Randomized partition trees for exact nearest neighbor search

Sanjoy Dasgupta and Kaushik Sinha

COLT 2013

The complexity of nearest neighbor search

Given a data set of n points in $\mathbb{R}^d$, build a data structure for efficiently answering subsequent nearest neighbor queries q.

- Data structure should take space $O(n)$
- Query time should be $o(n)$

The complexity of nearest neighbor search

Given a data set of n points in $\mathbb{R}^d$, build a data structure for efficiently answering subsequent nearest neighbor queries q.

- Data structure should take space $O(n)$
- Query time should be $o(n)$

Unproven but common conjecture: either data structure size or query time must be exponential in d.

Bad case: for any $0 < \epsilon < 1$,

- Pick $2^{O(\epsilon^2 d)}$ points uniformly from the unit sphere in $\mathbb{R}^d$
- With high probability, all interpoint distances are $(1 \pm \epsilon)\sqrt{2}$

The complexity of nearest neighbor search

Given a data set of n points in $\mathbb{R}^d$, build a data structure for efficiently answering subsequent nearest neighbor queries q.

- Data structure should take space $O(n)$
- Query time should be $o(n)$

Unproven but common conjecture: either data structure size or query time must be exponential in d.

Bad case: for any $0 < \epsilon < 1$,

- Pick $2^{O(\epsilon^2 d)}$ points uniformly from the unit sphere in $\mathbb{R}^d$
- With high probability, all interpoint distances are $(1 \pm \epsilon)\sqrt{2}$

How can this bad case be defeated?

Approximate nearest neighbor

For data set $S \subset \mathbb{R}^d$ and query q, a c-approximate nearest neighbor is any $x \in S$ such that

$$\|x - q\| \leq c \cdot \min_{z \in S} \|z - q\|.$$

Approximate nearest neighbor

For data set $S \subset \mathbb{R}^d$ and query q, a c-approximate nearest neighbor is any $x \in S$ such that

$$\|x - q\| \leq c \cdot \min_{z \in S} \|z - q\|.$$

Locality-sensitive hashing (Indyk, Motwani, Andoni):

- Data structure size n^{1+1/c^2}
- Query time n^{1/c^2}

Approximate nearest neighbor

For data set $S \subset \mathbb{R}^d$ and query q, a c-approximate nearest neighbor is any $x \in S$ such that

$$\|x - q\| \leq c \cdot \min_{z \in S} \|z - q\|.$$

Locality-sensitive hashing (Indyk, Motwani, Andoni):

- Data structure size n^{1+1/c^2}
- Query time n^{1/c^2}

Disadvantage: the same value of c can have very different implications for different data sets.

Low intrinsic dimension

Several schemes for exact NN search have been analyzed under a *doubling measure* assumption.
We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

Low intrinsic dimension

Several schemes for exact NN search have been analyzed under a *doubling measure* assumption.
We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

For instance, *cover tree* (Beygelzimer, Kakade, Langford):

- Works in any metric space
- Data structure size $O(n)$
- Query time $2^{O(d_o)} \log n$

Low intrinsic dimension

Several schemes for exact NN search have been analyzed under a *doubling measure* assumption.
We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

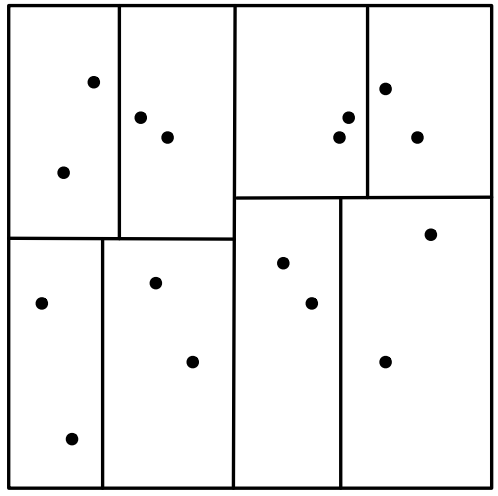
$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

For instance, *cover tree* (Beygelzimer, Kakade, Langford):

- Works in any metric space
- Data structure size $O(n)$
- Query time $2^{O(d_o)} \log n$

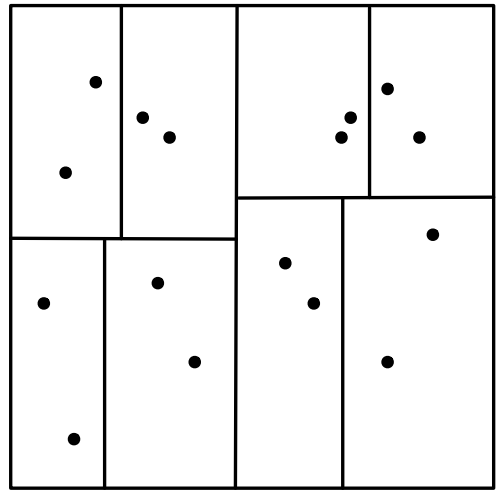
Disadvantage: "low doubling measure" unlikely to hold broadly.

The k-d tree: a hierarchical partition of $\mathbb{R}^d$



Typically used with *defeatist search*: return NN in query point's leaf node. This might fail to return the true NN.

The k-d tree: a hierarchical partition of $\mathbb{R}^d$

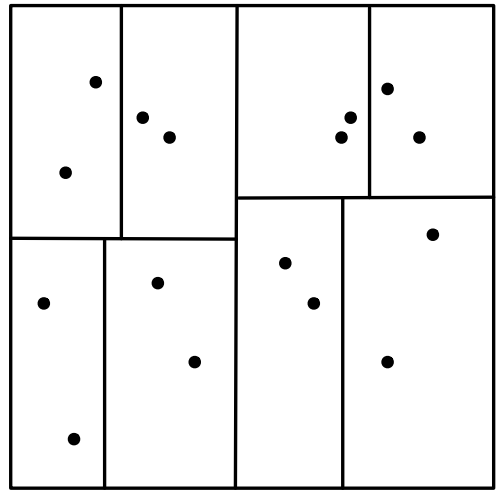


Typically used with *defeatist search*: return NN in query point's leaf node. This might fail to return the true NN.

Ways of reducing the failure probability in high dimension:

- Random coordinate basis, or random split directions (Liu, Moore, Gray, and Kang)
- Overlapping cells (Maneewongvatana and Mount; Liu et al)

The k-d tree: a hierarchical partition of $\mathbb{R}^d$



Typically used with *defeatist search*: return NN in query point's leaf node. This might fail to return the true NN.

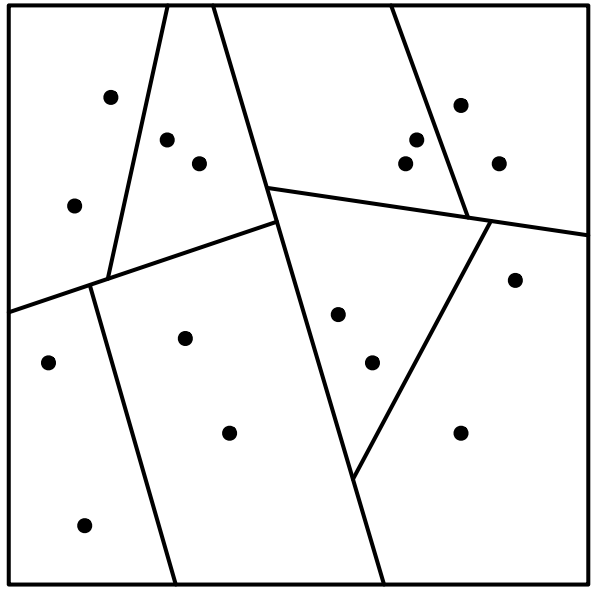
Ways of reducing the failure probability in high dimension:

- Random coordinate basis, or random split directions (Liu, Moore, Gray, and Kang)
- Overlapping cells (Maneewongvatana and Mount; Liu et al)

Analyze the failure probability of k-d trees and these variants

Random projection tree

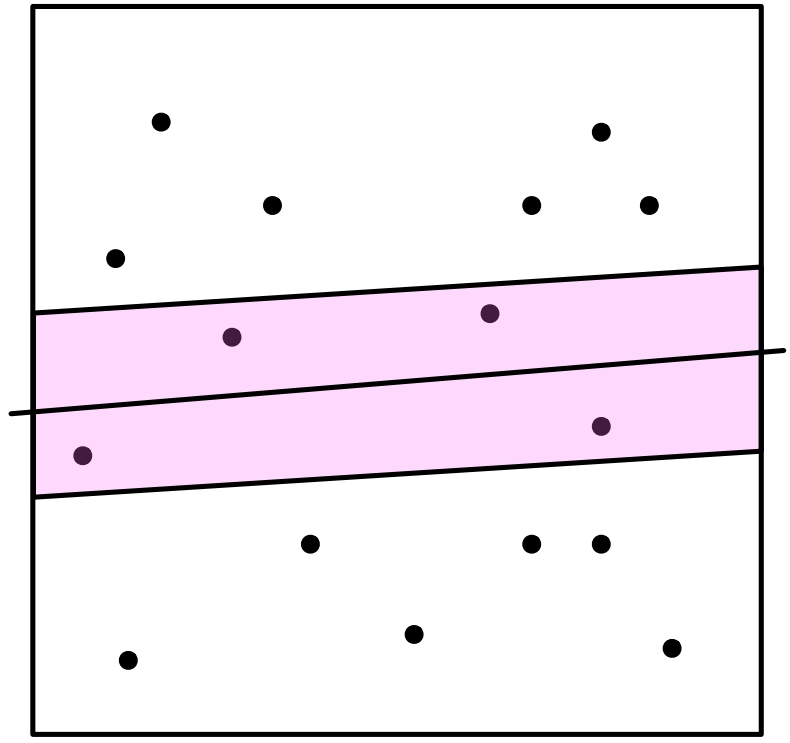
In each cell of the tree, pick split direction uniformly at random from the unit sphere in $\mathbb{R}^d$



Perturbed split: after projection, pick $\beta \in_R [1/4, 3/4]$ and split at the β-fractile point.

Overlapping cells: spill trees

Overlapping split points: $1/2 - \alpha$ fractile and $1/2 + \alpha$ fractile

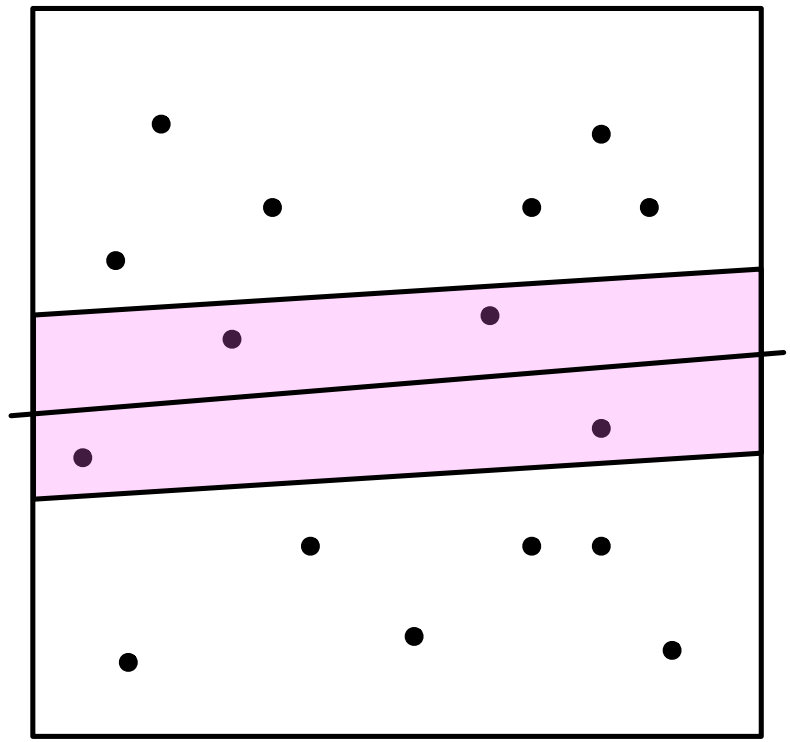


Procedure:

- Route data (to multiple leaves) using overlapping splits
- Route query (to single leaf) using median split

Overlapping cells: spill trees

Overlapping split points: $1/2 - \alpha$ fractile and $1/2 + \alpha$ fractile

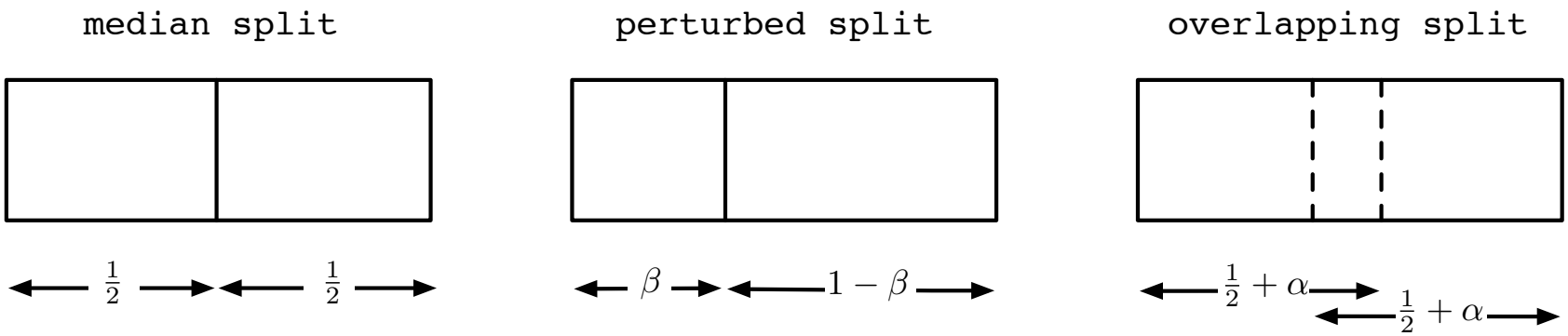


Procedure:

- Route data (to multiple leaves) using overlapping splits
- Route query (to single leaf) using median split

Spill tree has size $n^{1/(1-\lg(1+2\alpha))}$: e.g. $n^{1.159}$ for $\alpha = 0.05$.

Three trees



	Routing data	Routing queries
k-d tree	Median split	Median split
RP tree	Perturbed split	Perturbed split
Spill tree	Overlapping split	Median split
Virtual spill tree	Median split	Overlapping split

Failure probability

Pick any data set $x_1, \ldots, x_n$ and any query q.

- Let $x_{(1)}, \ldots, x_{(n)}$ be the ordering of data by distance from q.
- Probability of not returning the NN depends directly on

$$\Phi(q, \{x_1, \ldots, x_n\}) = \frac{1}{n} \sum_{i=2}^{n} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

(This probability is over the randomization in tree construction.)

Failure probability

Pick any data set $x_1, \ldots, x_n$ and any query q.

- Let $x_{(1)}, \ldots, x_{(n)}$ be the ordering of data by distance from q.
- Probability of not returning the NN depends directly on

$$\Phi(q, \{x_1, \ldots, x_n\}) = \frac{1}{n} \sum_{i=2}^{n} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

(This probability is over the randomization in tree construction.)

- Either of the two spill trees: failure probability $\propto \Phi$
- RP tree: failure probability $\propto \Phi \log 1/\Phi$

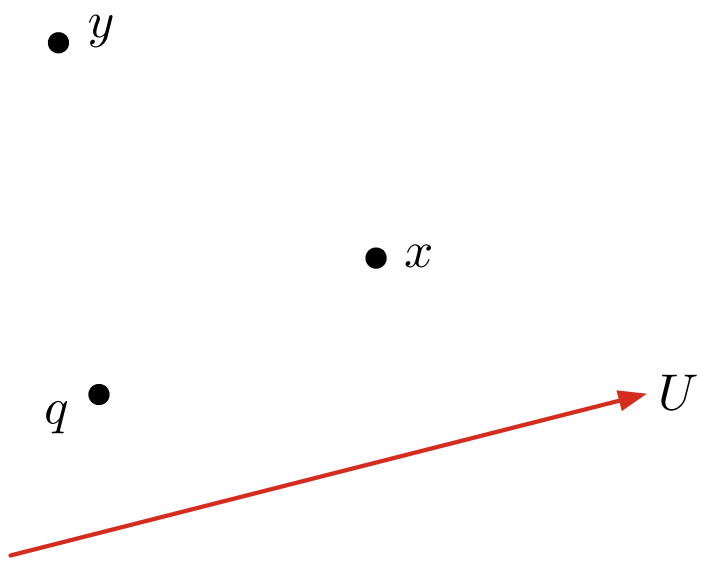
Random projection of three points

Lemma

Pick any three points $q, x, y \in \mathbb{R}^d$ such that $\|q - x\| < \|q - y\|$.
Pick U uniformly at random from the unit sphere S^{d-1} in $\mathbb{R}^d$.
Then

$$Pr(y \cdot U \text{ falls between } q \cdot U \text{ and } x \cdot U) \leq \frac{1}{2} \frac{\|q - x\|}{\|q - y\|}.$$

(Tight within a constant unless the points are almost-collinear)



Random projection of a set of points

Lemma

Pick any three points $q, x, y \in \mathbb{R}^d$ such that $\|q - x\| < \|q - y\|$. Pick U uniformly at random from the unit sphere S^{d-1} in $\mathbb{R}^d$. Then

$$Pr(y \cdot U \text{ falls between } q \cdot U \text{ and } x \cdot U) \leq \frac{1}{2}\frac{\|q - x\|}{\|q - y\|}.$$

Lemma

Pick any $x_1, \ldots, x_n$ and any query q. Pick $U \in_R S^{d-1}$ and project all points onto direction U. Then the expected fraction of the projected x_i that fall between q and $x_{(1)}$ is at most

$$\frac{1}{2n}\sum_{i=2}^{n}\frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|} = \frac{1}{2}\Phi$$

A single cell of the spill tree

Fix any data points $x_1, \ldots, x_n$ and query q. For $m \leq n$, define

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

A single cell of the spill tree

Fix any data points $x_1, \ldots, x_n$ and query q. For $m \leq n$, define

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

Spill tree overlap region is α fraction of the data on either side of median. Therefore:

Lemma

Suppose the query q reaches a cell of the spill tree containing m of the data points, including $x_{(1)}$. The probability that q is separated from $x_{(1)}$ when this cell is split is at most $\Phi_m/(2\alpha)$.

A single cell of the spill tree

Fix any data points $x_1, \ldots, x_n$ and query q. For $m \leq n$, define

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

Spill tree overlap region is α fraction of the data on either side of median. Therefore:

Lemma

Suppose the query q reaches a cell of the spill tree containing m of the data points, including $x_{(1)}$. The probability that q is separated from $x_{(1)}$ when this cell is split is at most $\Phi_m/(2\alpha)$.

Proof: when the m points (and q) are projected onto a random direction, expected fraction falling between q and $x_{(1)}$ is $\leq \Phi_m/2$. The only way q can get split from $x_{(1)}$ is if this fraction exceeds α.

Failure probability of NN search

Theorem

Suppose a randomized spill tree is built for data set $x_1, \ldots, x_n$ with leaf nodes of size n_o. For any query q, the probability that the NN query does not return $x_{(1)}$ is at most

$$\frac{1}{2\alpha} \sum_{i=0}^{\ell} \Phi_{\beta^i n}(q, \{x_1, \ldots, x_n\})$$

where $\beta = 1/2 + \alpha$ and $\ell = \log_{1/\beta}(n/n_o)$ is the tree's depth.

Failure probability of NN search

Theorem

Suppose a randomized spill tree is built for data set $x_1, \ldots, x_n$ with leaf nodes of size n_o. For any query q, the probability that the NN query does not return $x_{(1)}$ is at most

$$\frac{1}{2\alpha} \sum_{i=0}^{\ell} \Phi_{\beta^i n}(q, \{x_1, \ldots, x_n\})$$

where $\beta = 1/2 + \alpha$ and $\ell = \log_{1/\beta}(n/n_o)$ is the tree's depth.

- Virtual spill tree: same result, with $\beta = 1/2$

Failure probability of NN search

Theorem

Suppose a randomized spill tree is built for data set $x_1, \ldots, x_n$ with leaf nodes of size n_o. For any query q, the probability that the NN query does not return $x_{(1)}$ is at most

$$\frac{1}{2\alpha} \sum_{i=0}^{\ell} \Phi_{\beta^i n}(q, \{x_1, \ldots, x_n\})$$

where $\beta = 1/2 + \alpha$ and $\ell = \log_{1/\beta}(n/n_o)$ is the tree's depth.

- Virtual spill tree: same result, with $\beta = 1/2$
- RP tree: same result, with $\beta = 3/4$ and $\Phi \to \Phi \ln(2e/\Phi)$

Failure probability of NN search

Theorem

Suppose a randomized spill tree is built for data set $x_1, \ldots, x_n$ with leaf nodes of size n_o. For any query q, the probability that the NN query does not return $x_{(1)}$ is at most

$$\frac{1}{2\alpha} \sum_{i=0}^{\ell} \Phi_{\beta^i n}(q, \{x_1, \ldots, x_n\})$$

where $\beta = 1/2 + \alpha$ and $\ell = \log_{1/\beta}(n/n_o)$ is the tree's depth.

- Virtual spill tree: same result, with $\beta = 1/2$
- RP tree: same result, with $\beta = 3/4$ and $\Phi \to \Phi \ln(2e/\Phi)$
- Extension to k nearest neighbors is immediate

Doubling measures

Recall: We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

What does this imply about

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}?$$

Doubling measures

Recall: We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

What does this imply about

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}?$$

- NN of q is at distance Δ, say. Then, roughly:
  $\leq 2^{d_o}$ points at distance $\leq 2\Delta$
  $\leq 2^{2d_o}$ points at distance $\leq 4\Delta$

  and so on. Therefore $\Phi_m \leq 1/m^{1/d_o}$.

Doubling measures

Recall: We say a distribution μ on $\mathbb{R}^d$ is a doubling measure of dimension d_o if for all x in the support of μ and all $r > 0$,

$$\mu(B(x, 2r)) \leq 2^{d_o} \mu(B(x, r)).$$

What does this imply about

$$\Phi_m(q, \{x_1, \ldots, x_n\}) = \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}?$$

- NN of q is at distance Δ, say. Then, roughly:
  $\leq 2^{d_o}$ points at distance $\leq 2\Delta$
  $\leq 2^{2d_o}$ points at distance $\leq 4\Delta$
  and so on. Therefore $\Phi_m \leq 1/m^{1/d_o}$.
- For constant failure probability, use spill tree with leaf size $n_o = O(d_o^{d_o})$, and query time $O(n_o + \log n)$.

A simple topic model

Each document is a binary vector $X \in \{0,1\}^N$, generated thus:

- Pick a topic $1 \leq j \leq t$, with probabilities $w_1, \ldots, w_t$
- Coords of X are chosen independently, and $\Pr(X_i = 1) = p_i^{(j)}$

A simple topic model

Each document is a binary vector $X \in \{0,1\}^N$, generated thus:

- Pick a topic $1 \leq j \leq t$, with probabilities $w_1, \ldots, w_t$
- Coords of X are chosen independently, and $\Pr(X_i = 1) = p_i^{(j)}$

Not a good case for NN search!

- Doubling measure: # data points in $B(q, r)$ is doubled when r is multiplied by a constant
- Topic model: # data points in $B(q, r)$ is doubled when r is incremented by a constant

A simple topic model

Each document is a binary vector $X \in \{0,1\}^N$, generated thus:

- Pick a topic $1 \leq j \leq t$, with probabilities $w_1, \ldots, w_t$
- Coords of X are chosen independently, and $\Pr(X_i = 1) = p_i^{(j)}$

Not a good case for NN search!

- Doubling measure: # data points in $B(q, r)$ is doubled when r is multiplied by a constant
- Topic model: # data points in $B(q, r)$ is doubled when r is incremented by a constant

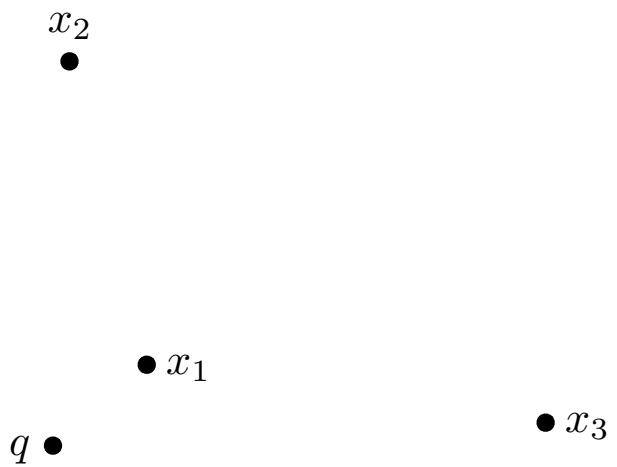
Resulting query time $\sim n \cdot 2^{-\sqrt{L}}$, where L is the expected document length.

Are random directions needed?

Suppose query q is the origin and $x_1, \ldots, x_n$ are chosen thus:

- x_1 is the all-ones vector in $\mathbb{R}^d$
- For each $i > 1$, pick x_i as follows:
  - Pick one coordinate at random and set it to a large number M
  - Set all other coords to values chosen uniformly from $[0, 1]$

For large enough M, the NN of q is x_1.

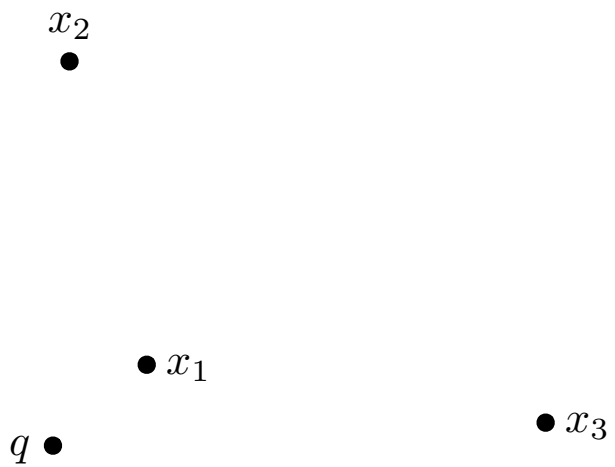


Are random directions needed?

Suppose query q is the origin and $x_1, \ldots, x_n$ are chosen thus:

- x_1 is the all-ones vector in $\mathbb{R}^d$
- For each $i > 1$, pick x_i as follows:
  - Pick one coordinate at random and set it to a large number M
  - Set all other coords to values chosen uniformly from $[0, 1]$

For large enough M, the NN of q is x_1.



On any coordinate projection, about $1 - 1/d$ fraction of the points fall between q and x_1. Yet $\Phi \approx 0$, so random directions work well.

Open problems

1. Can the function Φ be used to analyze other NN schemes?

$$\Phi(q, \{x_1, \ldots, x_n\}) = \frac{1}{n} \sum_{i=2}^{n} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

   Or is there perhaps some more general characterization of the difficulty of NN search instances?

Open problems

1. Can the function Φ be used to analyze other NN schemes?

$$\Phi(q, \{x_1, \ldots, x_n\}) = \frac{1}{n} \sum_{i=2}^{n} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}$$

   Or is there perhaps some more general characterization of the difficulty of NN search instances?

2. *Principal component trees* seem to do well in practice–how can they be analyzed?

Thanks

To Lawrence Cayton for initial discussions, and to the National Science Foundation for support under grant IIS-1162581.