \varepsilon)\)-spanners for a sequence of points on a line is \( \Omega(\varepsilon^{-1} \log n/\log \varepsilon^{-1}) \). Moreover, there is an online algorithm that maintains a \((1 + \varepsilon)\)-spanner with competitive ratio \( O(\varepsilon^{-1} \log n/\log \varepsilon^{-1}) \).

\section{Upper Bounds for Spanners in \( \mathbb{R}^d \) under the \( L_2 \) Norm}

We turn to online \((1 + \varepsilon)\)-spanners in Euclidean \( d \)-space for \( d \geq 2 \). The dynamic algorithm DefSpanner by Gao et al. [30], based on hierarchical clustering, achieves \( O(\varepsilon^{-(d+1)} \log n) \) competitive ratio in the online model. In Section 3.1, we recover the same bound with a new analysis, where we refine the hierarchical clustering with a partition of the edges into buckets of similar directions, locations, and weights. In Section 3.2, we extend the new analysis to show that the competitive ratio improves to \( O(\varepsilon^{(1-d)/2} \log n) \) if we are allowed to use Steiner points. Our spanner algorithm replaces each bucket of “similar” edges with a Steiner network using grids and shallow-light trees, for up to \( O(\varepsilon^{(1-d)/2}) \) directions.

\textbf{Preliminaries.} Well-separated pair-decomposition (for short, WSPD) of a finite point set \( S \) in a metric space is a classical tool for constructing \((1 + \varepsilon)\)-spanners \([19, 34, 47, 51]\). It is a collection of pairs \( \{(A_i, B_i) : i \in I\} \) such that for all \( i \in I \), we have \( A_i, B_i \subset S \) and \( \max\{\text{diam}(A_i), \text{diam}(B_i)\} \leq \varepsilon \text{dist}(A_i, B_i) \); and for every point pair \( \{s, t\} \subset \binom{S}{2} \), there is a pair \( (A_i, B_i) \) such that \( A_i \) and \( B_i \) each contains precisely one of \( s \) and \( t \). It was shown by Callahan and Kosaraju [18] that if a graph \( G = (S, E) \) contains an edge between arbitrary points in \( A_i \) and \( B_i \), for all \( i \in I \), then \( G \) is an \((1 + O(\varepsilon))\)-spanner for \( S \); see also [47, Ch. 9].

Dynamic spanners (including the fully dynamic algorithm by Roditty [49] and DefSpanner by Gao et al. [30]) rely on WSPDs and hierarchical clustering. In \( \mathbb{R}^d \), hierarchical clustering can be obtained by classical recursive space partitions such as quadtrees [27, Ch. 14]. Dynamic quadtrees and their variants have been studied extensively, due to their broad range
of applications; see [38, Ch. 2]. In general, dynamic quadtrees can handle both point insertion and deletion operations. However, in the context of an online algorithm, where the points are only inserted, note that no cell of the quadtree is ever deleted. We analyse the competitive ratio of the dynamic incremental algorithm by Fischer and Har-Peled [29] that maintains an $(1 + \varepsilon)$-spanner for $n$ points in Euclidean $d$-space in expected $O([\log n + \log \varepsilon^{-1}]\varepsilon^{-d} \log n)$ update time. However, they have not analyzed the ratio between the weight of the resulting $(1 + \varepsilon)$-spanner and the minimum weight of an $(1 + \varepsilon)$-spanner.

### 3.1 Online Algorithm without Steiner Points

**Online Algorithm.** We briefly review the algorithm in [29] and then analyze the weight. The input is a sequence of points $(s_1, s_2, \ldots)$ in $\mathbb{R}^d$; the set of the first $n$ points is denoted by $S_n = \{s_i : 1 \leq i \leq n\}$. For every $n$, we dynamically maintain a quadtree $T_n$ for $S_n$. Every node of $T_n$ corresponds to a cube. The root of $T_n$, at level 0, corresponds to a cube $Q_0$ of side length $a_0 = \Theta(\text{diam}(S_n))$. At every level $\ell \geq 0$, there are at most $2^{d\ell}$ interior-disjoint cubes, each of side length $a_0/2^\ell$. A cube $Q \in T_n$ is nonempty if $Q \cap S_n \neq \emptyset$. For every nonempty cube $Q$, we select an arbitrary representative $s(Q) \in Q \cap S_n$. At each level $\ell$, let $E_\ell$ be the set of all edges $s(Q_1)s(Q_2)$ for pairs of cubes $\{Q_1, Q_2\}$ on level $\ell$ such that $\frac{c_1a_0}{2^\ell} \leq \|s(Q_1)s(Q_2)\| \leq \frac{c_2a_0}{2^\ell}$ for some constants $0 < c_1 < c_2$ that depend on $d$; see Fig. 4(left).

The algorithm maintains the spanner $G = (S_n, E)$ where $E = \bigcup_{\ell \geq 0} E_\ell$. A classical argument by Callahan and Kosaraju [18] (see also [34, 47, 51]) shows that $G$ is a $(1 + \varepsilon)$-spanner for $S_n$.

![Figure 4](image)

Figure 4 Left: Nonempty squares at level $\ell = 4$ of a quadtree, each with a representative (red dots). Point $s(Q)$ is connected to all other representatives in the annulus between the concentric circles $C_1$ and $C_2$ of radii $c_1/(\varepsilon 2^\ell)$ and $c_2/(\varepsilon 2^\ell)$. Right: Ellipse $B_{ab}$ with foci $a$ and $b$, an $ab$-path of weight $(1 + \varepsilon)\|ab\|$. The bold edges make an angle at most $\varepsilon^{1/2}$ with $ab$.

**Theorem 4.** For every constant $d \geq 2$, parameter $\varepsilon > 0$, and a sequence of $n \in \mathbb{N}$ points in Euclidean $d$-space, the competitive ratio of the online algorithm above is in $O(\varepsilon^{-(d+1)} \log n)$.

**Proof.** For the set $S_n$ of the first $n$ points of a sequence in $\mathbb{R}^d$, let $G = (S_n, E)$ be the $(1 + \varepsilon)$-spanner produced by the online algorithm, and let $G^* = (S_n, E^*)$ be an $(1 + \varepsilon)$-spanner of minimum weight. We show that $\|G\|/\|G^*\| = O(\varepsilon^{-(d+1)} \log n)$.

**Short edges.** Note that the weight of every edge in $E_\ell \subset E$ at level $\ell$ is $\Theta(\varepsilon^{-1}\text{diam}(S_n)/2^\ell)$, since it connects representatives at $\Theta(\varepsilon^{-1}\text{diam}(S_n)/2^\ell)$ distance apart. In particular, an edge at any level $\ell \geq 2 \log n$ has weight at most $O(\varepsilon^{-1}\text{diam}(S_n)/n^2)$; and the total weight of
these edges is \( O(\varepsilon^{-1}\text{diam}(S_n)) \leq O(\varepsilon^{-1}\text{OPT}) \). It remains to bound the weight of the edges on levels \( \ell = 1, \ldots , \lfloor 2 \log n \rfloor \). We consider each level separately.

\textbf{Ellipsoids and directions.} For every edge \( ab \in E \), let \( B_{ab} \) denote the ellipsoid with foci \( a \) and \( b \), and great axis of length \( (1 + \varepsilon)\|ab\| \). Note that every \( ab \)-path of weight at most \( (1 + \varepsilon)\|ab\| \) lies in \( B_{ab} \). The set of directions of line segments in \( \mathbb{R}^d \) is represented by a hemisphere of \( S^{d-1} \). The distance between two directions is measured by angles in the range \([0, \pi]\). Recently, Bhore and Tóth [13, Lemma 3] proved that every \( ab \)-path of weight at most \( (1 + \varepsilon)\|ab\| \) contains edges of total weight at least \( \frac{1}{2}\|ab\| \) that make an angle at most \( \varepsilon^{1/2} \) with \( ab \) (i.e., they are near-parallel to \( ab \)); see Fig. 4(right).

Since \( G^* \) is a \((1 + \varepsilon)\)-spanner for \( S_n \), it contains an \( ab \)-path of weight at most \((1 + \varepsilon)\|ab\| \) for every \( ab \in E \). This path lies in the ellipsoid \( B_{ab} \), and contains edges of \( G^* \) of weight at least \( \frac{1}{2}\|ab\| \) and with direction with at most \( \varepsilon^{1/2} \) from \( ab \). We next define suitable disjoint sets of ellipsoids, in order to establish a lower bound on \( \|G^*\| \).

\textbf{Edge partition by directions.} First, we partition the edge set \( E_\ell \) into subsets based on the directions of the edges. We use standard volume argument to construct a homogeneous set of directions. Let \( H \subset S^{d-1} \) be the hemisphere of unit vectors in \( \mathbb{R}^d \), then the direction vector of a line segment \( ab \), denoted \( \text{dir}(ab) \), is a unique point in \( H \). Consider a maximal packing of \( H \) with (spherical) balls of radius \( \frac{1}{2}\varepsilon^{1/2} \). Since the spherical volume of \( H \) is \( \Theta(1) \) and the volume of each ball is \( \Theta(\varepsilon^{(d-1)/2}) \), the number of balls is \( K = \Theta(\varepsilon^{(1-d)/2}) \).

By doubling the radii of the spherical balls to \( \frac{1}{2}\varepsilon^{1/2} \), we obtain a covering of \( H \) with a set of balls \( \mathcal{D} = \{ D_i : i = 1, \ldots , K \} \). For each spherical ball \( D_i \in \mathcal{D} \), denote by \( 2D_i \) the concentric ball of radius \( \frac{1}{2}\varepsilon^{1/2} \). By standard packing argument, the ball \( 2D_i \) intersects only \( O(1) \) balls in \( \mathcal{D} \) (where \( d = O(1) \)). We can now define a partition \( E_\ell = \bigcup_{i=1}^K E_\ell,i \) as follows: let an \( ab \in E_\ell \) be in \( E_\ell,i \) if \( i \) is the smallest index such that \( \text{dir}(ab) \in D_i \). Now for every \( i = 1, \ldots , K \), let \( E_\ell,i^* \) be the set of edges \( e^* \in E^* \) such that \( \text{dir}(e^*) \in 2D_i \). By construction, every edge \( e^* \in E^* \) lies in \( O(1) \) sets \( E_\ell,i^* \); consequently \( \sum_{i=1}^K \| E_\ell,i^* \| = \Theta(\|G^*\|) \). Furthermore, for every edge \( ab \in E_\ell,i \), all edges in \( E_\ell,i^* \) that make an angle at most \( \varepsilon^{1/2} \) with \( ab \) are in \( E_\ell,i^* \).

\textbf{Disjoint ellipsoids.} For every \( i = 1, \ldots , K \), let \( B_{\ell,i} \) be the set of ellipsoids \( B_{ab} \) with \( ab \in E_\ell,i \). We show that \( B_{\ell,i} \) contains a subset \( B'_{\ell,i} \) of disjoint ellipsoids such that \( |B'_{\ell,i}| \geq \Omega(\varepsilon^{d+1}|B_{\ell,i}|) \).

We claim that every ellipsoid in \( B_{\ell,i} \) intersects \( O(\varepsilon^{-(d+1)}) \) other ellipsoids in \( B_{\ell,i} \). We make use of a volume argument. Let \( M_\ell = \max\{\|e\| : e \in E_\ell \} \); and note that the side length of every cube at level \( \ell \) of the quadtree is \( \Theta(\varepsilon M_\ell) \).

For every ellipsoid \( B_{ab} \in B_{\ell,i} \), the great axis has length \((1 + \varepsilon)\|ab\| \), and the \( d - 1 \) minor axes each have length \( \sqrt{1 + \varepsilon^{2} - \|ab\|^{2}} < 2\varepsilon^{1/2}\|ab\| \), where \( \|ab\| \leq M_\ell \). Hence \( B_{ab} \) is contained in a cylinder \( C_{ab} \) of height \((1 + \varepsilon)M_\ell \) whose base is a \((d - 1)\)-dimensional ball of diameter \( 2\varepsilon^{1/2}M_\ell \). Any other ellipsoid in \( B_{\ell,i} \) with great axis parallel to \( ab \) is contained in a translate of \( C_{ab} \). If we rotate \( B_{ab} \) about its center by an angle at most \( \varepsilon^{1/2} \), then its orthogonal projection to the original great axis decreases, and the maximum distance from the original great axis increases by at most \( \|ab\|\frac{\varepsilon^{1/2}}{2} \sin \varepsilon^{1/2} < M_\ell\varepsilon^{1/2} \). Consequently, every ellipsoid in \( B_{\ell,i} \) is contained in a translated copy of \( 2C_{ab} \). Hence, every ellipsoid in \( B_{\ell,i} \) that intersects \( B_{ab} \) is contained in \( 3C_{ab} \). Every cube at level \( \ell \) of the quadtree that intersects \( 3C_{ab} \) is contained in the Minkowski sum of \( 3C_{ab} \) and such a cube, which is in turn contained in \( 4C_{ab} \). Note that the volume of the cylinder \( 4C_{ab} \) is \( O(\varepsilon^{(d-1)/2}M_\ell^d) \); while the volume of a cube at level \( \ell \) of the quadtree is \( \Theta(\varepsilon^d M_\ell^d) \). Therefore \( 4C_{ab} \) contains \( O(\varepsilon^{(d-1)/2}M_\ell^d) = O(\varepsilon^{-(d+1)/2}) \) such cubes. Recall that the algorithm maintains one representative from each cube, and the edges \( ab \in E_\ell,i \) are pairs of representative. Thus \( O(\varepsilon^{-(d+1)/2}) \) representatives in \( 4C_{ab} \) can form \( O(\varepsilon^{-(d+1)}) \) pairs (i.e., edges, hence ellipsoids).

This completes the proof of the claim that every ellipsoid in \( B_{\ell,i} \) intersects \( O(\varepsilon^{-(d+1)}) \)
other ellipsoids in $B_{i,t}$. Hence the intersection graph of $B_{i,t}$ is $O(\varepsilon^{-(d+1)})$-degenerate; and has an independent set $B_{i,t}^*$ of size $|B_{i,t}^*| \geq \Omega(\varepsilon^{d+1}|B_{i,t}|) = \Omega(\varepsilon^{d+1}\|E_{i,t}\|)$.

**Weight analysis.** As noted above, all edges in $E_t$ have length $\Theta(M_t)$. For every $i = 1, \ldots, K$ and for every ellipsoid $B_{ab} \in B_{i,t}$, we have $\|E_i \cap B_{ab}\| \geq \frac{1}{2}\|ab\|\Omega(M_t)$. Summing over a set of disjoint ellipsoids, we obtain

$$\|E_i\| \geq \sum_{ab \in B_{i,t}} \|E_i \cap B_{ab}\| \geq \sum_{ab \in B_{i,t}} \frac{1}{2}\|ab\|$$

$$\geq |B_{i,t}^*| \cdot \frac{1}{2} \min \{\|ab\| : ab \in E_{i,t}\}$$

$$\geq \varepsilon^{-(d+1)}\|E_{i,t}\| \cdot \Omega(M_t) = \Omega(\varepsilon^{-(d+1)}\|E_{i,t}\|).$$

Summation over all directions $i = 1, \ldots, K$ yields

$$\|G^*\| = \Theta\left(\sum_{i=1}^K \|E_i^*\|\right) \geq \Omega\left(\sum_{i=1}^K \varepsilon^{-(d+1)}\|E_{i,t}\|\right) = \Omega(\varepsilon^{-(d+1)}\|E_{1,t}\|).$$

Finally, summation over all $t \geq 1$ yields

$$\|E\| = \sum_{t \geq 1} \|E_t\| = \sum_{t=1}^{\lceil 2\log n \rceil} \|E_t\| + \sum_{t > \lceil 2\log n \rceil} \|E_t\| \leq \varepsilon^{-(d+1)}\|G^*\| \log n + \varepsilon^{-1}\|G^*\|,$$

as required.

### 3.2 Online Algorithm with Steiner Points

When Steiner points are allowed, we can substantially improve the competitive ratio in terms of $\varepsilon$. We describe an algorithm with competitive ratio $O(\varepsilon^{(1-d)/2}\log n)$. As a counterpart, we show in Section 4 that the dependence on $n$ is unavoidable in dimensions $d \geq 2$; it remains an open problem whether the dependence on $\varepsilon$ is necessary.

▶ **Theorem 5.** For every $\varepsilon > 0$, an online algorithm can maintain, for a sequence of $n \in \mathbb{N}$ points in the plane, a Euclidean Steiner $(1 + \varepsilon)$-spanner of weight $O(\varepsilon^{-1/2}\log n) \cdot \text{OPT}$.

**Proof.** Our online algorithm has two stages: $A_1$ and $A_2$. Algorithm $A_1$ is the same as in Section 3.1, it maintains a quadtree $T_n$ for the point set $S_n$, and a “primary” $(1 + \varepsilon)$-spanner $G_1$ without Steiner points. Algorithm $A_2$ maintains a Steiner $(1 + 3\varepsilon)$-spanner $G_2$ as follows: for each edge $ab$ in $G_1$, it creates an $ab$-path of length $(1 + \varepsilon)\|ab\|$ using Steiner points in $G_2$. Importantly, algorithm $A_2$ can bundle together “similar” edges of $G_1$, and handle them together using shallow-light trees [52].

In particular, we partition the space of all possible edges of $G_1$ into *buckets* (edges with similar directions, locations, and weights). For each bucket $U$, when algorithm $A_1$ inserts the first edge $ab \in U$ into $G_1$, then algorithm $A_2$ creates a “backbone” Steiner tree $T = T(U)$ of weight $O(\|ab\|)$, which contains an $ab$-path of length at most $(1 + \varepsilon)\|ab\|$. For any subsequent edge $a'b' \in U$, is suffices to add paths from $a'$ and $b'$ to $T$, of weight $O(\varepsilon \|ab\|)$, to obtain $a'b'$-path of length at most $(1 + \varepsilon)\|a'b'\|$. Overall, between any two points $s_i, s_j \in S$, the primary spanner contains a path of weight at most $(1 + \varepsilon)\|s_is_j\|$, and $G_2$ contains an Steiner path of weight at most $(1 + \varepsilon)^2\|s_is_j\| < (1 + 3\varepsilon)\|s_is_j\|$, as claimed.

It remains to define the buckets $U$, the backbone $T(U)$ for the first edge in $U$, and the “connectors” added for each subsequent edge in $U$. We first describe the algorithm in
the plane, where we establish a competitive ratio \(O(\varepsilon^{-1/2} \log n)\), and then generalize the construction to higher dimensions.

**Buckets.** We define buckets for all potential edges in the primary spanner \(G_1\). We analyze a single level \(\ell\) of the quadtree \(T\). Without loss of generality, assume that the side length of all quadtree cubes in level \(\ell\) have unit length, hence the weight of every edge in \(E_\ell\) is \(\Theta(\varepsilon^{-1})\).

In Section 3.2, we have covered the set \(H \subset S^1\) of directions with a set \(\mathcal{D} = \{D_i : i = 1, \ldots, K\}\) of balls of diameter \(\varepsilon^{1/2}\). For each ball in \(\mathcal{D}\), we define a set of buckets. Let \(D \in \mathcal{D}\) and let \(L\) be a line such that \(\text{dir}(L)\) corresponds to the center of \(D\); refer to Fig. 5(left).

Partition the plane into parallel strips of width \(\frac{1}{2}\varepsilon^{1/2}\) by a set of lines parallel to \(L\); and partition each strip further into rectangles of height \(2\varepsilon^{-1}\). By scaling up the rectangles by a factor of 2, we obtain a covering of the square \(Q_0\) with a set \(\mathcal{R}\) of \(4\varepsilon^{-1} \times \varepsilon^{1/2}\) rectangles such that each point is covered by \(O(1)\) rectangles in \(\mathcal{R}\).

For each rectangle \(R \in \mathcal{R}\), we create a bucket \(U\) comprising all edges \(ab \in E_\ell\) such that \(ab \subset R\) and \(\text{dir}(ab) \in D\) (hence \(\angle(\text{dir}(ab), \text{dir}(L)) \leq \varepsilon^{1/2}\)). Note that every edge \(ab \in E_\ell\) lies in at least one and at most \(O(1)\) buckets.

![Figure 5 Left: The overlay the the quadtree with a partition of \(\mathbb{R}^2\) into \(\frac{1}{2}\varepsilon^{-1/2} \times 2\varepsilon^{-1}\) rectangles aligned with \(L\). Top-Right: A rectangle \(R \in \mathcal{R}\), the median \(L(U)\), the grid \(G(U)\), and the partition of \(R\) into \(\varepsilon^{-1/2} \times \varepsilon^{-1/2}\) squares. Bottom-Right: A shallow-light tree between a side of an \(\frac{1}{2}\varepsilon^{-1/2} \times \frac{1}{2}\varepsilon^{-1/2}\) square and a source \(r \in L(U)\).](image)

**Backbones and Connectors.** Let \(U\) be a bucket defined above for a rectangle \(R \in \mathcal{R}\). Let \(L(U)\) denote the median of the rectangle \(R\) parallel to \(L\). When the primary algorithm \(A_1\) inserts the first edge \(ab \in U\) into \(G_1\), then Algorithm \(A_2\) constructs a unit grid graph \(G(U)\), formed by a subdivision of \(R\) into unit squares; see Fig. 5(top-right). Since \(R\) is a \(4\varepsilon^{-1} \times \varepsilon^{-1/2}\) rectangle, \(\|G(U)\| = O(\varepsilon^{-3/2})\). Furthermore, we partition \(R\) into \(4\varepsilon^{-1/2}\) squares of side length \(\varepsilon^{-1/2}\). For each such square, we insert two shallow-light trees [52] between the two sides of the square orthogonal to \(L\) and two points in \(L(U)\) at distance \(\varepsilon^{-1}\) from the square on either side; Fig. 5(bottom-right). The weight of each shallow-light tree is \(O(\varepsilon^{-1})\) [52], and so the combined weight of \(O(\varepsilon^{-1/2})\) shallow-light trees is \(O(\varepsilon^{-3/2})\). The grid \(G(U)\) together with the shallow-light trees forms the backbone for the bucket \(U\) in \(G_2\).

We add connector edges between \(a\) (resp., \(b\)) and the four corners of unit square of the grid \(G(U)\) that contains it. For any subsequent edge \(a'b' \in U\) that algorithm \(A_1\) inserts into \(G_1\), the backbone does not change, we only add connectors between \(a'\) (resp., \(b'\)) and the four corners of the unit square in \(G(U)\) that contains it. The weight of the four connectors is
$O(1)$ per point. Since area$(R) = \Theta(\varepsilon^{-3/2})$, then $R$ intersects at most $O(\varepsilon^{-3/2})$ unit squares of the quadtree at level $\ell$, and so the total weight of all connectors is $O(\varepsilon^{-3/2})$, as well.

**Stretch analysis.** Suppose algorithm $A_1$ inserts an edge $cd$ into $G_1$. As noted above, $cd$ lies in $\Theta(1)$ buckets; refer to Fig. 6. Suppose bucket $U$ contains $cd$; and in the partition of the rectangle $R = R(U)$, the endpoint $c$ (d) lies squares $R_c$ ($R_d$) of side length $\varepsilon^{-1/2}$, associated with shallow-light trees rooted at $r_c$ ($r_d$). Then $G_2$ contains a $\varepsilon$-path comprised of: (i) connectors from $c$ and $d$, resp., to the closest point in the grid $G(U)$; (ii) paths in $G(U)$ from the connectors to the boundary of squares $R_c$ and $R_d$, (iii) paths along the shallow-light trees to the roots $r_c,r_d \in L(U)$, and (iv) the line segment $r_cr_d$ in $G(U)$. The weight of each connector in (i) is at most $2\sqrt{2}$, which is bounded by $O(\varepsilon)||cd||$ since $||cd|| = \Theta(\varepsilon^{-1})$. The edges in (ii) and (iv) are parallel to $L$, hence they make an angle less than $\varepsilon^{1/2}$ with $cd$. Finally, consider the two subpaths in part (iii) in shallow-light trees: The line segment between the two endpoints of each such subpath makes an angle less than $\varepsilon^{1/2}$ with $L$, hence less than $2\varepsilon^{1/2}$ with $cd$; and the weight of a root-to-leaf path in a shallow-light tree is a $(1 + \varepsilon)$-approximation of the straight-line segment between its endpoints. Overall, the total weight of the $\varepsilon$-path described above is $(1 + O(\varepsilon)||cd||$, as required.

\[
\begin{array}{c}
\text{Figure 6 A } \varepsilon\text{-path in the Steiner spanner } G_2.
\end{array}
\]

For every point pair $a,b \in S_n$, the primary graph $G_1$ contains an $ab$-path $P = (p_0,\ldots,p_m)$ of length $||P|| \leq (1+\varepsilon)||ab||$, since $G_1$ is a $(1+\varepsilon)$-spanner. We have shown that for every edge $p_{i-1}p_i$ of $G_1$, the Steiner spanner $G_2$ contains a $p_{i-1}p_i$-path of weight $O(\varepsilon)||p_{i-1}p_i||$. The concatenation of these paths yields an $ab$-path in $G_2$, of weight $\sum_{i=1}^m (1 + O(\varepsilon))||p_{i-1}p_i|| = (1 + O(\varepsilon))||P|| = (1 + O(\varepsilon))(1 + \varepsilon)||ab|| = (1 + O(\varepsilon))||ab||$.

**Competitive Analysis.** Denote by $E_\ell$ the set of edges of $G_2$ added at level $\ell = 1,\ldots,2\log n$, and let $b_\ell$ be the number of nonempty buckets at level $\ell$. We have seen that for each nonempty bucket at level $\ell$, $E_\ell$ contains a subgraph of weight $O(\varepsilon^{-3/2}\text{diam}(S_n)/2^\ell)$; hence $||E_\ell|| \leq O(b_\ell \cdot \varepsilon^{-3/2}\text{diam}(S_n)/2^\ell)$.

Let $G^* = (S_n,E^*)$ be the a Euclidean Steiner $(1+\varepsilon)$-spanner for $S_n$ of minimum weight OPT. Consider a nonempty bucket $U$ associated with a line $L$ and a rectangle $R(U)$. Since $U$ is nonempty, there is an edge $ab \in U$ in $G_1$. Recall that $ab \in R$ and $\angle(\text{dir}(ab),\text{dir}(L)) \leq \varepsilon^{1/2}$. Since $G^*$ is a $(1+\varepsilon)$-spanner, it contains an $ab$-path $P_{ab}$ of weight at most $(1 + \varepsilon)||ab||$.

As noted in Section 3.1, $P_{ab}$ lies in the ellipse $B_{ab}$, and contains edges of weight at least $\frac{1}{2}||ab||$ that make an angle at most $\varepsilon^{1/2}$ with $ab$. All points in the ellipse $B_{ab}$ are at distance less than $\varepsilon^{1/2}$ from the line segment $ab$. The segment $ab$ lies in the $4\varepsilon^{-1} \times \varepsilon^{1/2}$ rectangle $R(U)$. Thus we have $P_{ab} \subset B_{ab} \subset 2R(U)$, and so $2R(U)$ contains edges of $G^*$ of weight $\frac{1}{2}||ab|| = \Omega(\varepsilon^{-1}\text{diam}(S_n)/2^\ell)$ whose directions are within $2\varepsilon^{1/2}$ from $L$; denote by $E^*(U) \subset E^*$ the set of these edges. By construction, each edge $e^*$ of $G^*$ lies in $E^*(U)$ for only $O(1)$ buckets. Indeed, there are $O(1)$ lines $L'$ with $\angle(\text{dir}(L),\text{dir}(L')) \leq \varepsilon^{1/2}$, and for each such direction $L'$, every point in $\mathbb{R}^2$ lies in $O(1)$ rectangles $2R(U')$ aligned with $L'$. We conclude that $\text{OPT} = ||G^*|| = \Omega(b_\ell \cdot \varepsilon^{-1}\text{diam}(S_n)/2^\ell)$. This implies $||E_\ell||/\text{OPT} \leq O(\varepsilon^{-1/2})$.
for \( \ell = 1, \ldots, 2 \log n \). Summation over all levels yields

\[
\frac{\text{ALG}}{\text{OPT}} = \frac{\sum_{\ell=1}^{\infty} \|E_\ell\|}{\text{OPT}} \leq \sum_{\ell=1}^{2 \log n} O(\varepsilon^{-1/2}) + O(1) = O(\varepsilon^{-1/2} \log n),
\]

as claimed. ▶

**Generalization to \( \mathbb{R}^d \).** Our algorithm and its analysis generalize to Euclidean \( d \)-space.

**Theorem 6.** For every \( \varepsilon > 0 \), an online algorithm can maintain, for a sequence of \( n \in \mathbb{N} \) points in \( \mathbb{R}^d \), a Euclidean Steiner \((1 + \varepsilon)\)-spanner of weight \( O(\varepsilon^{(1-d)/2} \log n) \cdot \text{OPT} \).

**Proof sketch.** The proof is analogous to that of Theorem 5, we highlight only the differences in the algorithm and its analysis. The bottleneck of the competitive analysis is the size of the unit grids \( G(U) \) which is \( \Theta(\varepsilon^{-(d+1)/2}) \) in \( \mathbb{R}^d \), which is contrasted with a path of weight \( \Omega(\varepsilon^{-1}) \) in \( \text{OPT} \).

Similarly to Section 3.1, we choose a homogeneous set \( D \) of \( \Theta(\varepsilon^{(1-d)/2}) \) directions (i.e., any direction is within angle \( \varepsilon^{1/2} \) from a direction in \( D \), and the angle between any two directions in \( D \) is at least \( \frac{\pi}{2e^{1/2}} \)). For each direction \( L \in D \), we construct a tiling of \( \mathbb{R}^d \) with congruent hyper-rectangles aligned with \( L \) of dimensions \( \varepsilon^{-1} \times \varepsilon^{-1/2} \times \ldots \varepsilon^{-1/2} \); and a covering of \( Q_0 \) after scaling up the hyperrectangles by a factor of 2.

We associate a bucket to each hyperrectangle \( R \) in the covering: an edge \( ab \) of \( G_1 \) is in bucket \( U \) if \( ab \subset R \) and \( \angle(\text{dir}(ab), \text{dir}(L)) \leq \varepsilon^{1/2} \). The construction ensures that every edge \( ab \in E_\ell \) is in at least one bucket, and at most \( O(1) \) buckets.

For each nonempty bucket \( U \), the grid \( G(U) \), shallow-light trees [52], and the connectors are analogous to the planar construction. However, the weight of the unit grid is \( \|G(U)\| = \Theta(\varepsilon^{-(d+1)/2}) \) in \( \mathbb{R}^d \). The stretch analysis carries over to higher dimensions. The lower bound for \( \text{OPT} \) is the same as in the plane: for each nonempty bucket at level \( \ell \), the rectangle \( 2R(U) \) contains edges of \( G_2 \) of weight \( \Omega(\varepsilon^{-1} \text{diam}(S_n)/2^\ell) \) with direction within \( 2\varepsilon^{1/2} \) from \( L \). This yields an upper bound \( \|E_\ell\|/\text{OPT} \leq O(\varepsilon^{(1-d)/2}) \) for levels \( \ell = 1, \ldots, 2 \log n \); and \( \frac{\text{ALG}}{\text{OPT}} \leq O(\varepsilon^{(1-d)/2} \log n) \) overall. ▶

### 4 Lower Bound with Steiner Points

Recall that when Steiner points are allowed, the algorithm may subdivide existing edges with Steiner points. It follows that in one-dimension, an online algorithm can maintain a Hamiltonian path on \( S_n \), which is the minimum \((1 + \varepsilon)\) spanner for all \( \varepsilon \geq 0 \). This property carries over to Euclidean Steiner \((1 + \varepsilon)\)-spanners (i.e., the case \( \varepsilon = 0 \)), where we need to maintain the complete straight-line graph on \( n \) points. However, we show that for \( \varepsilon > 0 \) in dimensions \( d \geq 2 \), the competitive ratio of an online algorithm with Steiner points must depend on \( n \).

**Theorem 7.** For every \( \varepsilon > 0 \), the competitive ratio of any online algorithm that maintains a Euclidean Steiner \((1 + \varepsilon)\)-spanner for a sequence of \( n \) points in \( \mathbb{R}^d \) is \( \Omega(f(n)) \) for some function \( f(n) \) such that \( \lim_{n \to \infty} f(n) = \infty \).

**Proof.** We describe and analyze an adversarial strategy for placing points in the plane in stages. In stage 1, the adversary places two points at \( s = (0,0) \) and \( t = (1,0) \), both on the \( x \)-axis. In subsequent stages, new points are arranged so that the optimum solution remains an \( x \)-monotone path of length at most \( 1 + \varepsilon \) at all times.

Let us denote by \( A_i \) the points placed in stage \( i \). At the end of stage \( i \), adversary constructs the point set \( A_{i+1} \) based on the current \((1 + \varepsilon)\)-spanner built by the algorithm.
ALG, and then placed the points in \( A_{i+1} \) in an arbitrary order. The objective is that in each stage, ALG has to add new edges of total weight at least \( 1/2 \). Since \( \text{OPT} \leq 1 + \varepsilon \) at all times, and ALG \( \geq \frac{1}{2}(i+1) \) after \( i \) stages, the competitive ratio goes to infinity.

We describe stage 2 in more detail; subsequent stages are similar; see Fig. 7. At the end of stage 1, our point set is \( A_1 = \{(s = (0,0), t = (1,0)\} \), the optimal spanner is a single edge of unit weight, and ALG has constructed a Euclidean Steiner \((1 + \varepsilon)\)-spanner \( G_1 \) for \( A_1 \). Let \( k_1 = \lceil \|G_1\| \rceil \). The adversary considers \( 2k_1 + 1 \) circular arcs between \( s \) and \( t \), each of weight at most \( 1 + \frac{\varepsilon}{2} \). The arcs define \( 2k_1 \) interior-disjoint bounded regions. Let \( R_1 \) be a region that minimizes the weight \( \|G_1 \cap R_1\| \), in particular, \( \|G_1 \cap R_1\| \leq \frac{1}{2\varepsilon} \|G_1\| \leq \frac{1}{2} \). In the interior of \( R_1 \), let \( \gamma_1 \) be another circular arc between \( s \) and \( t \), of weight \( \|\gamma_1\| \leq 1 + \frac{\varepsilon}{2} \); and let \( A_2 = \{t_1, \ldots, t_N\} \) be a set of points along \( \gamma_1 \), labeled in \( x \)-monotone increasing order with the following properties: (1) For every \( i = 1, \ldots, N-1 \), the ellipse \( R_i \) with foci \( t_i \) and \( t_{i+1} \), and great axis \((1 + \varepsilon)\|t_i, t_{i+1}\| \) lies entirely in \( R_1 \); and (2) the weight of the \( x \)-monotone path \( (t_1, t_2, \ldots, t_N) \) is at least 1.

![Figure 7](image-url)

In stage 2, the adversary presents the points in \( A_2 \) in an arbitrary order. By the end of stage 2, ALG augments \( G_1 \) to a Euclidean Steiner \((1 + \varepsilon)\)-spanner \( G_2 \) for \( A_1 \cup A_2 \). In particular, for every \( i = 1, \ldots, N-1 \), the graph \( G_2 \) contains a \( t_i, t_{i+1} \)-path of length at most \((1 + \varepsilon)\|t_i, t_{i+1}\| \), which lies in the ellipse \( E_i \), hence in the interior of the region \( R_i \). The part of the path between the vertical lines passing through \( t_i \) and \( t_{i+1} \) has weight at least \( \|t_i, t_{i+1}\| \). Since these parts are disjoint, the total weight all \( N-1 \) paths is at least \( \sum_{i=1}^{N-1} \|t_i, t_{i+1}\| \geq 1 \). Consequently, \( \|G_1 \cap R_i\| \geq 1 \). Since we had \( \|G_1 \cap R_i\| \leq \frac{1}{2} \), ALG must have added new edges of weight at least \( \frac{1}{2} \) in stage 2, as claimed.

In phase \( i + 1 \), in general, let \( k_i = \lceil \|G_i\| \rceil \). Label the points in the current point set \( S = \bigcup_{j=1}^{i} A_j \) by \( s_0, \ldots, s_n \) in \( x \)-monotone order, and assume that the \( x \)-monotone path spanned by \( S \) has weight \( \text{OPT} = 1 + (1 - \frac{1}{2\varepsilon}) \varepsilon \). For all segments \( s_j, s_{j+1} \), we consider \( 2k_i + 1 \) \( x \)-monotone circular arcs such that the total weight of any concatenation of the circular arcs from \( s = s_0 \) to \( t = s_n \) is at most \( 1 + (1 - \frac{1}{2\varepsilon}) \varepsilon \). For each segment \( s_j, s_{j+1} \), we choose one of \( 2k_i \) regions that has a minimum-weight intersection with \( G_i \), and let \( R_i \) be the union of these regions. Note that \( \|G_i \cap R_i\| \leq \frac{1}{2\varepsilon} \|G_i\| \leq \frac{1}{2} \). Let \( \gamma_i \) be an \( s \)-path \( \gamma_i \) that connects the points \( s_0, \ldots, s_n \) via circular arcs in the region \( R_i \), and has weight at most \( 1 + (1 - \frac{1}{2\varepsilon}) \varepsilon \). Now the adversary can choose a finite point set \( A_{i+1} = \{t_1, \ldots, t_N\} \) along \( \gamma_i \) with properties (1)-(2) above. This completes the description of the adversarial strategy.

Similarly to stage 2, when ALG augments \( G_i \) to a Euclidean Steiner \((1 + \varepsilon)\)-spanner \( G_{i+1} \) for \( \bigcup_{j=1}^{i+1} A_j \), he must add new edges of weight at least \( \frac{1}{2} \) in the region \( R_i \). It follows that the competitive ratio for any online algorithm goes to infinity as \( n \) goes to infinity.

5 Lower Bounds for Spanners in \( \mathbb{R}^d \) under the \( L_1 \) Norm

In this section, we study the online \((1 + \varepsilon)\)-spanners for points in \( \mathbb{R}^d \), under the \( L_1 \) norm. Here the distance between any pair of points is the sum of absolute difference between the
coordinates in all dimensions. For instance in $\mathbb{R}^2$, the distance between points $p = (p_1, p_2)$ and $q = (q_1, q_2)$ is the rectilinear distance between them, i.e., $|p_1 - q_1| + |p_2 - q_2|$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{image}
\caption{(a) The point sets $S_1$ and $S_2$ in two opposite quadrants. (b) The entire point set $(S_1 \cup \hat{S}_1) \cup (S_2 \cup \hat{S}_2)$; (c) A Manhattan network which is $(1 + \varepsilon)$-spanner consisting of two binary trees $T_1$ and $T_2$ and two staircase paths $P_1$ and $P_2$. The green path is an $xy$-monotone path between point pairs $s_p$ and $s_q$.}
\end{figure}

**Construction.** First, we describe the construction for points in $\mathbb{R}^2$. We use the following adversarial strategy to build the construction.

Let $k = \lceil \log \varepsilon^{-1} \rceil$. The adversary places $2^k$ collinear points in the positive quadrant, and $2^k$ points in the negative quadrant such that the points are placed in a quadrant maintain uniform spacing between consecutive points. Let $S_1 = \{s_i : i = 0, \ldots, 2^k - 1\}$, where $s_i = (i, 2^k - i)$. Then, $S_2$ is a reflected copy of $S_1$ about the origin. See Fig. 8(a) for an illustration. Next, the adversary introduces a set of additional points $\hat{S}_1 \cup \hat{S}_2$ in order to reduce the weight of an optimum solution. Consider the set $S_1$. We consider the binary partition of $\{1, \ldots, 2^k\}$ into intervals, associated with a binary tree. At level 0, the root corresponds to the interval $[1, 2^k]$. Then, at each level $j$ we have the intervals $[i \cdot 2^{k-j} + 1, (i + 1) \cdot 2^{k-j}]$, for $i = 0, \ldots, 2^j$. For each such interval, the adversary considers the bounding box of the points lying in this interval, and adds the lower-left corner into $\hat{S}_1$. Then, $\hat{S}_2$ is a reflected copy of $\hat{S}_1$ about the origin. The entire point set is $(S_1 \cup \hat{S}_1) \cup (S_2 \cup \hat{S}_2)$; see Fig. 8(b)

**Competitive Ratio.** For the point set $(S_1 \cup \hat{S}_1) \cup (S_2 \cup \hat{S}_2)$, we describe a Manhattan network which is a $(1 + \varepsilon)$-spanner. The network is comprised of the paths $P_1$ and $P_2$ and the two binary trees $T_1$ and $T_2$ on the point set $(S_1 \cup \hat{S}_1)$ and $(S_2 \cup \hat{S}_2)$, respectively; see Fig. 8(c). First, we argue that why it is a Manhattan network. In order to show this, we need to argue that for each pair of points in $(S_1 \cup \hat{S}_1) \cup (S_2 \cup \hat{S}_2)$, there is a rectilinear $xy$-monotone path of optimum length comprised of horizontal and vertical segments.

Consider a pair of points $s_p, s_q \in (S_1 \cup \hat{S}_1)$. Without loss of generality, assume that $y(s_p) > y(s_q)$. If, $s_q$ is a descendent of $s_p$ in $T_1$, then we are done since there is an unique $xy$-monotone path between them in $T_1$. Otherwise, we take the left path form $s_p$ to the leaf point in $S_1$ and the right path from $s_q$ to the leaf point in $S_1$, and the sub-path from $P_1$ between these leaf points. These paths together from a $xy$-monotone rectilinear path; see Fig. 8. Moreover, if both $s_p$ and $s_q$ are leaves, then we have a $xy$-monotone path between them which is a subpath of $P_1$. The same arguments holds for any point pair in $(S_2 \cup \hat{S}_2)$.
for which we have the complete binary tree $T_2$ and the path $P_2$. Now, what is left two show that there is an $xy$-monotone path between any pair of points $s_p, s_q$, where $s_p \in (S_1 \cup \hat{S}_1)$ and $s_q \in (S_2 \cup \hat{S}_2)$. For this, we consider the unique $xy$-monotone path from $s_p$ to the root in $T_1$ and $xy$-monotone path from $s_q$ to the root in $T_2$. We take the union of them, which is a $xy$-monotone rectilinear path between $s_p$ and $s_q$.

The weight of the Manhattan network is the weight of the two trees $T_1$ and $T_2$ and the two paths $P_1$ and $P_2$. The distance between the root and $S_1$ is $\min\{|i| + |2^k - i| : i = 0, \ldots, 2^k - 1\} = 2^k$. At each level $j$, we construct $2^j$ segments. Each segment at level $j$ is of weight $2^{k-j}$. Summation over all levels yields $O(k 2^k) = O(\varepsilon^{-1} \log \varepsilon^{-1})$.

For the point set $S_1 \cup S_2$, every $(1 + \varepsilon)$-spanner contains a complete bipartite graph between $S_1$ and $S_2$. Indeed, the $L_1$-distance between any two points in $S_1$ (resp., $S_2$) is 2 or more, and the $L_1$ distance between any points in $S_1$ and $S_2$ is exactly $\text{dist}(S_1, S_2) = 2(2^k - 1)$. Hence $(1 + \varepsilon)\|s_p s_q\| \leq (1 + 2^{-k}) \cdot 2(2^k - 1) < \text{dist}(S_1, S_2) + 2$. However, the weight of any $s_p s_q$-path via any third point in $S_1 \cup S_2$ would be at least $\text{dist}(S_1, S_2) + 2$. Therefore, $\text{ALG} \geq \Omega(\|S_1 \cdot |S_2| \cdot \text{dist}(S_1, S_2)) = \Omega(\varepsilon^{-1})$ for any online algorithm. Contrasted with the upper bound $\text{OPT} \leq O(\varepsilon^{-1} \log \varepsilon^{-1})$ for $(S_1 \cup \hat{S}_1) \cup (S_2 \cup \hat{S}_2)$, the competitive ratio is $\text{ALG}/\text{OPT} \geq \Omega(\varepsilon^{-2} / \log \varepsilon^{-1})$. We can conclude the following theorem.

**Theorem 8.** For every $\varepsilon > 0$, the competitive ratio of any online algorithm for $(1 + \varepsilon)$-spanners in $\mathbb{R}^2$ under the $L_1$ norm is $\Omega(\varepsilon^{-2} / \log \varepsilon^{-1})$.

**Construction in $\mathbb{R}^d$.** For a given $\varepsilon > 0$, let $k = \lceil \log \varepsilon^{-1} \rceil$. First, the adversary introduces point sets $S_1$ and $S_2$ on two opposite faces of a cross polytope: Let $S_1$ be the set of all points with non-negative integer coordinates in the hyperplane $H_1 : \sum_{i=1}^d x_i = 2^k - 2$, and let $S_2$ be the reflected image of $S_1$ in the origin. Note that $|S_1| = |S_2| = \Theta(2^{k(d-1)}) = \Theta(\varepsilon^{1-d})$, and the bounding box of $S_1$ is $Q = [0, 2^k - 2]^d$. Next, the adversary introduces a set of additional points $\hat{S}_1 \cup \hat{S}_2$ in order to reduce to weight of an optimum solution. Consider the quadtree $Q_1$ for $S_1$, which partitions $Q$ into congruent cubes and recurses on all nonempty subcubes. Let $\tilde{S}_1$ be the set of all vertices of the cubes in the quadtree; and let $\tilde{S}_2$ be the reflected image of $\tilde{S}_1$ in the origin. By construction, all points in $(S_1 \cup \tilde{S}_1) \cup (S_2 \cup \tilde{S}_2)$ have integer coordinates.

**Competitive Ratio.** Similarly to the planar construction, every $(1 + \varepsilon)$-spanner will contain a complete bipartite graph between $S_1$ and $S_2$. As $\|s_p s_q\| \geq 2^k = \Omega(\varepsilon^{-1})$ for all $s_p \in S_1$ and $s_q \in S_2$, then $\text{ALG} \geq \Omega(\varepsilon^{-1}|S_1| \cdot |S_2|) \geq \Omega(\varepsilon^{-1-2d})$.

Next, we give a lower bound for $\text{OPT}$ for the entire point set $(S_1 \cup S_2) \cup (\tilde{S}_1 \cup \tilde{S}_2)$. Let $T_1$ is the graph formed by all edges of the quadtree $Q_1$, and $T_2$ is the reflected image of $T_1$. Since the depth of $Q_1$ is $k$, each nonleaf node has 3 or more children, and the edge lengths of the cubes in $Q_1$ decrease by factors of 2. Consequently, the weight $\|T_1\|$ is dominated by the weight of level $k$, which contains $\Theta(|S_1|) = \Theta(2^{k(d-1)}) = \Theta(\varepsilon^{1-d})$ unit cubes, hence the total weight of (unit-length) edges is also $\Theta(\varepsilon^{1-d})$. Overall, we have $\|T_1\| = \Theta(\varepsilon^{1-d})$.

We show that $T_1 \cup T_2$ is a Manhattan network for $(S_1 \cup S_2) \cup (\tilde{S}_1 \cup \tilde{S}_2)$. Consider first $T_1$. At each level $\ell = 0, \ldots, k$ the quadtree $Q_{\ell}$, contains all cubes that intersect the hyperplane $\sum_{i=1}^d x_i = 2^k - 2$. These cubes have the same size. For any two vertices of two cubes on level $\ell$ of $Q_{\ell}$, say $s_a \in Q_{\ell}$ and $s_b \in Q_{\ell}$, let $s'_a$ and $s'_b$ be the points in $H_1$ such that $s'_a s'_a$ and $s'_b s'_b$ are parallel to the $x_\ell$-axis. The line segment $s'_a s'_b$ lies in $H_1$, and is covered by quadtree cubes at level $\ell$. The orthogonal projections of these cubes to each coordinate axis is comprises consecutive intervals, hence we can find a Manhattan path between $s_a$ and $s_b$ along the edges of these cubes. Next assume that $s_a \in Q_a$ and $s_b \in Q_b$ are vertices of two cubes at different
levels of $Q_1$. If $Q_a$ and $Q_b$ are in ancestor-descendant relation, we can find a Manhattan path between $s_a$ and $s_b$ by tracing the edges created by the recursive subdivision. Otherwise, $Q_a$ and $Q_b$ are interior-disjoint. Assume that the side-length of $Q_a$ is less than that of $Q_b$; and let $Q'_b \subset Q_b$ be a quadtree cube on the same level as $Q_a$, and closest to $Q_a$. Then we can find a Manhattan path from $s_a$ to $s_b$ as a concatenation of two Manhattan paths via a vertex of $Q'_b$. Finally, for two vertices in $T_1$ and $T_2$, resp., we obtain a Manhattan path by concatenating two Manhattan paths via the origin, which is a vertex of both $T_1$ and $T_2$.

Since $T_1 \cup T_2$ is a Manhattan network for $(S_1 \cup S_2) \cup (\hat{S}_1 \cup \hat{S}_2)$, then $\text{OPT} \leq \|T_1\| + \|T_2\| = \Theta(\varepsilon^{1-d})$. Combined with $\text{ALG} \geq \Omega(\varepsilon^{1-2d})$, this yields a lower bound of $\Omega(\varepsilon^{1-2d}/\varepsilon^{1-d}) = \Omega(\varepsilon^{-d})$ for the competitive ratio of any online algorithm in $\mathbb{R}^d$ under the $L_1$ norm. The following theorem summarizes our results in dimensions $d \geq 3$.

**Theorem 9.** Let $d \geq 3$. For every $\varepsilon > 0$, the competitive ratio of any online algorithm for $(1 + \varepsilon)$-spanners in $\mathbb{R}^d$ under the $L_1$ norm is $\Omega(\varepsilon^{-d})$.

**Remark 10.** Recall from the discussion in Section 1 that there are point sequences for which the insertion of a point may reduce the weight of the spanner (Fig. 1). So, the weight of the optimum solution need not increase monotonically in the number of points. In this section, this phenomenon leads to a significant improvement over the weight of an optimum spanner under $L_1$ norm, and helped to design an improved lower bound for the competitive ratio. We do not know whether this phenomenon can be exploited to produce improved lower bounds under the $L_2$-norm.

## 6 Conclusions

We have studied online spanners for sequences of points in $\mathbb{R}^d$, in fixed dimensions $d \geq 1$, under $L_2$ and $L_1$ norms. We established a tight bound of $\Theta(\varepsilon^{-1} \log n/ \log \varepsilon^{-1})$ for the competitive ratio of any online $(1 + \varepsilon)$-spanner algorithms on a real line (Theorem 3). However it remains an open problem to close the gap between the lower and upper bounds in $\mathbb{R}^d$, for $d \geq 2$. Under the $L_2$ norm, previously known algorithms achieve competitive ratio $O(\varepsilon^{-(d+1)} \log n)$ (Theorem 4). The best lower bound we are aware of holds for $d = 1$. It is unclear whether the lower bound can be improved to $\varepsilon^{-d} \log n$ for $d \geq 2$.

Next, we have showed that, if an online algorithm is allowed to use Steiner points, it can achieve a substantially better competitive ratio in terms of $\varepsilon$, namely $O(\varepsilon^{(1-d)/2} \log n)$, for a sequence of $n$ points in $\mathbb{R}^d$ and any constant $d \geq 2$, under the $L_2$ norm (Theorem 6). As a counterpart, we proved that any online spanner algorithm for a sequence of $n$ points in $\mathbb{R}^d$ under $L_2$ norm has competitive ratio $\Omega(f(n))$, where $\lim_{n \to \infty} f(n) = \infty$ (Theorem 7). It remains an open problem whether the competitive ratio depends on $\varepsilon$ for Euclidean Steiner spanners. Another open problem is whether the factor $\log n$ in the upper bounds can be reduced, e.g., to $\log n/ \log \log n$; similar to the work by Alon and Azar [2] who established such a lower bound for Euclidean minimum Steiner trees (EMST) for $n$ points in $\mathbb{R}^2$.

We have established a lower bound $\Omega(\varepsilon^{-d})$ for the competitive ratio under the $L_1$-norm in $\mathbb{R}^d$. It is unclear whether it can be improved by a log $n$ factor in dimensions $d \geq 2$. Designing online algorithms that match these bounds under the $L_1$ norm is left for future research.

In online spanner algorithms, the decisions are irrevocable, which means that once an edge is added to the spanner by an online algorithm, it can never be deleted. However, if some of the decisions are reversible, better bounds may be possible. This model is commonly known as online algorithms with recourse [33, 39, 45]. In 1-dimension, for instance, an optimum spanner is just a monotone path connecting the points in linear order, and any
online algorithm that is allowed to remove at least one edge at per iteration can maintain such a path. In higher dimensions, however, it is unclear whether a $O(1)$-approximation of the minimum-weight $(1 + \varepsilon)$-spanner can be maintained with $O(\varepsilon^{-d+1})$ recourse.

References

1. Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. A general approach to online network optimization problems. *ACM Transactions on Algorithms (TALG)*, 2(4):640–660, 2006.
2. Noga Alon and Yossi Azar. On-line Steiner trees in the Euclidean plane. *Discrete & Computational Geometry*, 10:113–121, 1993.
3. Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9(1):81–100, 1993.
4. Sunil Arya, David M Mount, and Michiel Smid. Randomized and deterministic algorithms for geometric spanners of small diameter. In *Proc. 35th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 703–712, 1994.
5. Sunil Arya and Michiel Smid. Efficient construction of a bounded-degree spanner with low weight. *Algorithmica*, 17(1):33–54, 1997.
6. Baruch Awerbuch, Yossi Azar, and Yair Bartal. On-line generalized Steiner problem. *Theoretical Computer Science*, 324(2-3):313–324, 2004.
7. Baruch Awerbuch, Alan E. Baratz, and David Peleg. Cost-sensitive analysis of communication protocols. In *Proc. 9th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 177–187, 1990.
8. Surender Baswana, Sumeet Khurana, and Soumojit Sarkar. Fully dynamic randomized algorithms for graph spanners. *ACM Trans. Algorithms*, 8(4):35:1–35:51, 2012.
9. Thiago Bergamaschi, Monika Henzinger, Maximilian Probst Gutenberg, Virginia Vassilevska Williams, and Nicole Wein. New techniques and fine-grained hardness for dynamic near-additive spanners. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1836–1855, 2021.
10. Piotr Berman and Chris Coulston. On-line algorithms for steiner tree problems. In *Proc. 29th ACM Symposium on Theory of Computing (STOC)*, pages 344–353, 1997.
11. Aaron Bernstein, Sebastian Forster, and Monika Henzinger. A deamortization approach for dynamic spanner and dynamic maximal matching. In *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1899–1918, 2019.
12. Sujoy Bhore and Csaba D. Tóth. Light Euclidean Steiner spanners in the plane. In *Proc. 37th International Symposium on Computational Geometry (SoCG)*, volume 189 of LIPIcs, pages 31:1–17. Schloss Dagstuhl, 2021.
13. Sujoy Bhore and Csaba D. Tóth. On Euclidean Steiner $(1+\varepsilon)$-spanners. In *Proc. 38th Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 187 of LIPIcs, pages 13:1–13:16. Schloss Dagstuhl, 2021.
14. Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.
15. Prosenjit Bose, Joachim Gudmundsson, and Pat Morin. Ordered theta graphs. *Computational Geometry*, 28(1):11–18, 2004.
16. Prosenjit Bose and Michiel H. M. Smid. On plane geometric spanners: A survey and open problems. *Comput. Geom.*, 46(7):818–830, 2013.
17. Paul B. Callahan. Optimal parallel all-nearest-neighbors using the well-separated pair decomposition. In *Proc. 34th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 332–340, 1993.
18. Paul B. Callahan and S. Rao Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In Vijaya Ramachandran, editor, *Proc. 4th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 291–300, 1993.
19 Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to \( k \)-nearest-neighbors and \( n \)-body potential fields. *J. ACM*, 42(1):67–90, 1995.

20 Paz Carmi and Lilach Chaitman-Yerushalmi. Minimum weight Euclidean \( t \)-spanner is NP-hard. *Journal of Discrete Algorithms*, 22:30–42, 2013.

21 Timothy M. Chan, Sariel Har-Peled, and Mitchell Jones. On locality-sensitive orderings and their applications. *SIAM J. Comput.*, 49(3):583–600, 2020.

22 L. Paul Chew. There is a planar graph almost as good as the complete graph. In *Proc. 2nd Symposium on Computational Geometry (SoCG)*, pages 169–177. ACM Press, 1986.

23 L. Paul Chew. There are planar graphs almost as good as the complete graph. *J. Comput. Syst. Sci.*, 39(2):205–219, 1989.

24 Kenneth L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. 19th ACM Symposium on Theory of Computing (STOC)*, pages 56–65, 1987.

25 Gautam Das, Paul Heffernan, and Giri Narasimhan. Optimally sparse spanners in 3-dimensional Euclidean space. In *Proc. 9th Symposium on Computational Geometry (SoCG)*, pages 53–62. ACM Press, 1993.

26 Gautam Das, Giri Narasimhan, and Jeffrey S. Salowe. A new way to weigh malnourished Euclidean graphs. In *Proc. 6th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 215–222. 1995.

27 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3 edition, 2008.

28 Michael Elkin and Shay Solomon. Steiner shallow-light trees are exponentially lighter than spanning ones. *SIAM Journal on Computing*, 44(4):996–1025, 2015.

29 John Fischer and Sariel Har-Peled. Dynamic well-separated pair decomposition made easy. In *Proc. 17th Canadian Conference on Computational Geometry (CCCG)*, pages 235–238, 2005.

30 Jie Gao, Leonidas J. Guibas, and An Nguyen. Deformable spanners and applications. *Comput. Geom.*, 35(1-2):2–19, 2006.

31 Lee-Ad Gottlieb, Aryeh Kontorovich, and Robert Krauthgamer. Efficient regression in metric spaces via approximate Lipschitz extension. *IEEE Transactions on Information Theory*, 63(8):4838–4849, 2017.

32 Lee-Ad Gottlieb and Liam Roditty. An optimal dynamic spanner for doubling metric spaces. In *Proc. 16th European Symposium on Algorithms (ESA)*, volume 5193 of *LNCS*, pages 478–489. Springer, 2008.

33 Albert Gu, Anupam Gupta, and Amit Kumar. The power of deferral: Maintaining a constant-competitive Steiner tree online. *SIAM Journal on Computing*, 45(1):1–28, 2016.

34 Joachim Gudmundsson and Christian Knauer. Dilation and detours in geometric networks. In Teofilo F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*, volume 2. Chapman and Hall/CRC, 2nd edition, 2018.

35 Joachim Gudmundsson, Christos Levcopoulos, and Giri Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM J. Comput.*, 31(5):1479–1500, 2002.

36 Joachim Gudmundsson, Christos Levcopoulos, Giri Narasimhan, and Michiel Smid. Approximate distance oracles for geometric spanners. *ACM Transactions on Algorithms (TALG)*, 4(1):1–34, 2008.

37 Mohammad Taghi Hajiaghayi, Vahid Liaghaf, and Debmalya Panigrahi. Online node-weighted Steiner forest and extensions via disk paintings. In *Proc. 54th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 558–567, 2013.

38 Sariel Har-Peled. *Geometric Approximation Algorithms*, volume 173 of *Mathematical Surveys and Monographs*. AMS, Providence, RI, 2011.

39 Makoto Imase and Bernard M. Waxman. Dynamic Steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

40 J. Mark Keil. Approximating the complete Euclidean graph. In *Proc. 1st Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 318 of *LNCS*, pages 208–213. Springer, 1988.
22 Online Euclidean Spanners

41 Samir Khuller, Balaji Raghavachari, and Neal E. Young. Balancing minimum spanning and shortest path trees. In Proc. 4th ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 243–250, 1993.

42 Hung Le and Shay Solomon. Truly optimal Euclidean spanners. In Proc. 60th IEEE Symposium on Foundations of Computer Science (FOCS), pages 1078–1100, 2019.

43 Hung Le and Shay Solomon. Light Euclidean spanners with Steiner points. In Proc. 28th European Symposium on Algorithms (ESA), volume 173 of LIPIcs, pages 67:1–67:22. Schloss Dagstuhl, 2020.

44 Hung Le and Shay Solomon. A unified and fine-grained approach for light spanners. CoRR, abs/2008.10582, 2020. arXiv:2008.10582.

45 Nicole Megow, Martin Skutella, José Verschae, and Andreas Wiese. The power of recourse for online MST and TSP. SIAM Journal on Computing, 45(3):859–880, 2016.

46 Joseph Naor, Debmalya Panigrahi, and Mohit Singh. Online node-weighted Steiner tree and related problems. In Proc. 52nd IEEE Symposium on Foundations of Computer Science (FOCS), pages 210–219, 2011.

47 Giri Narasimhan and Michiel Smid. Geometric Spanner Networks. Cambridge University Press, 2007.

48 Satish B. Rao and Warren D. Smith. Approximating geometrical graphs via “spanners” and “banyans”. In Proc. 13th ACM Symposium on Theory of Computing (STOC), pages 540–550, 1998.

49 Liam Roditty. Fully dynamic geometric spanners. Algorithmica, 62(3-4):1073–1087, 2012.

50 Christian Schindelhauer, Klaus Volbert, and Martin Ziegler. Geometric spanners with applications in wireless networks. Comput. Geom., 36(3):197–214, 2007.

51 Michiel H. M. Smid. The well-separated pair decomposition and its applications. In Teofilo F. Gonzalez, editor, Handbook of Approximation Algorithms and Metaheuristics, volume 2. Chapman and Hall/CRC, 2nd edition, 2018.

52 Shay Solomon. Euclidean Steiner shallow-light trees. J. Comput. Geom., 6(2):113–139, 2015.

53 Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. SIAM J. Comput., 11(4):721–736, 1982.