Algorithms for Streaming Data

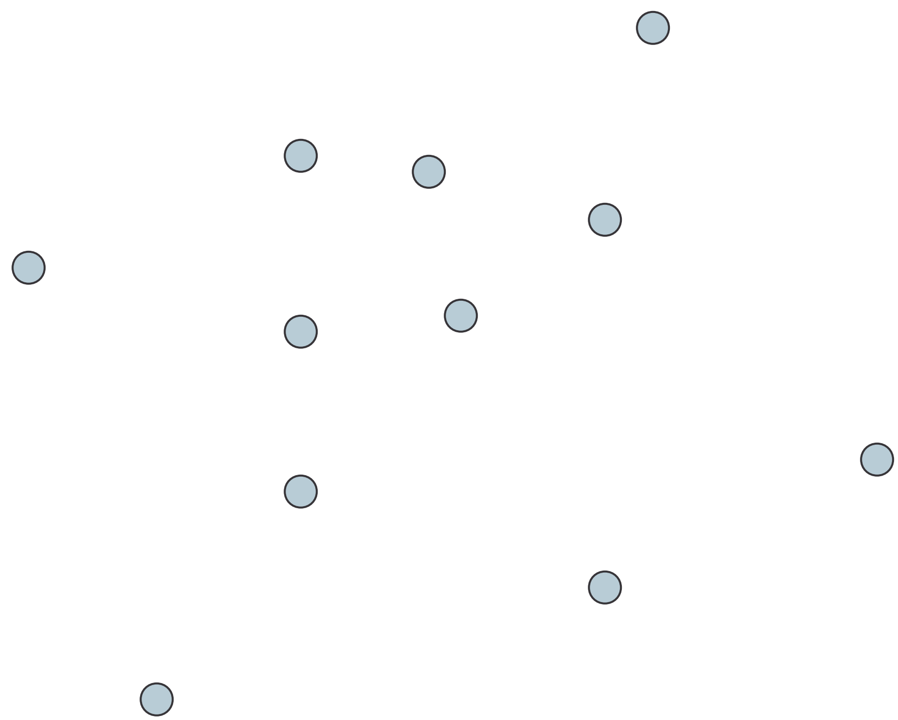
Piotr Indyk

Streaming Data

- Problems defined over points $P=\{p_1,\ldots,p_n\}$
- The algorithm sees p_1, then p_2, then p_3,…
- Key fact: it has limited storage
  - Can store only $s<<n$ points
  - Can store only $s<<n$ bits (need to assume finite precision)

$p_1 \ldots p_2 \ldots p_3 \ldots p_4 \ldots p_5 \ldots p_6 \ldots p_7 \ldots$

Example - diameter



Problems

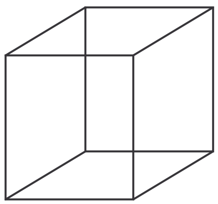
- Diameter
- Minimum enclosing ball
- l_2 norm of a vector

Diameter in $l^{d'}_{\infty}$

- Assume we measure distances according to the l_{∞} norm
- What can we do ?

Diameter in l_∞, ctd.

- From previous lecture we know that
  $Diam_\infty(P)=max_{i=1\ldots d'}\,[max_{p\in P}\, p_i - min_{p\in P}\, p_i]$
- Can maintain max/min in constant space
- Total space $= O(d')$
- What about l_1 ?



Diameter in l_1

- Let $f:l_1^d \rightarrow l_\infty^{2^d}$ be an isometric embedding
- We will maintain $Diam_\infty(f(P))$
  - For each point p, we compute f(p) and feed it to the previous algorithm
  - Return the pair p,q that maximizes $||f(p)-f(q)||_\infty$
- This gives $O(2^d)$ space for l_1^d
- What about l_2 ?

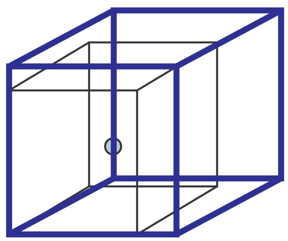
Diameter in l_2

- Let $f: l_2^d \rightarrow l_\infty^{d'}$, $d' = O(1/\varepsilon)^{(d-1)/2}$, be a $(1+\varepsilon)$-distortion embedding
- Apply the same algorithm as before
- Parameters:
  - Space: $O(1/\varepsilon)^{(d-1)/2}$
  - Time: ?

Minimum Enclosing Ball

- Problem: given $P=\{p_1 \ldots p_n\}$, find center o and radius $r>0$ such that
  - $P \subseteq B(o,r)$
  - r is as small as possible
- Solve the problem in l_∞
- Generalize to l_1 and l_2 via embeddings

MEB in l_∞



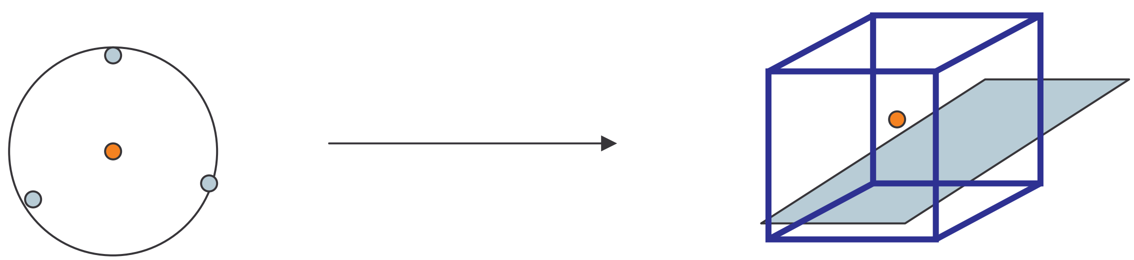
- Let C be the hyper-rectangle defined by max/min in every dimension
- Easy to see that min radius ball B(o,r) is a min size hypercube that contains C
- Min radius = min side length/2
- How to solve it in l_2 ?

MEB in l_2

- Firstly, assume $(1+\varepsilon)\approx 1$
- Let $f: l_2^d \rightarrow l_\infty^{d'}$ be an "almost" isometric embedding
- Algorithm:
  - For each point p, compute $f(p)$
  - Maintain MEB_∞ $B'(o',r)$ of $f(p_1)\ldots f(p_n)$
  - Compute o such that $f(o)=o'$
  - Report $B(o,r)$

Problem

- There might be NO o such that f(o)=o’
- If it was the case, then we would always have MEB radius=Diameter/2, which is not true:



- The problem is that f is into, not onto

The Correct Version

- Algorithm:
  - Maintain the min/max points $f(p_1)\ldots f(p_{2d'})$, two points per dimension
  - Compute MEB $B(o,r)$ of $p_1\ldots p_{2d'}$
  - Report $B(o,r)$

Correctness

MEB radius for P

$= \text{Min } r \text{ s.t. } \exists o \;\; P \subseteq B(o,r)$

$\approx \text{Min } r \text{ s.t. } \exists o \; f(P) \subseteq B(f(o),r)$

$= \text{Min } r \text{ s.t. } \exists o \; \{f(p_1)\ldots f(p_{2d'})\} \subseteq B(f(o),r)$ [see next slide]

$\approx \text{Min } r \text{ s.t. } \exists o \; \{p_1\ldots p_{2d'}\} \subseteq B(o,r)$

$= \text{MEB radius for } \{p_1\ldots p_{2d'}\}$

- Total error at most $(1+\varepsilon)^2$
- In reality, at most $(1+\varepsilon)$

Digression: Core Sets

- In the previous slide we use the fact that in l_∞, for any set P of points, there is a subset P' of P, |P|=2d' , such that MEB(P')=MEB(P)
- P' is called a "core-set" for the MEB of P in l_∞
- For more on core-sets, see the web page by Sariel Har-Peled

Maintaining l_2 norm of a vector

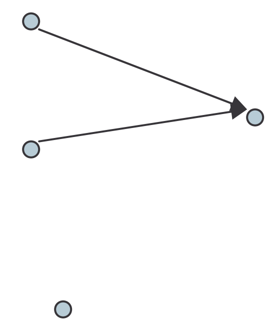
- Implicit vector $x=(x_1,\ldots,x_n)$
- Start with $x=0$
- Stream: sequence of pairs (i,b) , meaning

$$x_i=x_i+b$$

- Goal: maintain (approximately) $||x||_2$

Motivation

- Consider a set of web pages, stored in some order
- Two pages are “similar” if they link to the same page
- Note that each page is similar to itself
- Want to know the number of pairs of similar web pages



Connection to l_2 norm

- Let
  - In(i) be the # in-links to page i
  - Out(i) be the # out-links of page i
- Out(i) is easy to compute, In(i) is not
- We want to compute

  $\frac{1}{2} * \sum_i In(i)\,(In(i)+1) = \frac{1}{2}\,[\sum_i In(i)^2 + \sum_i In(i)]$
- Every time we see link to i: In(i):=In(i)+1

Approximate Algorithm

- We will use roughly $O(\log(1/P)/\varepsilon^2)$ numbers, where $1-P$ is the probability of successful estimation

Algorithm

- From JL lemma, it suffices to maintain Ax for "random" A, since $||Ax|| \approx ||x||$
- Assume
  - we have Ax
  - Need to compute Ay, where $y=x$ except for $y_i=x_i+b$
- Use linearity:

$$Ay = A(y-x)+Ax = A(be_i)+Ax = b\, a^i + Ax$$

Pseudo-randomness

- In practice: use A[i,j]=Normal(RND(i,j))
- In theory: one can use bounded space random generators to generate A using only $O(\log n * \log(1/P)/\varepsilon^2)$ random numbers