CSE520 Computational Geometry
Lecture 21
Geometric Approximation Algorithms I

Antoine Vigneron

Ulsan National Institute of Science and Technology

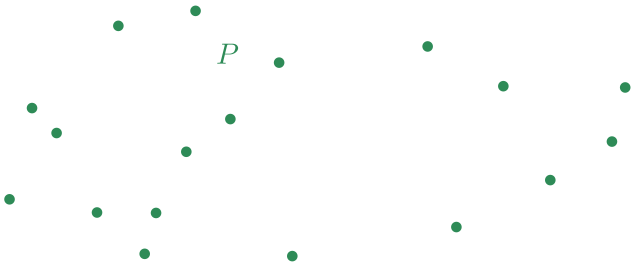
June 15, 2020

Outline

- This lecture is an introduction to geometric approximation algorithms through an example: computing the diameter of a point set.
- Our algorithm will be based on *rounding* to a grid.

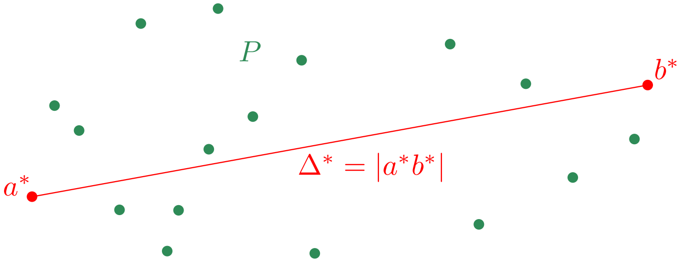
- References:
- Sariel Har Peled's book.
- Paper by T. Chan, *Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus*, Section 2.

The Diameter Problem



- Input: a set P of n points in $\mathbb{R}^d$.

The Diameter Problem



- Input: a set P of n points in $\mathbb{R}^d$.
- Output: the maximum distance Δ^* between any two points of P.
- Δ^* is called the *diameter* of P.

Brute Force Algorithm

- Computing the distance between two points:
- If $a = (a_1, a_2, \ldots, a_d)$ and $b = (b_1, b_2, \ldots, b_d)$, then

$$|ab| = \sqrt{(b_1 - a_1)^2 + (b_2 - a_2)^2 + \cdots + (b_d - a_d)^2}.$$

- It takes time $O(d)$.
- Here we assume that d is constant: $d = O(1)$.
  - We are in *fixed dimension*.
- Then we can compute $|ab|$ in $O(1)$ time.

- Computing the diameter by brute force:
- Check all pairs in P^2 and keep the maximum distance.
- It takes $O(n^2)$ time.

Approximation Algorithms

Definition (c-factor approximation)

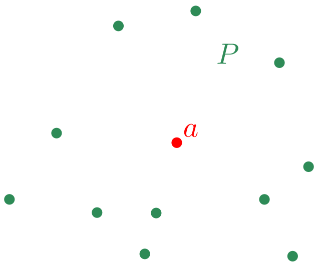
We say that Δ is a c-factor approximation to Δ^* if $\Delta \leqslant \Delta^* \leqslant c\Delta$.

Definition (c-approximation algorithm)

A c-approximation algorithm for a maximization problem is an algorithm that computes a c-factor approximation of the optimum in polynomial time.

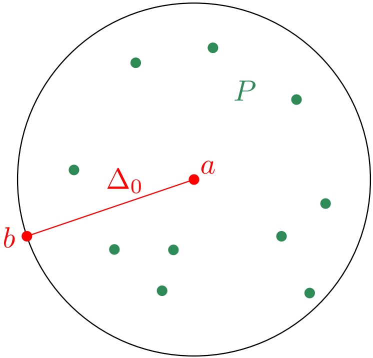
- We will give a 2-approximation algorithm for the diameter problem.
- That is, we find Δ_0 such that $\Delta_0 \leqslant \Delta^* \leqslant 2\Delta_0$.
- $O(n)$ time, very simple.
- Best known exact algorithms are slower.

2-Approximation Algorithm



- Pick a point $a \in P$.

2-Approximation Algorithm



- Pick a point $a \in P$.
- Find b such that $|ab|$ is maximum.
- $\Delta_0 = |ab|$.

Analysis

- Pick any point $a \in P$.
  - $O(1)$ time.
- Find $b \in P$ such that $|ab|$ is maximum.
  - Go through the list of points in P and keep the maximum distance to a.
  - It takes $O(n)$ time.
- Conclusion: This algorithm runs in $O(n)$ time.

Proof of Correctness

- We need to prove that Δ_0 is a 2-factor approximation of Δ^*.
- It means $\Delta_0 \leqslant \Delta^* \leqslant 2\Delta_0$.

- Since $(a, b) \in P^2$, and Δ^* is the diameter of P, we have $\Delta_0 = |ab| \leqslant \Delta^*$.
- We also need to prove that $\Delta^* \leqslant 2\Delta_0$.
  - Let a^* and b^* denote two points such that $|a^* b^*| = \Delta^*$.
  - As b is the farthest point from a, we have $|aa^*| \leqslant |ab|$ and $|ab^*| \leqslant |ab|$.
  - By the triangle inequality

$$\begin{aligned}\Delta^* &= |a^* b^*| \\ &\leqslant |a^* a| + |ab^*| \\ &\leqslant |ab| + |ab| \\ &= 2\Delta_0.\end{aligned}$$

Concluding Remark

- The running time is optimal.

- If we do not assume $d = O(1)$, then this algorithm is still correct.
- But the running time has to be written $O(dn)$.
- It is still optimal as we need $\Theta(nd)$ time to read the input.

$(1+\varepsilon)$-Approximation Algorithms

- We just found a 2-approximation algorithm.
- We would like to obtain a better approximation.

- Let $\varepsilon > 0$ be a real number.
- In this lecture, we assume $\varepsilon < 1$.
- ε should be thought of as being small, say $\varepsilon = 0.1$ or $\varepsilon = 0.01$.

- We want to design a $(1+\varepsilon)$-approximation algorithm.
- The result will be Δ such that $\Delta \leqslant \Delta^* \leqslant (1+\varepsilon)\Delta$.
- In other words, the *relative error* we allow is ε.
- So $\varepsilon = 0.01$ means a 1% error.

Analysis

- How to analyze a $(1+\varepsilon)$-approximation algorithm?
- The running time will be expressed as a function of n and ε using $O(\cdot)$ notation.
- For instance, the last algorithm in this lecture runs in time

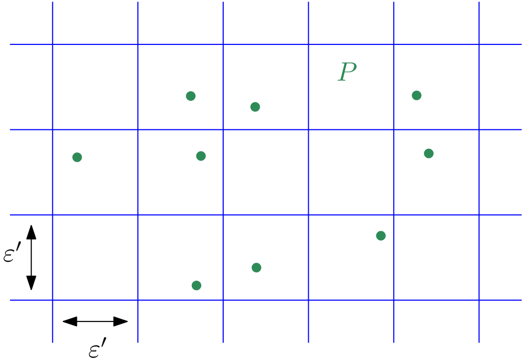
$$O\left(n + (1/\varepsilon)^{2d}\right)$$

- Since $d = 0(1)$, it is polynomial in n and $1/\varepsilon$.
- Such algorithms are called *FPTAS*: Fully Polynomial Time Approximation Schemes.
- The running time is linear in n, but exponential in d.
- So this algorithm is only useful in low dimension d.

Approach

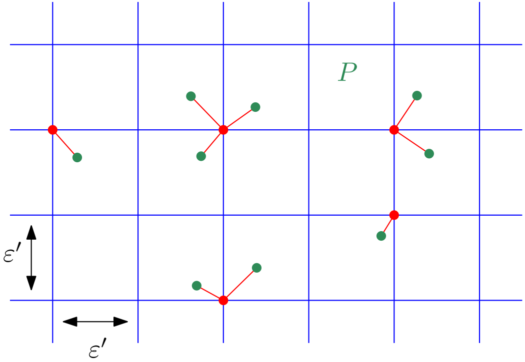
- Start with a set P of n points.
- Transform it into a set P' such that:
  - Its cardinality $|P'|$ is small.
  - The diameter Δ' of P' is a good approximation of Δ^*.
- Then find Δ' by brute force

Rounding to a Grid



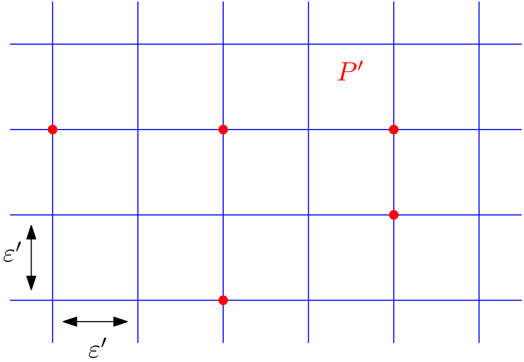
- Consider a regular grid over $\mathbb{R}^d$.
- The side length of the grid is ε', to be specified later.
- Intuition: we will choose $\varepsilon' \approx \varepsilon\Delta^*$, which is the error we allow.

Rounding to a Grid



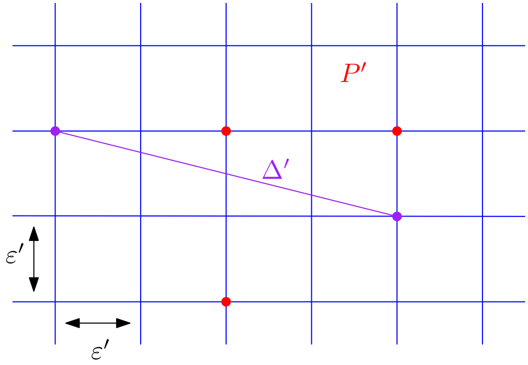
- Replace each point of P with the nearest grid point.
- This operation is called *rounding*.

Rounding to a Grid



- The grid points we obtain form the set P'.

Rounding to a Grid



- Compute the diameter Δ' of P' by brute force.

Intuition

- P' is a d-dimensional point set with diameter Δ'.
- The points are on a grid with side length ε'; we will choose it such that $\varepsilon' = \Theta(\Delta'\varepsilon)$.
- So in the worst case, there are about as many points in P' as in a $(1/\varepsilon) \times (1/\varepsilon) \cdots \times (1/\varepsilon)$ grid in $\mathbb{R}^d$.
- There are $O((1/\varepsilon)^d)$ such points.
- We can compute Δ' by brute force in time $O((1/\varepsilon)^{2d})$.

How to Perform Rounding?

- The grid points have coordinates $(k_1\varepsilon', k_2\varepsilon', \dots k_d\varepsilon')$ where each k_i is an integer.
- Let $p = (p_1, p_2, \dots p_d) \in P$.
- How can we find the closest grid point p'?
- We need to find the closest integer k_i to p_i/ε'.
- It is given by the formula

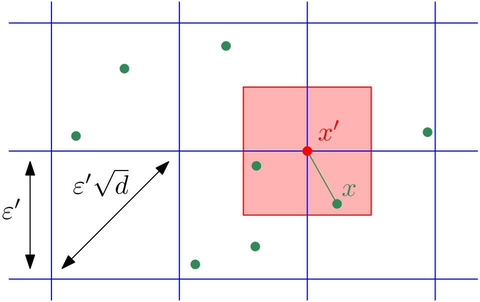
$$k_i = \left\lfloor \frac{p_i}{\varepsilon'} + \frac{1}{2} \right\rfloor .$$

- p is rounded to $p' = (k_1\varepsilon', k_2\varepsilon', \dots k_d\varepsilon')$.
- It takes $O(d) = O(1)$ time.

Rounding Error

Property

Let $x \in \mathbb{R}^d$, and let x' be the closest grid point to x. Then $|xx'| \leqslant \varepsilon' \sqrt{d}/2$.



- x is in a hypercube centered at x' with side length ε'.
- This hypercube diagonal has length $\varepsilon' \sqrt{d}$.

Approximation Factor

- Let a' and b' be the closest grid points to a^* and b^*, respectively.
- Then from previous slide, $|a'a^*| \leqslant \varepsilon'\sqrt{d}/2$ and $|b'b^*| \leqslant \varepsilon'\sqrt{d}/2$.
- By the triangle inequality,

$$\begin{aligned}\Delta^* &= |a^*b^*| \\ &\leqslant |a^*a'| + |a'b'| + |b'b^*| \\ &\leqslant |a'b'| + \varepsilon'\sqrt{d} \\ &\leqslant \Delta' + \varepsilon'\sqrt{d}.\end{aligned}$$

Approximation Factor

- Let c' and d' be two points of P' such that $|c'd'| = \Delta'$.
- Let c and d be points of P that have been rounded to c' and d', respectively.
- Then $|cc'| \leqslant \varepsilon'\sqrt{d}/2$ and $|dd'| \leqslant \varepsilon'\sqrt{d}/2$.
- It follows that

$$\begin{aligned}
\Delta' &= |c'd'| \\
&\leqslant |c'c| + |cd| + |dd'| && \text{by the triangle inequality} \\
&\leqslant |cd| + \varepsilon'\sqrt{d} \\
&\leqslant \Delta^* + \varepsilon'\sqrt{d} && \text{because } \Delta^* \text{ is the diameter of } P.
\end{aligned}$$

Approximation Factor

- We obtained the inequalities $\Delta' - \varepsilon'\sqrt{d} \leqslant \Delta^* \leqslant \Delta' + \varepsilon'\sqrt{d}$.
- We want to find Δ such that $\Delta \leqslant \Delta^* \leqslant (1+\varepsilon)\Delta$.
- So we let $\Delta = \Delta' + \varepsilon'\sqrt{d}$, which yields $\Delta \leqslant \Delta^* \leqslant \Delta + 2\varepsilon'\sqrt{d}$.

- How to choose ε'? As we can compute Δ_0 efficiently, let $\varepsilon' = C\varepsilon\Delta_0$ for some constant C, to be determined later.
- As Δ_0 is a 2-approximation of Δ^*, it will give a $(1+\Theta(\varepsilon))$-approximation.
- More precisely, we know that $\Delta^* \geqslant \Delta_0$, so $\varepsilon'\sqrt{d} \leqslant C\varepsilon\sqrt{d}\Delta^*$, and thus

$$\Delta^* \leqslant \Delta + 2C\varepsilon\sqrt{d}\Delta^*$$

$$\Delta^* \leqslant \Delta \frac{1}{1 - 2C\sqrt{d}\varepsilon}.$$

Approximation Factor

Lemma

For all $0 \leqslant x \leqslant \frac{1}{2}$, *we have* $\dfrac{1}{1-x} \leqslant 1 + 2x$.

- So if we choose $C \leqslant 1/(4\sqrt{d})$, since $\varepsilon < 1$, it follows that

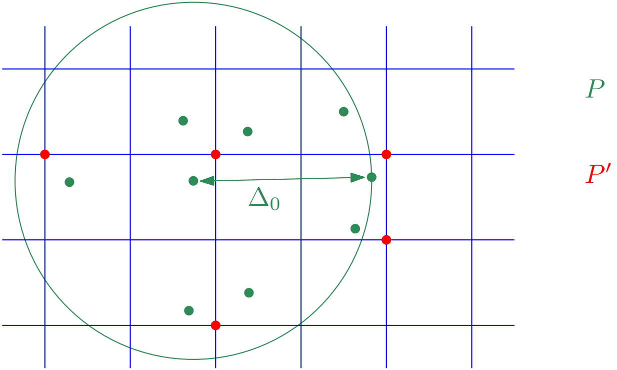
$$\Delta^* \leqslant \Delta(1 + 4C\sqrt{d}\varepsilon).$$

- We now set $C = 1/(4C\sqrt{d})$, and we obtain the desired inequalities

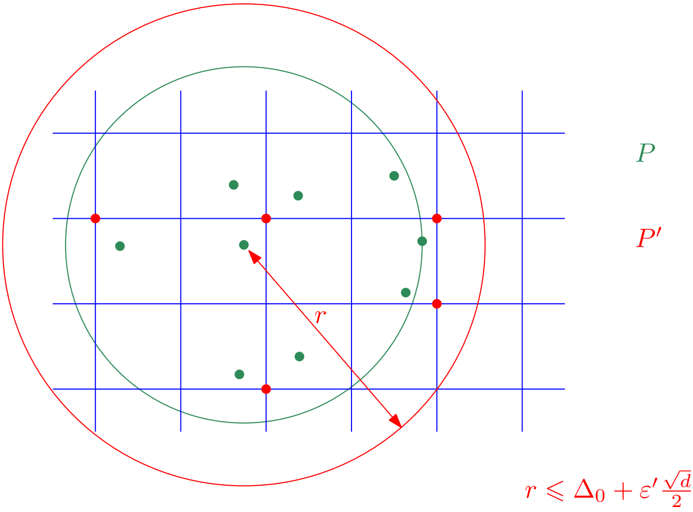
$$\Delta \leqslant \Delta^* \leqslant \Delta(1 + \varepsilon).$$

- In other words, $\Delta = \Delta' + \varepsilon'\sqrt{d}$ is a $(1 + \varepsilon)$-approximation of Δ^*.

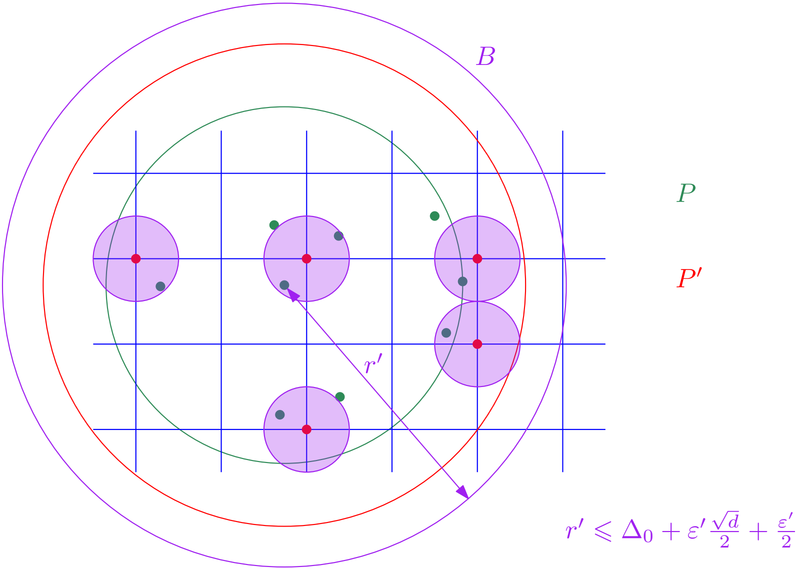
Cardinality of P'



Cardinality of P'



Cardinality of P'



Cardinality of P'

- All the points of P are in a sphere with radius Δ_0.
- So all the points of P' are in a sphere with radius $\Delta_0 + \varepsilon'\sqrt{d}/2$.
- To each point $p \in P'$, we associate a ball $b(p)$ centered at p with radius $\varepsilon'/2$.
- These balls are disjoint and contained in a ball B with radius $\Delta_0 + \varepsilon' \frac{\sqrt{d}}{2} + \frac{\epsilon'}{2}$.
- As $\varepsilon' = \frac{\varepsilon}{4\sqrt{d}}\Delta_0$, this radius is less than $2\Delta_0$.

How Many Grid Points are There?

- The volume of a ball with radius r in dimension d is $C_d r^d$, where C_d depends only on d.
- So the number of balls $b(p), p \in P'$ is at most

$$\frac{C_d(2\Delta_0)^d}{C_d(\varepsilon'/2)^d} = \left(\frac{16\sqrt{d}}{\varepsilon}\right)^d = O((1/\varepsilon)^d).$$

  as we assumed that $d = O(1)$.

- Each point $p \in P'$ is in exactly one ball $b(p)$.
- So $|P'| = O((1/\varepsilon)^d)$.

Summary

- First compute Δ_0 in $O(n)$ time.
- Round all the points to a grid with side length $\varepsilon' = \frac{\varepsilon}{4\sqrt{d}}\Delta_0$.
- It takes $O(n)$ time, and there are $O((1/\varepsilon)^d)$ such points.
- We denote by P' the set of rounded points.
- Compute the diameter Δ' of P' by brute force in $O((1/\varepsilon)^{2d})$ time.
- $\Delta' - \varepsilon'\sqrt{d}$ is a $(1+\varepsilon)$-factor approximation of the diameter Δ^* of P.
- Overall running time: $O(n + (1/\varepsilon)^{2d})$.
- So if ε and d are fixed, this is *linear* time.
- For instance, we can compute an approximation of the diameter of a 3D point set with a 1% error in linear time.