Proximity in the Age of Distraction: Robust Approximate Nearest Neighbor Search

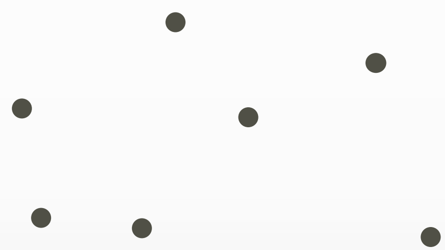
Sariel Har-Peled
UIUC

Sepideh Mahabadi
MIT

Nearest Neighbor Problem

Nearest Neighbor

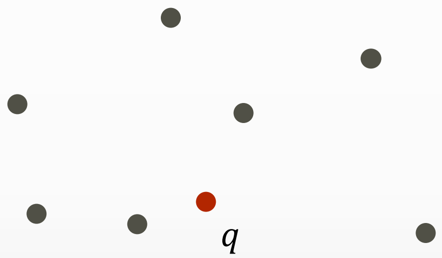
Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$



Nearest Neighbor

Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$

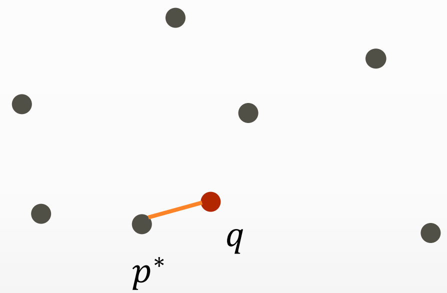
A query point q comes online



Nearest Neighbor

Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$

A query point q comes online



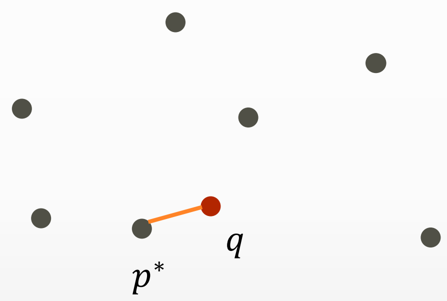
Goal:

- Find the nearest data point p^*

Nearest Neighbor

Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$

A query point q comes online



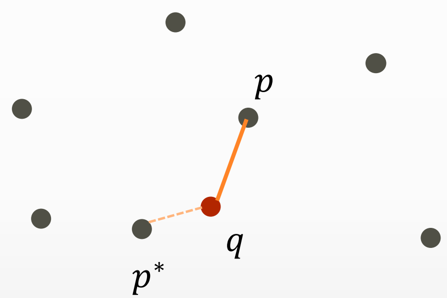
Goal:

- Find the nearest data point p^*
- Do it in sub-linear time and small space

Approximate Nearest Neighbor

Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$

A query point q comes online



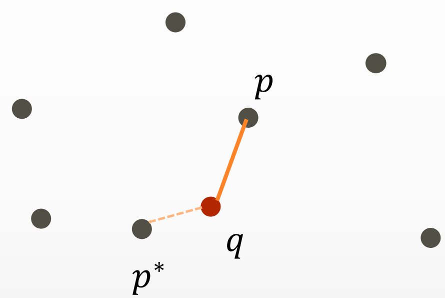
Goal:

- Find the nearest data point p^*
- Do it in sub-linear time and small space
- Approximate Nearest Neighbor
  - If optimal distance is r, report a point in distance cr for $\mathrm{c} = (1 + \epsilon)$

Approximate Nearest Neighbor

Dataset of n points P in a metric space (X, d_X), e.g. $\mathbb{R}^d$

A query point q comes online

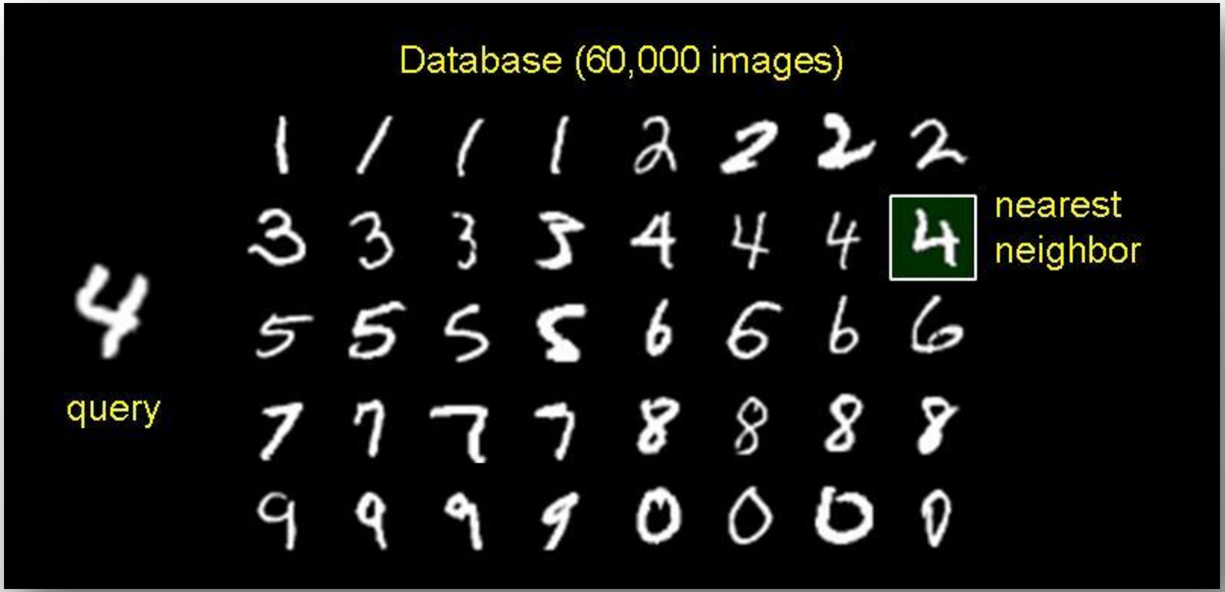


Goal:

- Find the nearest data point p^*
- Do it in sub-linear time and small space
- Approximate Nearest Neighbor
  - If optimal distance is r, report a point in distance cr for $c = (1 + \epsilon)$
  - For Hamming (and L_1) query time is $n^{1/O(c)}$ [IM98]
  - and for Euclidean (L_2) it is $n^{\frac{1}{O(c^2)}}$ [AI08]

Applications of NN

Searching for the closest object



Robust NN Problem

Robustness

The data points are:

Robustness

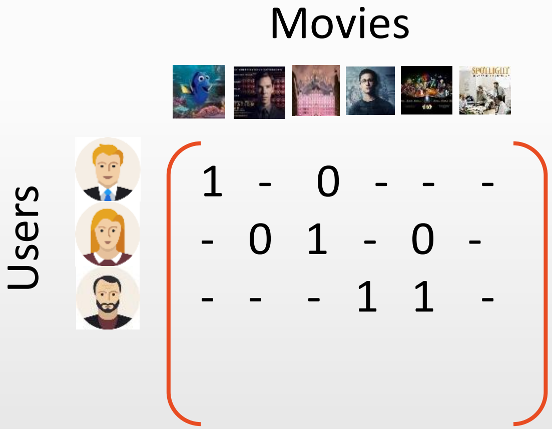
The data points are:

- corrupted, noisy
  - Image denoising

Robustness

The data points are:

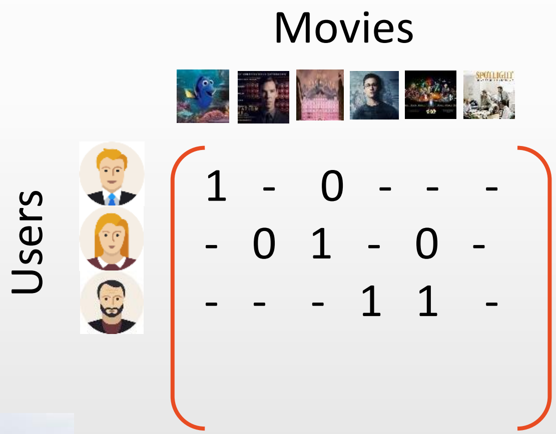
- corrupted, noisy
  - Image denoising
- Incomplete
  - Recommendation: Sparse matrix



Robustness

The data points are:


- corrupted, noisy
  - Image denoising



- Incomplete
  - Recommendation: Sparse matrix

- Irrelevant
  - Occluded image


The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$

n=3

$$p_1 = (3,4,0,5)$$
$$p_2 = (3,2,1,2)$$
$$p_3 = (2,3,3,1)$$

The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$
- **A parameter k**

n=3,k=2

$$p_1 = (3,4,0,5)$$
$$p_2 = (3,2,1,2)$$
$$p_3 = (2,3,3,1)$$

The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$
- **A parameter k**
- A query point q comes online
- Find the closest point after **removing k coordinates**

$$q = (1,2,1,5)$$
$$p_1 = (3,4,0,5)$$
$$p_2 = (3,2,1,2)$$
$$p_3 = (2,3,3,1)$$

n=3,k=2

The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$
- **A parameter k**
- A query point q comes online
- Find the closest point after **removing k coordinates**

$$q = (1,2,1,5)$$
$$p_1 = (3,4,0,5)$$
$$p_2 = (3,2,1,2)$$
$$p_3 = (2,3,3,1)$$

n=3,k=2

dist=1

dist=0

dist=2

The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$
- **A parameter k**
- A query point q comes online
- Find the closest point after **removing k coordinates**

$q = (1,2,1,5)$ **n=3,k=2**

$p_1 = (3,4,0,5)$ **dist=1**

$p_2 = (3,2,1,2)$ **dist=0**

$p_3 = (2,3,3,1)$ **dist=2**

The Robust NN problem

- Dataset of n points P in $\mathbb{R}^d$
- **A parameter k**
- A query point q comes online
- Find the closest point after **removing k coordinates**

$q = (1,2,1,5)$ n=3,k=2

$p_1 = (3,4,0,5)$ dist=1

$p_2 = (3,2,1,2)$ dist=0

$p_3 = (2,3,3,1)$ dist=2

- ➢ Different set of coordinates for different points
- ➢ Applying this naively would require $\binom{d}{k} \approx d^k$

Budgeted Version

- Dataset of n points P in $\mathbb{R}^d$
- d weights
  $w = (w_1, w_2, \dots, w_d) \in [0,1]^d$

$w = (0.5, 0.5, 0.8, 0.3)$

n=3

$p_1 = (1,4,0,3)$
$p_2 = (3,2,4,2)$
$p_3 = (4,6,3,4)$

Budgeted Version

- Dataset of n points P in $\mathbb{R}^d$
- d weights
  $w = (w_1, w_2, \dots, w_d) \in [0,1]^d$
- A query point q comes online
- Find the closest point after removing a set of coordinates B of **weight at most 1**.

$w = (0.5, 0.5, 0.8, 0.3)$

$q = (1,2,5,5)$ **n=3**

$p_1 = (1,4,0,3)$

$p_2 = (3,2,4,2)$

$p_3 = (4,6,3,4)$

Budgeted Version

- Dataset of n points P in $\mathbb{R}^d$
- d weights
  $w = (w_1, w_2, \dots, w_d) \in [0,1]^d$
- A query point q comes online
- Find the closest point after removing a set of coordinates B of **weight at most 1**.

$w = (0.5, 0.5, 0.8, 0.3)$

$q \ = \ (1,2,5,5)$ **n=3**

$p_1 = (1,4,0,3)$

$p_2 = (3,2,4,2)$

$p_3 = (4,6,3,4)$

Budgeted Version

- Dataset of n points P in $\mathbb{R}^d$
- d weights
  $w = (w_1, w_2, \ldots, w_d) \in [0,1]^d$
- A query point q comes online
- Find the closest point after removing a set of coordinates B of **weight at most 1**.

$w = (0.5, 0.5, 0.8, 0.3)$

$q = (1,2,5,5)$ **n=3**

$p_1 = (1,4,0,3)$ **dist=4**

$p_2 = (3,2,4,2)$ **dist=1**

$p_3 = (4,6,3,4)$ **dist=3**

Budgeted Version

- Dataset of n points P in $\mathbb{R}^d$
- d weights
  $w = (w_1, w_2, \ldots, w_d) \in [0,1]^d$
- A query point q comes online
- Find the closest point after removing a set of coordinates B of **weight at most 1**.

$w = (0.5, 0.5, 0.8, 0.3)$

$q = (1,2,5,5)$ **n=3**

$p_1 = (1,4,0,3)$ **dist=4**

$p_2 = (3,2,4,2)$ **dist=1**

$p_3 = (4,6,3,4)$ **dist=3**

Results

Bicriterion Approximation, for L_1 norm

- Suppose that for $p^* \subset P$ we have $dist(q, p^*) = r$ after ignoring k coordinates

Results

Bicriterion Approximation, for L_1 norm

- Suppose that for $p^* \subset P$ we have $dist(q, p^*) = r$ after ignoring k coordinates

- For $\delta \in (0,1)$
  - Report a point p s.t. $dist(q, p) = O(r/\delta)$ after ignoring $O(k/\delta)$ coordinates.
  - Query time equals to n^δ queries in 2-ANN data-structure

Results

Bicriterion Approximation, for L_1 norm

- Suppose that for $p^* \subset P$ we have $dist(q, p^*) = r$ after ignoring k coordinates

- For $\delta \in (0,1)$
  - Report a point p s.t. $dist(q, p) = O(r/\delta)$ after ignoring $O(k/\delta)$ coordinates.
  - Query time equals to n^δ queries in 2-ANN data-structure

Why not single criterion?

- Equivalent to exact near neighbor in Hamming: there is a point within distance r of the query iff there is a point within distance 0 after ignoring $k = r$ coordinates

Results

	distance	#ignored coordinates	Query Time	
			#Queries	Query type
Opt	r	k		

Results

	distance	#ignored coordinates	Query Time	
			#Queries	Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^{δ}	2-ANN

Results

	distance	#ignored coordinates	Query Time #Queries	Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^δ	2-ANN
L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/\mathrm{p}})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^δ	$c^{1/\mathrm{p}}$-ANN

Results

	distance	#ignored coordinates	Query Time #Queries	 Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^{δ}	2-ANN
L_{p}	$O(r\left(c+\frac{1}{\delta}\right)^{1/\mathrm{p}})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/\mathrm{p}}$-ANN
$(1+\epsilon)$-approximation	$r(1+\epsilon)$	$O(\frac{k}{\epsilon\delta})$	$O(\frac{n^{\delta}}{\epsilon})$	$(1+\epsilon)$ −ANN

Results

	distance	#ignored coordinates	Query Time #Queries	Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^{δ}	2-ANN
L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN
$(1+\epsilon)$-**approximation**	$r(1+\epsilon)$	$O(\frac{k}{\epsilon\delta})$	$O(\frac{n^{\delta}}{\epsilon})$	$(1+\epsilon)$ −ANN
Budgeted Version	$O(r)$	Weight of $O(1)$	n^{δ} $+O(n^{\delta}d^4)$	2-ANN

Algorithm

High Level Algorithm

Theorem. If for a point $p^* \subset P$, the L_1 distance of q and p^* is at most r after removing k coordinates, there exists an algorithm which reports a point p whose distance to q is $O(r/\delta)$ after removing $O(k/\delta)$ coordinates.

High Level Algorithm

Theorem. If for a point $p^* \subset P$, the L_1 distance of q and p^* is at most r after removing k coordinates, there exists an algorithm which reports a point p whose distance to q is $O(r/\delta)$ after removing $O(k/\delta)$ coordinates.

- Cannot apply randomized dimensionality reduction e.g. Johnson-Lindenstrauss

High Level Algorithm

Theorem. If for a point $p^* \subset P$, the L_1 distance of q and p^* is at most r after removing k coordinates, there exists an algorithm which reports a point p whose distance to q is $O(r/\delta)$ after removing $O(k/\delta)$ coordinates.

- Cannot apply randomized dimensionality reduction e.g. Johnson-Lindenstrauss
- A set of randomized maps $\boldsymbol{f_1, f_2, \ldots f_m}: \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
  - All of them map far points from query to far points
  - At least one of them maps a close point to a close point

High Level Algorithm

Theorem. If for a point $p^* \subset P$, the L_1 distance of q and p^* is at most r after removing k coordinates, there exists an algorithm which reports a point p whose distance to q is $O(r/\delta)$ after removing $O(k/\delta)$ coordinates.

- Cannot apply randomized dimensionality reduction e.g. Johnson-Lindenstrauss
- A set of randomized maps $\boldsymbol{f_1, f_2, \ldots f_m} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
  - All of them map far points from query to far points
  - At least one of them maps a close point to a close point
- W.l.o.g. assume that the query is the origin
  - Find the data point with minimum norm.

A Randomized Map

- Embed all the points using a random mapping $f\colon \mathbb{R}^d \to \mathbb{R}^{d'}$:

A Randomized Map

- Embed all the points using a random mapping $\boldsymbol{f}: \mathbb{R}^{\boldsymbol{d}} \to \mathbb{R}^{\boldsymbol{d}'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k

A Randomized Map

- Embed all the points using a random mapping $f: \mathbb{R}^d \to \mathbb{R}^{d'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k
  - E.g. $d = 5$
    - round 1: coordinates (1,3,4) sampled
    - round 2: coordinate (4) sampled
    - $v = (3,6,1,2,4)$ maps to $f(v) = (3,1,2,2)$

A Randomized Map

- Embed all the points using a random mapping $f: \mathbb{R}^d \to \mathbb{R}^{d'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k
  - E.g. $d = 5$
    - round 1: coordinates (1,3,4) sampled
    - round 2: coordinate (4) sampled
    - $v = (3,6,1,2,4)$ maps to $f(v) = (3,1,2,2)$

$$\mathbb{E}[d'] = O(d \ln n \cdot \frac{\delta}{k})$$

A Randomized Map

- Embed all the points using a random mapping $\boldsymbol{f}: \mathbb{R}^{\boldsymbol{d}} \rightarrow \mathbb{R}^{\boldsymbol{d}'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k
  - E.g. $d = 5$
    - round 1: coordinates (1,3,4) sampled
    - round 2: coordinate (4) sampled
    - $v = (3,6,1,2,4)$ maps to $f(v) = (3,1,2,2)$

$$\mathbb{E}[\boldsymbol{d}'] = \boldsymbol{O}(\boldsymbol{d} \ln \boldsymbol{n} \cdot \frac{\boldsymbol{\delta}}{\boldsymbol{k}})$$

- **Simple setup:** Consider a vector v where each coordinate is either 0 or ∞

A Randomized Map

- Embed all the points using a random mapping $f: \mathbb{R}^d \to \mathbb{R}^{d'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k
  - E.g. $d = 5$
    - round 1: coordinates (1,3,4) sampled
    - round 2: coordinate (4) sampled
    - $v = (3,6,1,2,4)$ maps to $f(v) = (3,1,2,2)$

$$\mathbb{E}[d'] = O(d \ln n \cdot \frac{\delta}{k})$$

- **Simple setup:** Consider a vector v where each coordinate is either 0 or ∞
- **Close point:**
  - v has at most k large coordinates
  - Probability of avoiding large coordinates is **at least** $\left(1 - \frac{\delta}{k}\right)^{k \cdot \ln n} \approx n^{-\delta}$

A Randomized Map

- Embed all the points using a random mapping $\boldsymbol{f}: \mathbb{R}^{d} \rightarrow \mathbb{R}^{d'}$:
  - Repeat $t = O(\ln n)$ times
    - Sample each coordinate in $[d]$ with probability δ/k
  - E.g. $d = 5$
    - round 1: coordinates (1,3,4) sampled
    - round 2: coordinate (4) sampled
    - $v = (3,6,1,2,4)$ maps to $f(v) = (3,1,2,2)$

$$\mathbb{E}[\boldsymbol{d'}] = \boldsymbol{O}\left(\boldsymbol{d} \ln \boldsymbol{n} \cdot \frac{\boldsymbol{\delta}}{\boldsymbol{k}}\right)$$

- **Simple setup:** Consider a vector v where each coordinate is either 0 or ∞
- **Close point:**
  - v has at most k large coordinates
  - Probability of avoiding large coordinates is **at least** $\left(1 - \frac{\delta}{k}\right)^{k \cdot \ln n} \approx n^{-\delta}$
- **Far point**
  - v has at least k/δ large coordinates
  - Probability of missing large coordinates is **at most** $\left(1 - \frac{\delta}{k}\right)^{(k/\delta) \cdot \ln n} \approx 1/n$

Outline

- Embed all the points using a random mapping $f: \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
- With probability $n^{-\delta}$
  - all far points will be mapped to far points under L_1 distance
  - a close by point will be mapped to a close by point under L_1 distance.

Outline

- Embed all the points using a random mapping $f: \mathbb{R}^d \to \mathbb{R}^{d'}$
- With probability $n^{-\delta}$
  - all far points will be mapped to far points under L_1 distance
  - a close by point will be mapped to a close by point under L_1 distance.
  - **We can use ANN as a black-box to find it**

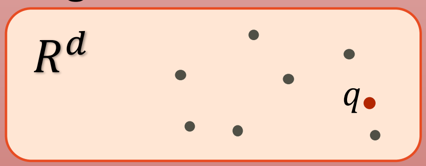
Outline

- Embed all the points using a random mapping $f: \mathbb{R}^d \to \mathbb{R}^{d'}$
- With probability $n^{-\delta}$
  - all far points will be mapped to far points under L_1 distance
  - a close by point will be mapped to a close by point under L_1 distance.
  - **We can use ANN as a black-box to find it**

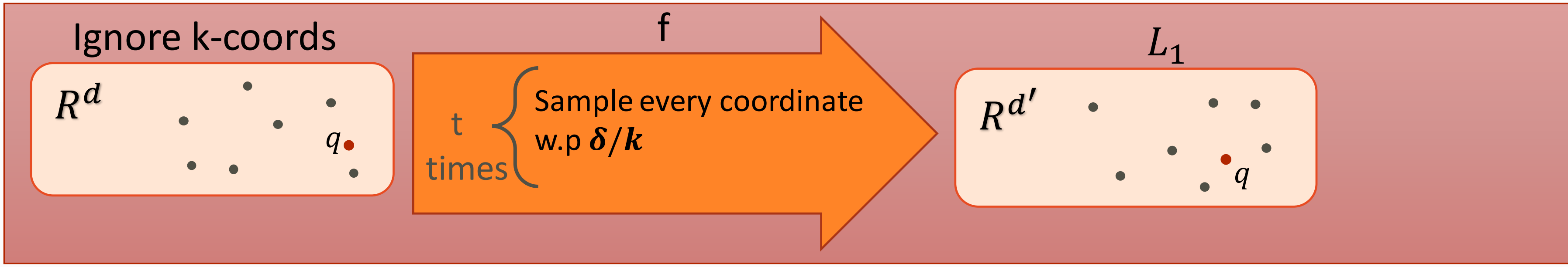
- Repeat this embedding $\mathrm{O}(n^\delta \log n)$ times and report the best.

Algorithm

Ignore k-coords

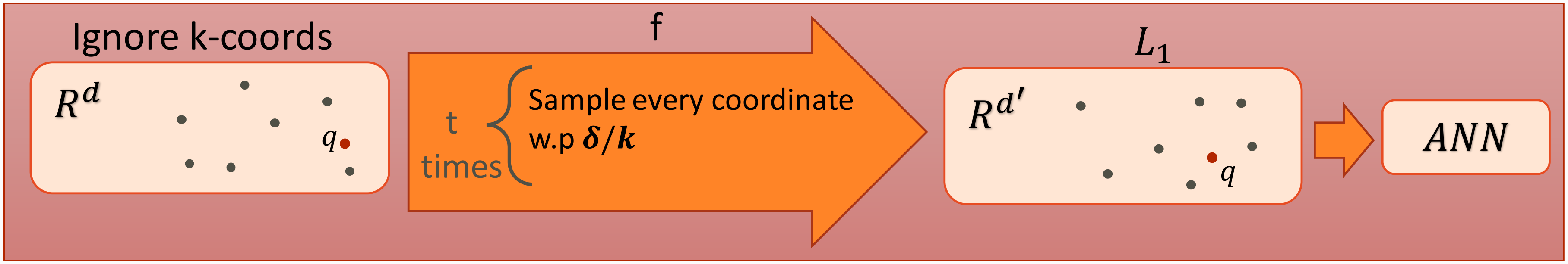


Algorithm



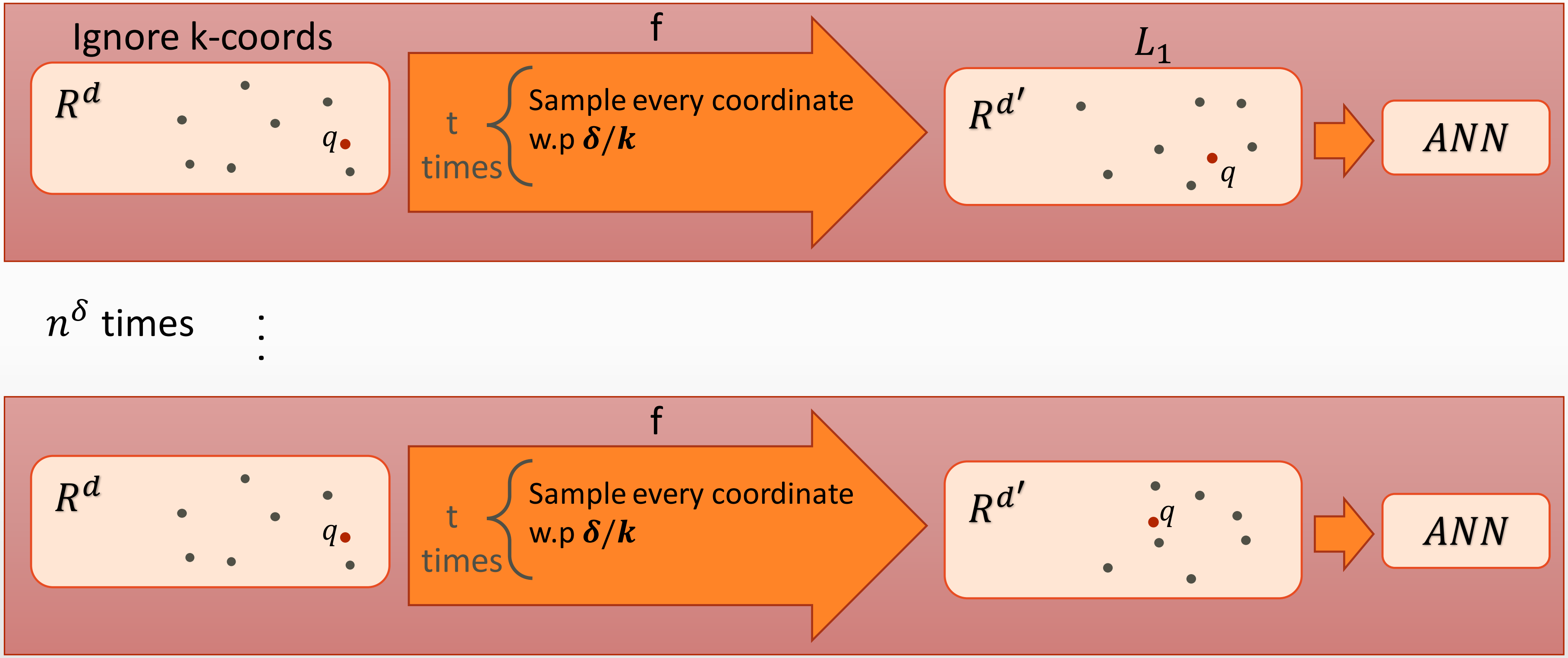
Ignore k-coords
R^d
q
f
t
times
Sample every coordinate
w.p $\boldsymbol{\delta/k}$
L_1
$R^{d'}$
q

Algorithm

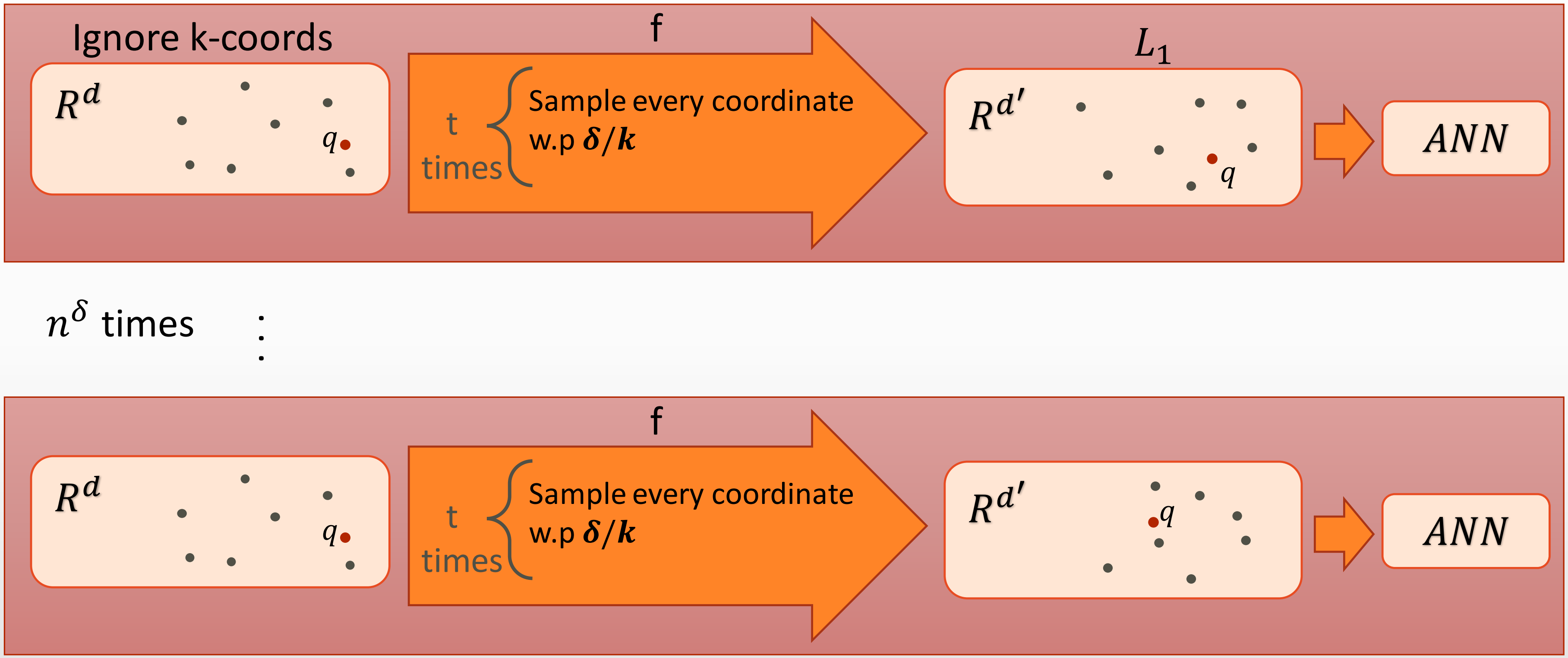


Ignore k-coords
R^d
q
f
t times
Sample every coordinate w.p $\boldsymbol{\delta/k}$
L_1
$R^{d'}$
q
ANN

Algorithm


Ignore k-coords
f
L_1
R^d
q
t times
Sample every coordinate w.p $\boldsymbol{\delta/k}$
$R^{d'}$
q
ANN
n^δ times
f
R^d
q
t times
Sample every coordinate w.p $\boldsymbol{\delta/k}$
$R^{d'}$
q
ANN

Algorithm



Check the distance of all n^δ candidates and report the closest one after ignoring k coordinates

Analysis

Truncation

Coordinates are not necessarily 0 and ∞

Truncation

Coordinates are not necessarily 0 and ∞

Let f' be obtained by sampling every coordinate with probability $\tau = \delta / k$

Truncation

Coordinates are not necessarily 0 and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$

Truncation

Coordinates are not necessarily 0 and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Truncation

Coordinates are not necessarily $\mathbf{0}$ and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta / k}$

- $\mathbb{E}[|f'(v)|_1] = \tau |v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

Truncation

Coordinates are not necessarily $\mathbf{0}$ and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$

Truncation

Coordinates are not necessarily $\mathbf{0}$ and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$
- $\boldsymbol{r}$ **−Light point:** a point with norm $\leq r$ after truncation
- **R− Heavy point:** a point with norm $\geq R$ after truncation

Truncation

Coordinates are not necessarily 0 and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$
- $\boldsymbol{r}$ **–Light point:** a point with norm $\leq r$ after truncation
- **R– Heavy point:** a point with norm $\geq R$ after truncation
- **Close point:** a point with norm $\leq r$ after ignoring k coordinates
- **Far point:** a point with norm $\geq r/\delta$ after ignoring k/δ coordinates

Truncation

Coordinates are not necessarily 0 and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$
- $\boldsymbol{r}$ **−Light point:** a point with norm $\leq r$ after truncation
- **R− Heavy point:** a point with norm $\geq R$ after truncation
- **Close point:** a point with norm $\leq r$ after ignoring k coordinates
- **Far point:** a point with norm $\geq r/\delta$ after ignoring k/δ coordinates

Claim:

- A close point is $2r$ −light.

Truncation

Coordinates are not necessarily $\mathbf{0}$ and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$
- $\boldsymbol{r}$ **−Light point:** a point with norm $\leq r$ after truncation
- **R− Heavy point:** a point with norm $\geq R$ after truncation
- **Close point:** a point with norm $\leq r$ after ignoring k coordinates
- **Far point:** a point with norm $\geq r/\delta$ after ignoring k/δ coordinates

Claim:

- A close point is $2r$ −light.
- A far point is $\frac{r}{\delta}$ −heavy.

Truncation

Coordinates are not necessarily 0 and ∞

Let $\boldsymbol{f'}$ be obtained by sampling every coordinate with probability $\boldsymbol{\tau = \delta/k}$

- $\mathbb{E}[|f'(v)|_1] = \tau|v|_1$
- $\mathbf{Var}[|f'(v)|_1] = \tau\,(1-\tau)|v|_2^2$

Need to **bound the influence** of every coordinate.

- Truncate every coordinate at $\boldsymbol{r/k}$,i.e., $\boldsymbol{v_i = \min\{v_i, r/k\}}$
- $\boldsymbol{r}$ **−Light point:** a point with norm $\leq r$ after truncation
- **R− Heavy point:** a point with norm $\geq R$ after truncation
- **Close point:** a point with norm $\leq r$ after ignoring k coordinates
- **Far point:** a point with norm $\geq r/\delta$ after ignoring k/δ coordinates

Claim:

- A close point is $2r$ −light.
- A far point is $\frac{r}{\delta}$ −heavy.

➢ **Analyze the behavior of the maps over the truncated points instead.**

L_1 Norm

Using truncation

- Bound the variance and prove concentration for $\boldsymbol{f'}$ by Chebyshev

L_1 Norm

Using truncation

- Bound the variance and prove concentration for f' by Chebyshev

f **is a concatenation of** $t = O(\ln n)$ **such** f'

L_1 Norm

Using truncation

- Bound the variance and prove concentration for $\boldsymbol{f'}$ by Chebyshev

$\boldsymbol{f}$ is a concatenation of $\boldsymbol{t = O(\ln n)}$ such $\boldsymbol{f'}$

- $\mathbb{E}[|f(v)|_1] = t\tau|v|_1$
- Prove concentration for $\boldsymbol{f}$ using Chernoff

L_1 Norm

Using truncation

- Bound the variance and prove concentration for f' by Chebyshev

f is a concatenation of $t = O(\ln n)$ such f'

- $\mathbb{E}[|f(v)|_1] = t\tau|v|_1$
- Prove concentration for f using Chernoff

	distance	#ignored coordinates	Query Time #Queries	Query Time Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^δ	2-ANN

Generalizations

- L_p norm

Generalizations

- **L_p norm**
  - Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

Generalizations

- ❑ $\boldsymbol{L_p}$ **norm**

- Minimize the $|\boldsymbol{v}|_{\boldsymbol{p}}^{\boldsymbol{p}}$ norm, i.e., $\sum_{\boldsymbol{i}} \boldsymbol{v}_{\boldsymbol{i}}^{\boldsymbol{p}}$ similar to the L_1 norm

$L_{\boldsymbol{p}}$	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/\mathrm{p}}$**-ANN**

Generalizations

- ❑ L_p **norm**

- Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-**ANN**

- ❑ **Budgeted**

Generalizations

- **L_p norm**

- Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN

- **Budgeted**

- **Map:**
  - sample coordinate i with probability proportional to $1/w_i$

Generalizations

- ❑ L_p **norm**
- Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN

- ❑ **Budgeted**
- **Map:**
  - sample coordinate i with probability proportional to $1/w_i$
  - To maintain the expectation multiply sampled coordinates by w_i

Generalizations

- L_p **norm**

- Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN

- **Budgeted**

- **Map:**
  - sample coordinate i with probability proportional to $1/w_i$
  - To maintain the expectation multiply sampled coordinates by w_i
- **Truncation:**
  - Truncate coordinate i with by value $\frac{r}{c/w_i - 1}$

Generalizations

MIT CSAIL

- **L_p norm**
  - Minimize the $|v|_p^p$ norm, i.e., $\sum_i v_i^p$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN

- **Budgeted**
  - **Map:**
    - sample coordinate i with probability proportional to $1/w_i$
    - To maintain the expectation multiply sampled coordinates by w_i
  - **Truncation:**
    - Truncate coordinate i with by value $\frac{r}{c/w_i-1}$
    - E.g. a coordinate of cost approaching 0 will be truncated to 0

Generalizations

- ❑ $\boldsymbol{L_p}$ **norm**

- Minimize the $|\boldsymbol{v}|_{\boldsymbol{p}}^{\boldsymbol{p}}$ norm, i.e., $\sum_i \boldsymbol{v}_{\boldsymbol{i}}^{\boldsymbol{p}}$ similar to the L_1 norm

L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/p}$-ANN

- ❑ **Budgeted**

- **Map:**
  - sample coordinate $\boldsymbol{i}$ with probability proportional to $\mathbf{1}/\boldsymbol{w}_{\boldsymbol{i}}$
  - To maintain the expectation multiply sampled coordinates by $\boldsymbol{w}_{\boldsymbol{i}}$
- **Truncation:**
  - Truncate coordinate $\boldsymbol{i}$ with by value $\frac{\boldsymbol{r}}{\boldsymbol{c}/\boldsymbol{w}_{\boldsymbol{i}}-\mathbf{1}}$
  - E.g. a coordinate of cost approaching 0 will be truncated to 0

Budgeted Version	$O(r)$	Weight of $O(1)$	n^{δ} $+O(n^{\delta}d^4)$	**2-ANN**

Conclusion

	distance	#ignored coordinates	Query Time #Queries	Query Time Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^{δ}	2-ANN
L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/\mathbf{p}})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^{δ}	$c^{1/\mathbf{p}}$-ANN
$(1+\epsilon)$-approximation	$r(1+\epsilon)$	$O(\frac{k}{\epsilon\delta})$	$O(\frac{n^{\delta}}{\epsilon})$	$(1+\epsilon)$ −ANN
Budgeted Version	$O(r)$	Weight of $O(1)$	n^{δ} $+O(n^{\delta}d^{4})$	2-ANN

Conclusion

	distance	#ignored coordinates	Query Time	
			#Queries	Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^δ	2-ANN
L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^δ	$c^{1/\mathbf{p}}$-ANN
$(1+\epsilon)$-**approximation**	$r(1+\epsilon)$	$O(\frac{k}{\epsilon\delta})$	$O(\frac{n^\delta}{\epsilon})$	$(1+\epsilon)$ −ANN
Budgeted Version	$O(r)$	Weight of $O(1)$	$n^\delta + O(n^\delta d^4)$	2-ANN

Open Problems

- Improve the dependence on δ
- Prove lower bounds

Conclusion

	distance	#ignored coordinates	Query Time #Queries	Query type
Opt	r	k		
L_1	$O(\frac{r}{\delta})$	$O(\frac{k}{\delta})$	n^δ	2-ANN
L_p	$O(r\left(c+\frac{1}{\delta}\right)^{1/p})$	$O(k\left(c+\frac{1}{\delta}\right))$	n^δ	$c^{1/\mathbf{p}}$-ANN
$(1+\epsilon)$-approximation	$r(1+\epsilon)$	$O(\frac{k}{\epsilon\delta})$	$O(\frac{n^\delta}{\epsilon})$	$(1+\epsilon)$ −ANN
Budgeted Version	$O(r)$	Weight of $O(1)$	n^δ $+O(n^\delta d^4)$	2-ANN

Open Problems

- Improve the dependence on δ
- Prove lower bounds

Thank You!