Diversity Maximization over Large Data Sets

Sepideh Mahabadi

Toyota Technological Institute at Chicago
(TTIC)

May 2, 2019

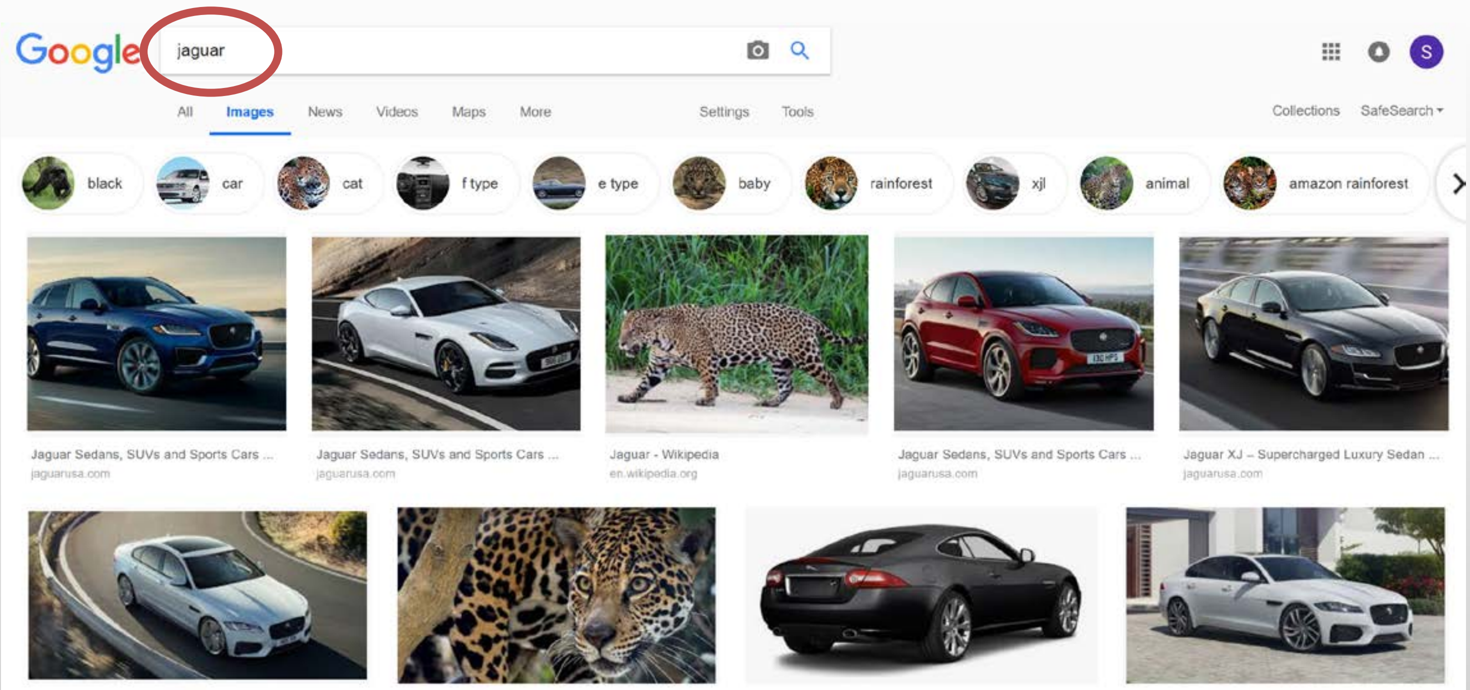
Diversity Maximization

Given a set of objects, how to pick a few of them while maximizing diversity?

Diversity Maximization

Given a set of objects, how to pick a few of them while maximizing diversity?

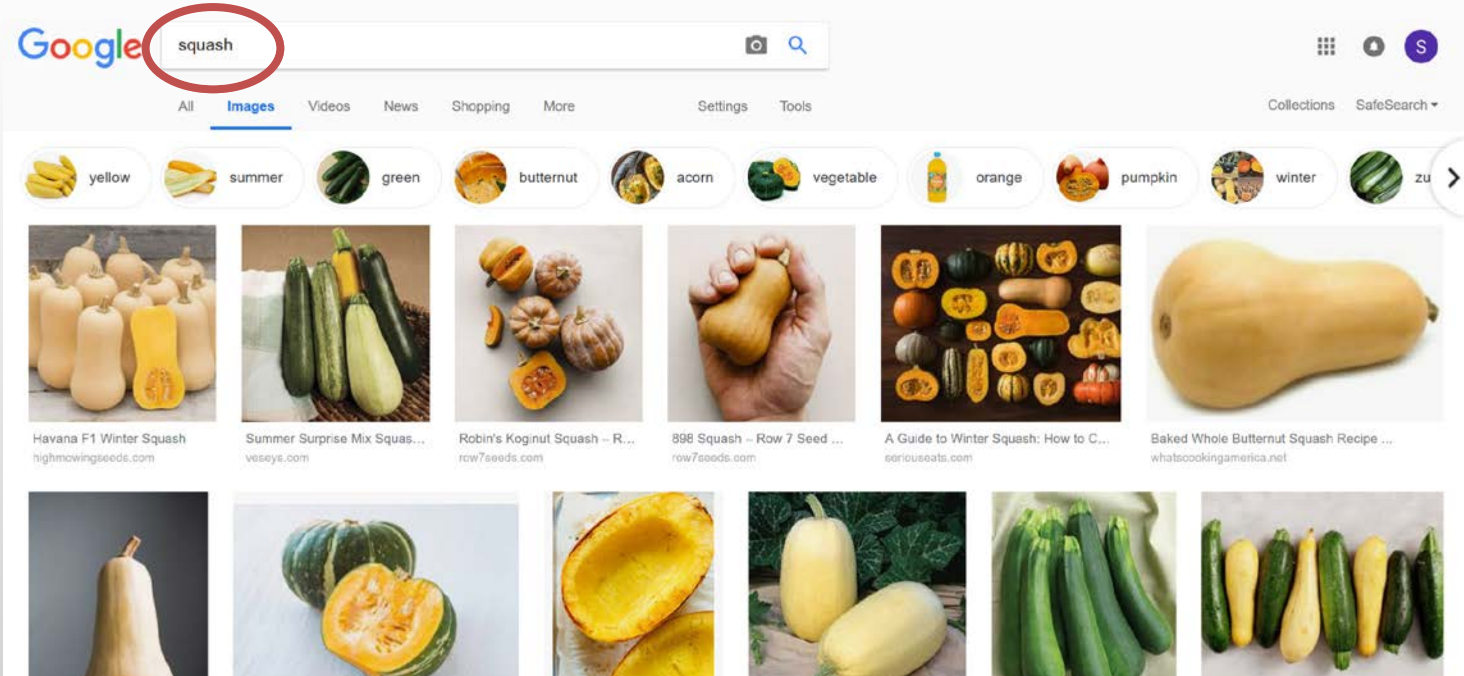
- **Searching**



Diversity Maximization

Given a set of objects, how to pick a few of them while maximizing diversity?

- **Searching**



Diversity Maximization

Given a set of objects, how to pick a few of them while maximizing diversity?

- Searching
- **Recommender Systems**

Image from: http://news.mit.edu/2017/better-recommendation-algorithm-1206

Diversity Maximization

Given a set of objects, how to pick a few of them while maximizing diversity?

- Searching
- Recommender Systems
- Summarization
- Object detection, …

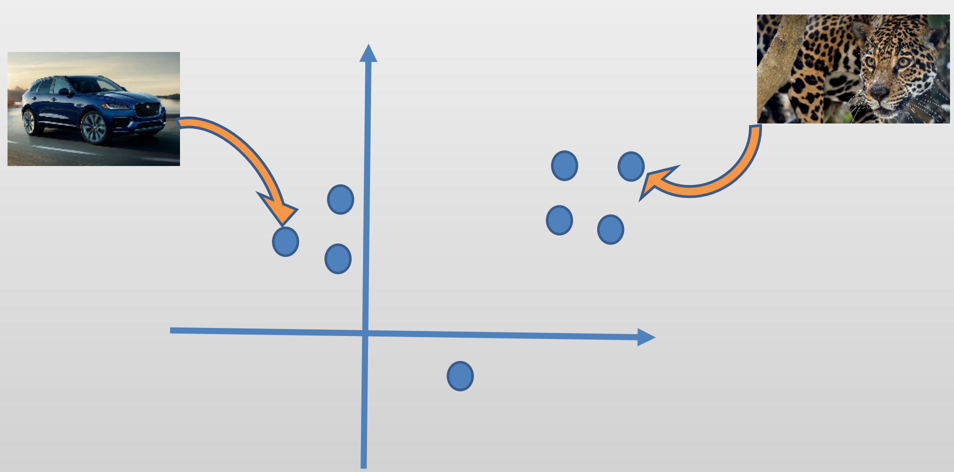
➢ **A small subset of items must be selected to represent the larger population**

Diversity Maximization: The Model

Objects (documents, images, etc) → Feature Vectors → **Points in a high dimensional space**

E.g.

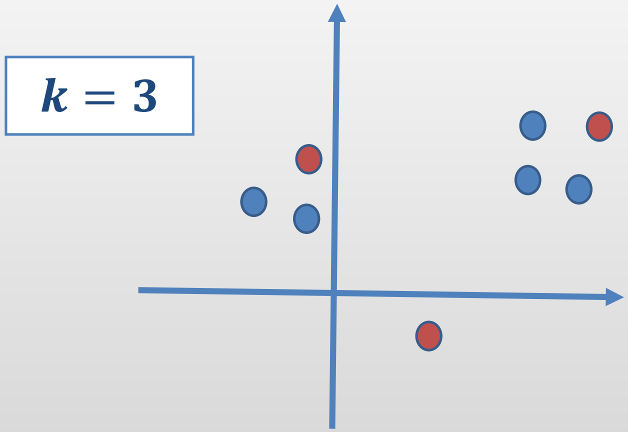
- **Objects:** images
- **Dimensions:** pixels
- **Values:** intensity of the image in the corresponding pixel



Diversity Maximization: The Model

Input: a set of n vectors $V \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points while maximizing "diversity".**

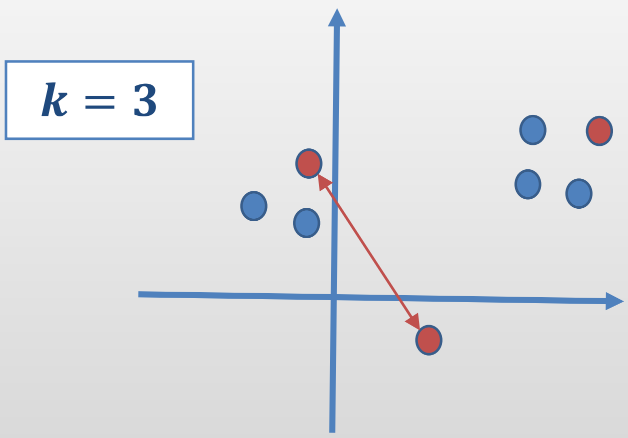


What is Diversity?

Diversity I: Minimum Pairwise Distance

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

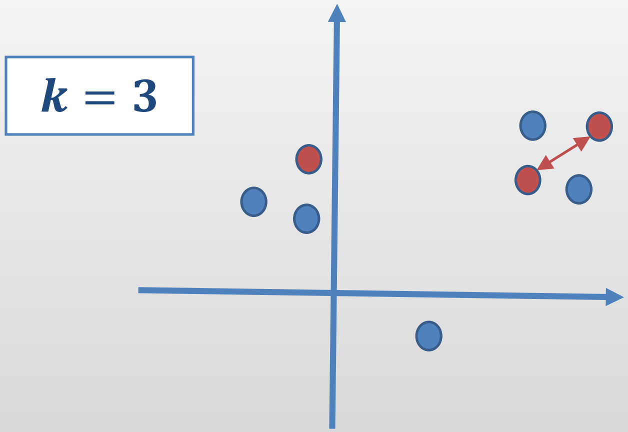
Goal: **pick k points s.t. the minimum pairwise distance of the picked points is maximized.**



Diversity I: Minimum Pairwise Distance

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the minimum pairwise distance of the picked points is maximized.**

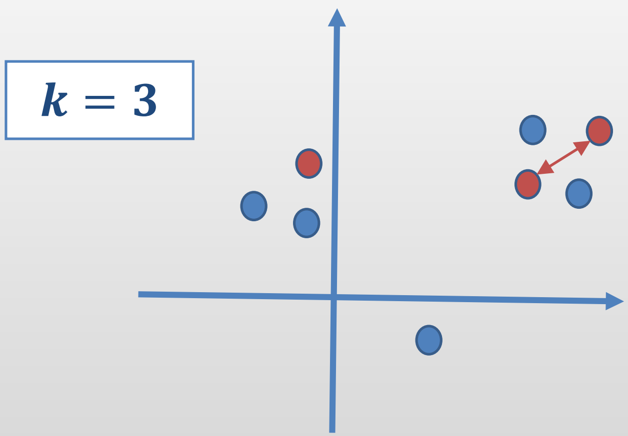


Diversity I: Minimum Pairwise Distance

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the minimum pairwise distance of the picked points is maximized.**

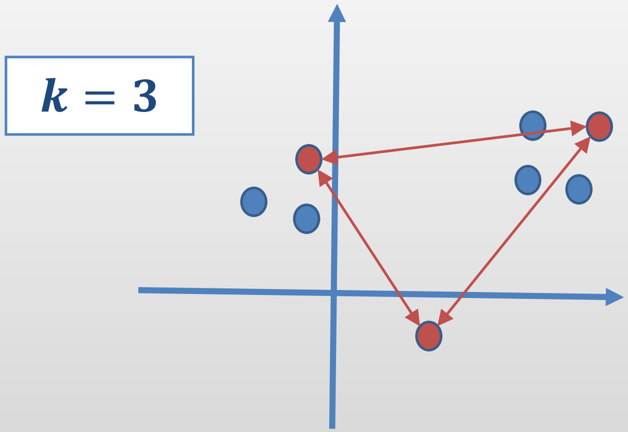
- **Greedy Algorithm**



Diversity II: Sum of Pairwise Distances

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

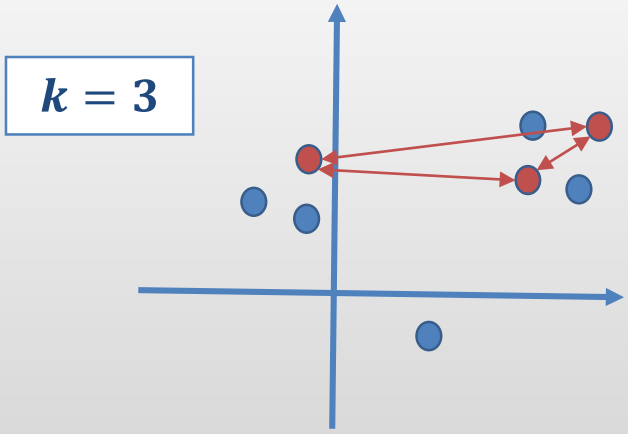
Goal: **pick k points s.t. the sum of pairwise distances of the picked points is maximized.**



Diversity II: Sum of Pairwise Distances

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the sum of pairwise distances of the picked points is maximized.**

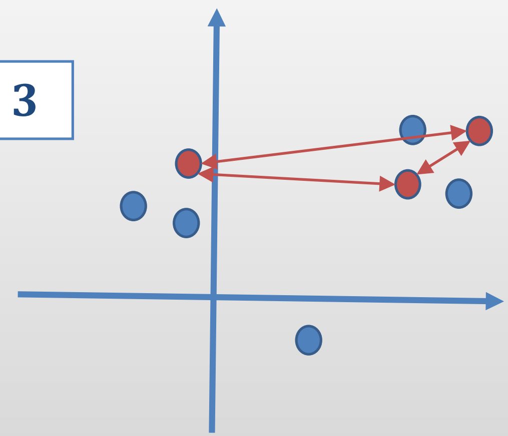


Diversity II: Sum of Pairwise Distances

Input: a set of n vectors $V \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the sum of pairwise distances of the picked points is maximized.**

$\boldsymbol{k = 3}$

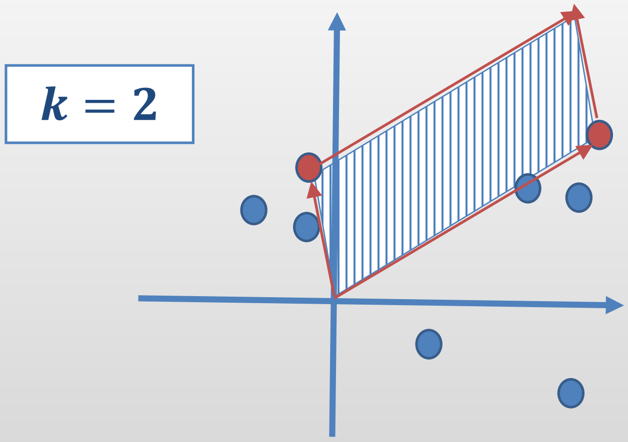


❑ **Local Search Algorithm**

Diversity III: Volume

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

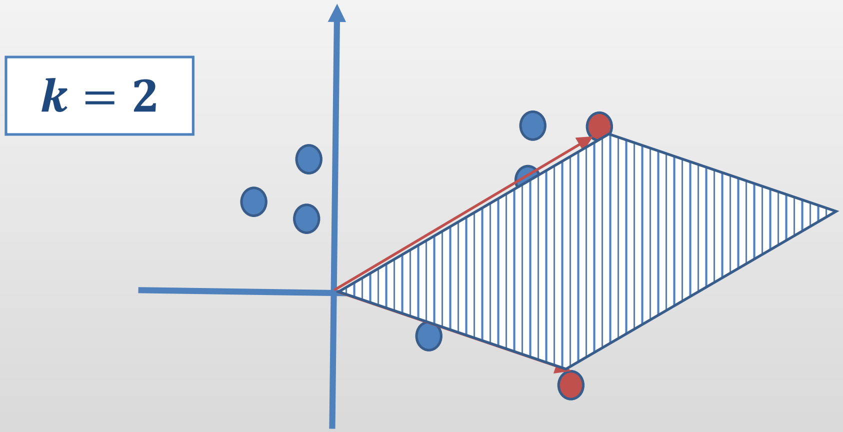
Goal: **pick k points s.t. the volume of the parallelepiped spanned by them is maximized.**



Diversity III: Volume

Input: a set of n vectors $V \subset \mathbb{R}^d$ and a parameter $k \leq d$,

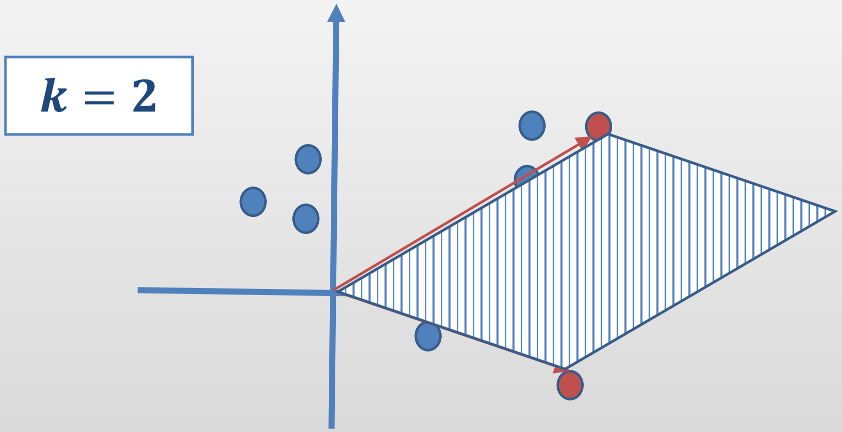
Goal: **pick k points s.t. the volume of the parallelepiped spanned by them is maximized.**



Diversity III: Volume

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the volume of the parallelepiped spanned by them is maximized.**

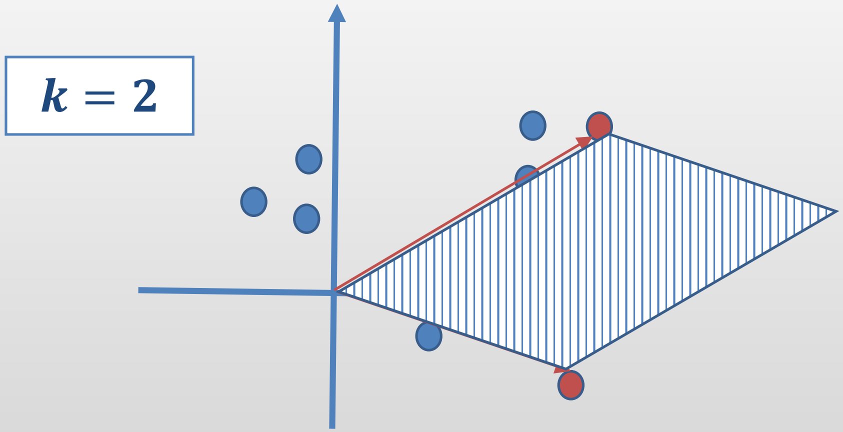


- ❑ **Convex optimization + randomized rounding**

Diversity III: Volume

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: **pick k points s.t. the volume of the parallelepiped spanned by them is maximized.**



- ❑ **Convex optimization + randomized rounding**
- ❑ **Higher order notion of diversity (not based on pairwise distances only)**

Existing Results on Diversity Maximization

Diversity maximization in the offline setting

Diversity Notion	Offline
Min Pairwise Distance	$\theta(1)$ [Ravi et al 94]
Sum of Pairwise distances	$\theta(1)$ [Hassin et al 97]
…	…
Volume	$O\left(c^k\right), \Omega(c'^k)$ [Nik'15],[CIM'13]

Diversity maximization over large data sets

- Background on diversity maximization and how to model diversity
- **Notion of composable core-sets**
- Algorithms for finding core-sets for diversity maximization

1. Maximizing the minimum pairwise distance
2. Maximizing the volume

Diversity maximization over large data sets

[MJK'17,GCGS'14]
Video summarization

[KT+'12, CGGS'15,KT'11]
Document summarization

[YFZ+'16]
Tweet generation

[LCYO'16]
Object detection
....

- Most applications deal with **massive data**
- Lots of effort for solving the problem in massive data models of computation [MJK'17, WIB'14, PJG+'14, MKSK'13, MKBK'15, MZ'15, MZ'15, BENW'15]
- e.g. streaming, distributed, parallel

Diversity maximization over large data sets

[MJK'17,GCGS'14]
Video summarization

[KT+'12, CGGS'15,KT'11]
Document summarization

[YFZ+'16]
Tweet generation

[LCYO'16]
Object detection
....

- Most applications deal with **massive data**
- Lots of effort for solving the problem in massive data models of computation [MJK'17, WIB'14, PJG+'14, MKSK'13, MKBK'15, MZ'15, MZ'15, BENW'15]
- e.g. streaming, distributed, parallel

Composable Core-sets

Composable Core-sets

Core-sets [AHV'05]**:** a subset S of the data V that represents it well

Solving the problem over S gives a good approximation of solving the problem over V

Composable Core-sets

Core-sets [AHV'05]**:** a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

Composable Core-sets

Core-sets [AHV'05]**:** a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let $\boldsymbol{f}$ be an optimization function

–E.g. $\boldsymbol{f}(\boldsymbol{V})$ is the solution of diversity maximization

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

Composable Core-sets

Core-sets [AHV'05]: a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let $\boldsymbol{f}$ be an optimization function

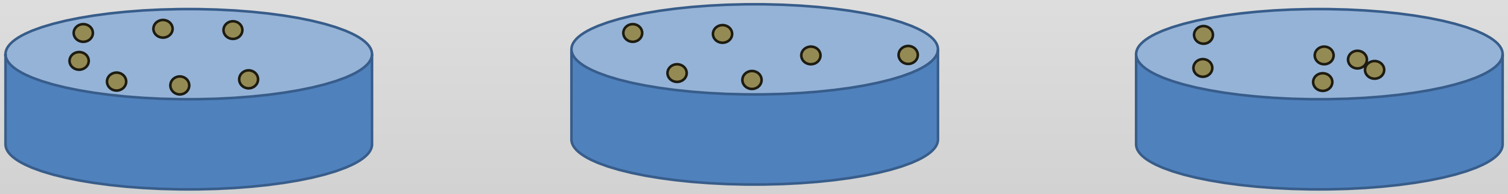
–E.g. $\boldsymbol{f}(\boldsymbol{V})$ is the solution of diversity maximization

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

i.e. for multiple data sets $\boldsymbol{V_1}, \cdots, \boldsymbol{V_m}$ and their coresets $\boldsymbol{S_1}, \cdots, \boldsymbol{S_m}$,

$\boldsymbol{f}(\boldsymbol{S_1} \cup \cdots \cup \boldsymbol{S_m})$ approximates $\boldsymbol{f}(\boldsymbol{V_1} \cup \cdots \cup \boldsymbol{V_m})$ by a factor $\boldsymbol{\alpha}$



Composable Core-sets

Core-sets [AHV'05]**:** a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let $\boldsymbol{f}$ be an optimization function

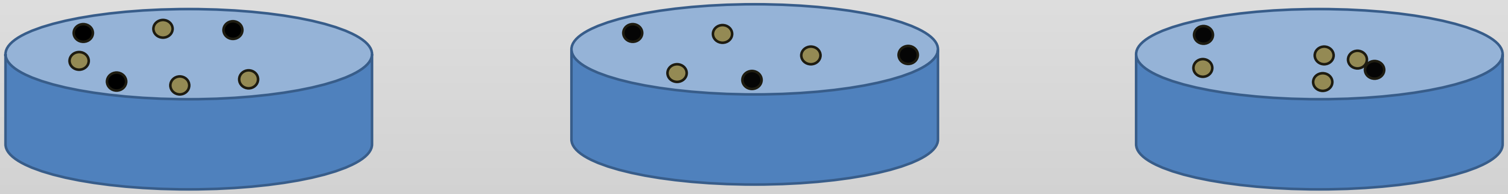
–E.g. $\boldsymbol{f}(\boldsymbol{V})$ is the solution of diversity maximization

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

i.e. for multiple data sets $\boldsymbol{V_1}, \cdots, \boldsymbol{V_m}$ and their coresets $\boldsymbol{S_1}, \cdots, \boldsymbol{S_m}$,

$\boldsymbol{f}(\boldsymbol{S_1} \cup \cdots \cup \boldsymbol{S_m})$ approximates $\boldsymbol{f}(\boldsymbol{V_1} \cup \cdots \cup \boldsymbol{V_m})$ by a factor $\boldsymbol{\alpha}$



Composable Core-sets

Core-sets [AHV'05]: a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let $\boldsymbol{f}$ be an optimization function

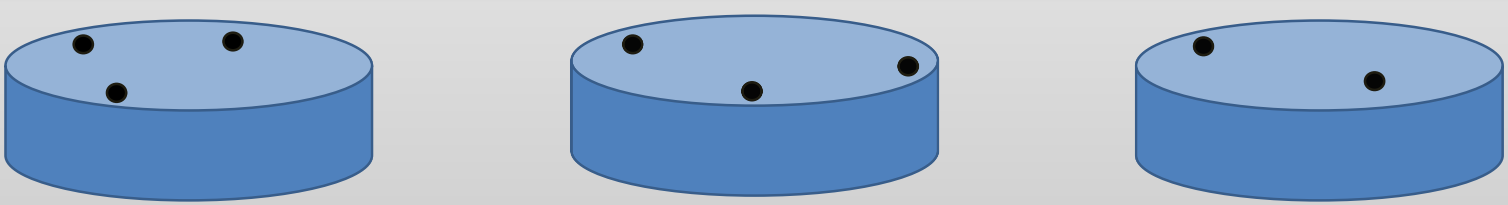
–E.g. $\boldsymbol{f}(\boldsymbol{V})$ is the solution of diversity maximization

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

i.e. for multiple data sets $\boldsymbol{V_1}, \cdots, \boldsymbol{V_m}$ and their coresets $\boldsymbol{S_1}, \cdots, \boldsymbol{S_m}$,

$\boldsymbol{f}(\boldsymbol{S_1} \cup \cdots \cup \boldsymbol{S_m})$ approximates $\boldsymbol{f}(\boldsymbol{V_1} \cup \cdots \cup \boldsymbol{V_m})$ by a factor $\boldsymbol{\alpha}$



Composable Core-sets

Core-sets [AHV'05]: a subset S of the data V that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let f be an optimization function

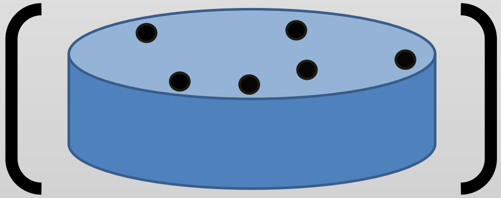
–E.g. $f(V)$ is the solution of diversity maximization

A subset $S \subset V$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

i.e. for multiple data sets $V_1, \cdots, V_m$ and their coresets $S_1, \cdots, S_m$,

$f(S_1 \cup \cdots \cup S_m)$ approximates $f(V_1 \cup \cdots \cup V_m)$ by a factor α

f 

Composable Core-sets

Core-sets [AHV'05]: a subset $\boldsymbol{S}$ of the data $\boldsymbol{V}$ that represents it well

Composable Core-sets [AAI**M**V'13 and I**M**MM'14]**:**

Let $\boldsymbol{f}$ be an optimization function

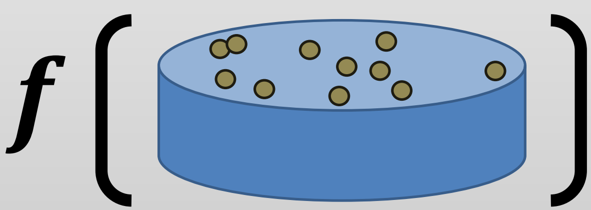
–E.g. $\boldsymbol{f}(\boldsymbol{V})$ is the solution of diversity maximization

A subset $\boldsymbol{S} \subset \boldsymbol{V}$ is called composable coreset if

–The union of coresets is an α-approximate coreset for the union

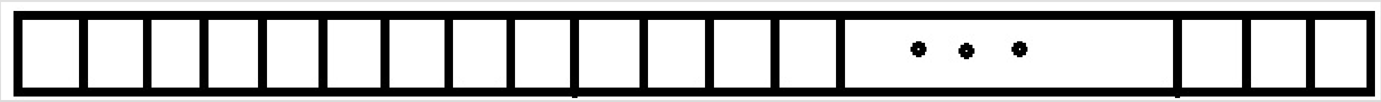
i.e. for multiple data sets $\boldsymbol{V_1}, \cdots, \boldsymbol{V_m}$ and their coresets $\boldsymbol{S_1}, \cdots, \boldsymbol{S_m}$,

$\boldsymbol{f}(\boldsymbol{S_1} \cup \cdots \cup \boldsymbol{S_m})$ approximates $\boldsymbol{f}(\boldsymbol{V_1} \cup \cdots \cup \boldsymbol{V_m})$ by a factor $\boldsymbol{\alpha}$



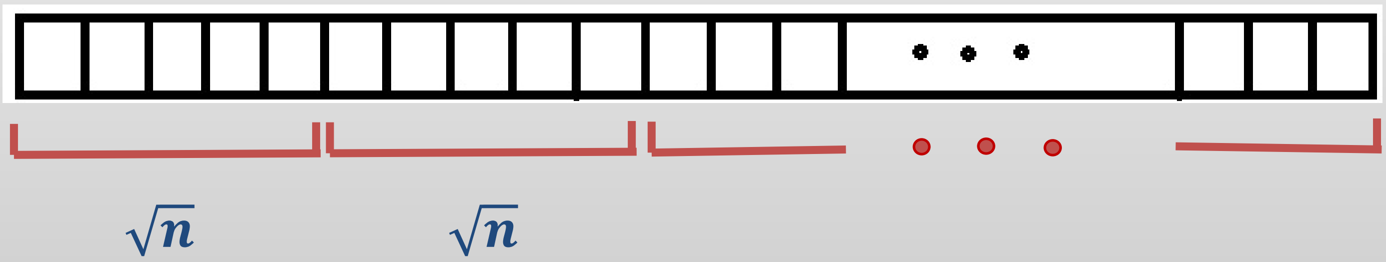
Applications: Streaming Computation

- **Streaming Computation:**
  - Processing sequence of n data elements “on the fly”
  - limited Storage



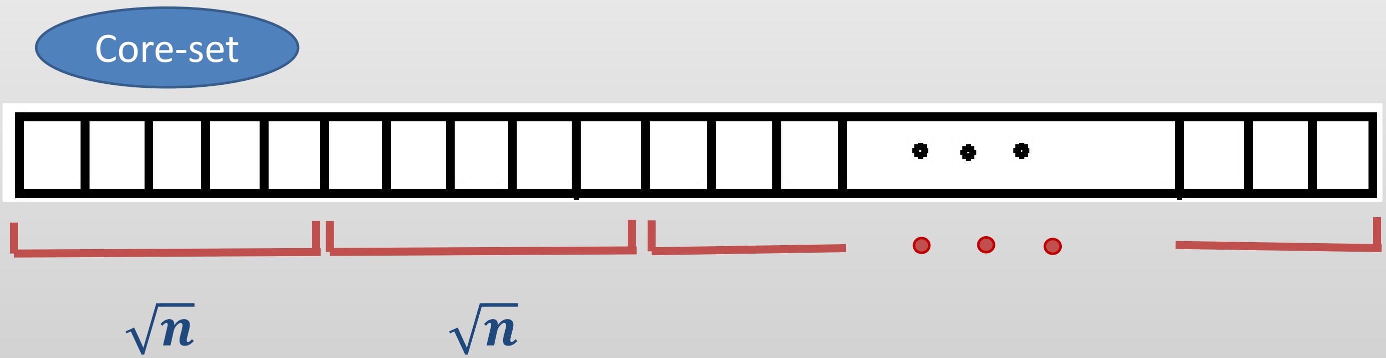
Applications: Streaming Computation

- **Streaming Computation:**
  - Processing sequence of n data elements "on the fly"
  - limited Storage
- **c-Composable Core-set of size k**
  - Chunks of size $\sqrt{n}$, thus number of chunks = $\sqrt{n}$



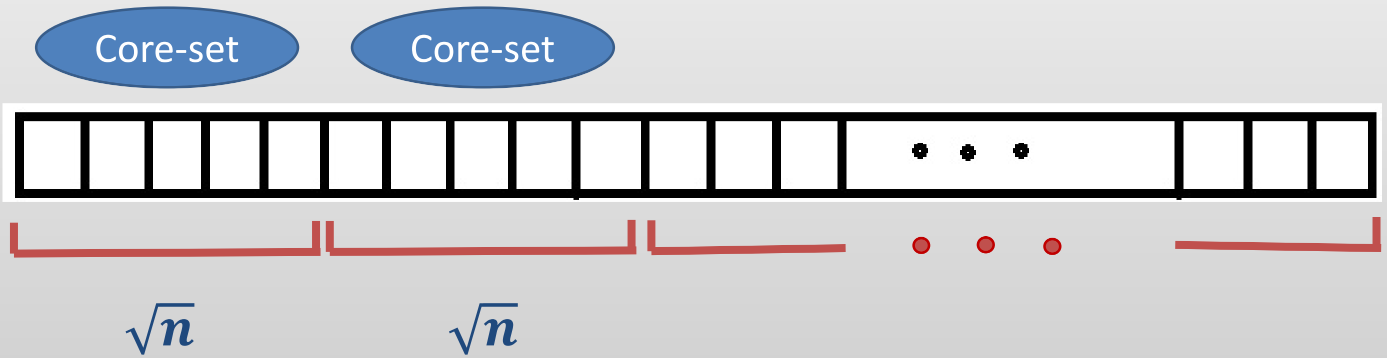
Applications: Streaming Computation

- **Streaming Computation:**
  - Processing sequence of n data elements "on the fly"
  - limited Storage
- **c-Composable Core-set of size k**
  - Chunks of size $\sqrt{n}$, thus number of chunks = $\sqrt{n}$
  - Core-set for each chunk



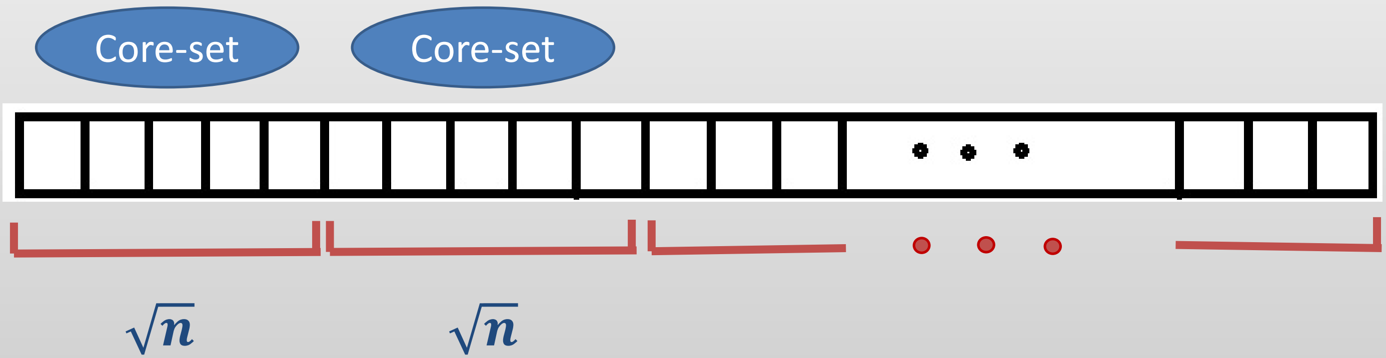
Applications: Streaming Computation

- **Streaming Computation:**
  - Processing sequence of n data elements "on the fly"
  - limited Storage
- **c-Composable Core-set of size k**
  - Chunks of size $\sqrt{n}$, thus number of chunks = $\sqrt{n}$
  - Core-set for each chunk



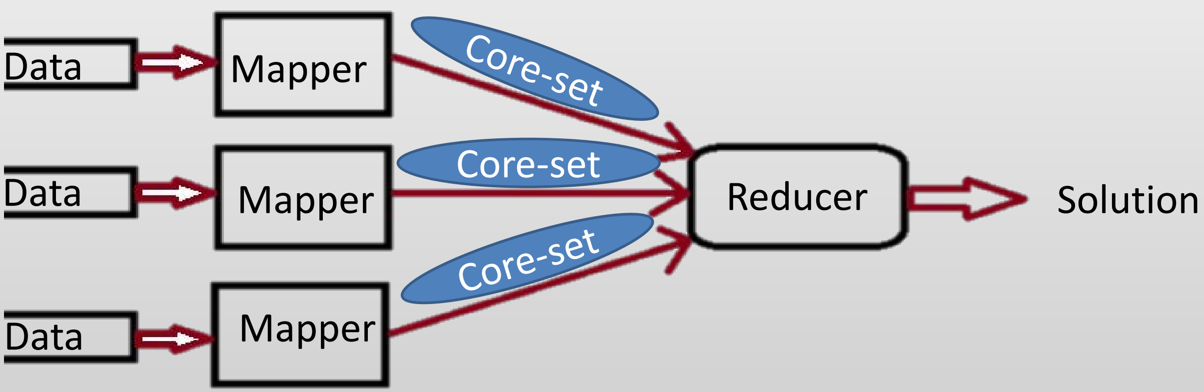
Applications: Streaming Computation

- **Streaming Computation:**
  - Processing sequence of n data elements "on the fly"
  - limited Storage
- **c-Composable Core-set of size k**
  - Chunks of size $\sqrt{n}$, thus number of chunks = $\sqrt{n}$
  - Core-set for each chunk
  - Total Space: (core-set size)$\cdot\sqrt{n}+\sqrt{n}$
  - Approximation Factor: c



Applications: Distributed Computation

- **Streaming Computation**
- **Distributed System:**
  - Each machine holds a block of data.
  - A composable core-set is computed and sent to the server
- **Map-Reduce Model:**
  - One round of Map-Reduce
  - $\sqrt{n}$ mappers each getting $\sqrt{n}$ points
  - Mapper computes a composable core-set of size k
  - Will be passed to a single reducer



Applications: Improving Runtime

- Streaming Computation
- Distributed System
- **Similar framework for improving the runtime**

Can we get a composable core-set of small size for the diversity maximization problem?

- Background on diversity maximization and how to model diversity
- Notion of composable core-sets
- **Algorithms for finding core-sets for diversity maximization**

1. Maximizing the minimum pairwise distance
2. Maximizing the volume

Results

Composable Core-sets for Diversity Maximization:

Diversity Notion	Coreset Size	Approx.	Reference	Offline
Min Pairwise Distance	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta(1)}$ **[Ravi et al 94]**
Sum of Pairwise distances	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta(1)}$ **[Hassin et al 97]**
…				**…**
Volume	$\tilde{O}(k)$	$\tilde{O}(k)^{k/2}$	[IMOR'18]	$\boldsymbol{O(c^k), \Omega(c^k)}$ **[Nik'15],[CIM'13]**

Results

Composable Core-sets for Diversity Maximization:

Diversity Notion	Coreset Size	Approx.	Reference	Offline
Min Pairwise Distance	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Ravi et al 94]
Sum of Pairwise distances	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Hassin et al 97]
…				…
Volume	$\tilde{O}(k)$	$\tilde{O}(k)^{k/2}$	[IMOR'18]	$\boldsymbol{O}(\boldsymbol{c}^{\boldsymbol{k}}), \boldsymbol{\Omega}(\boldsymbol{c}^{\boldsymbol{k}})$ [Nik'15],[CIM'13]

Results

Composable Core-sets for Diversity Maximization:

Diversity Notion	Coreset Size	Approx.	Reference	Offline
Min Pairwise Distance	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Ravi et al 94]
Sum of Pairwise distances	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Hassin et al 97]
…				…
Volume	$\tilde{O}(k)$	$\tilde{O}(k)^{k/2}$	[IMOR'18]	$\boldsymbol{O}(\boldsymbol{c}^{\boldsymbol{k}}), \boldsymbol{\Omega}(\boldsymbol{c}^{\boldsymbol{k}})$ [Nik'15],[CIM'13]

Diversity: Minimum Pairwise Distance

Joint work with S. Abbar, S. Amer-Yahia, P. Indyk, K. Varadarajan

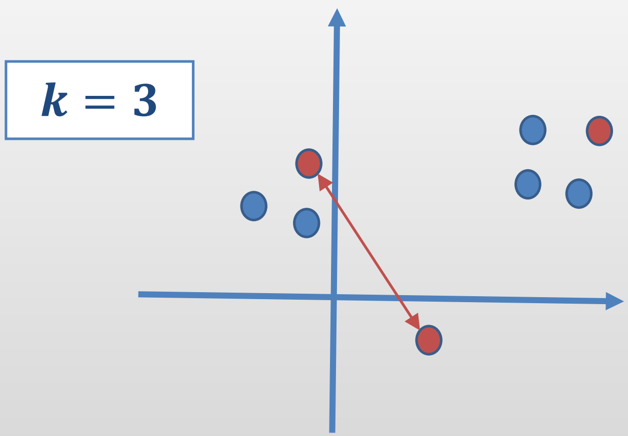
- Background on diversity maximization and how to model diversity
- Notion of composable core-sets
- Algorithms for finding core-sets for diversity maximization

1. **Maximizing the minimum pairwise distance**
2. Maximizing the volume

Minimum Pairwise Distance

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Goal: pick k points s.t. the minimum pairwise distance of the picked points is maximized.



Maximizing the minimum pairwise distance

The Greedy Algorithm produces a composable core-set of size k with approximation factor $O(1)$.

Maximizing the minimum pairwise distance

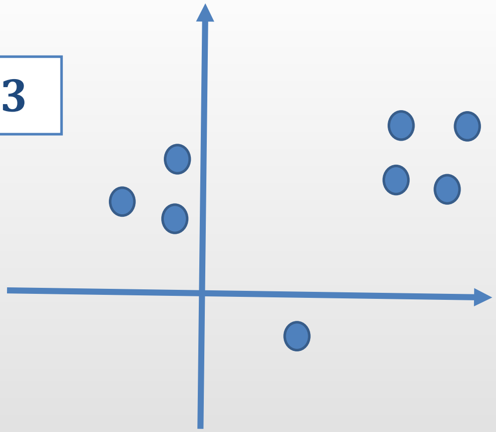
The Greedy Algorithm produces a composable core-set of size k with approximation factor $O(1)$.

Input: a set V of n points and a parameter k

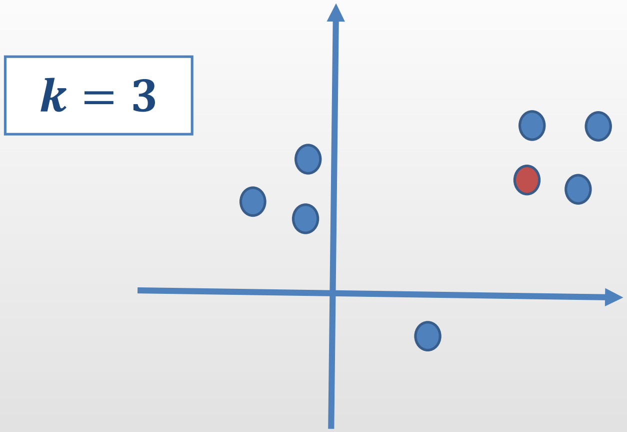
1. Start with an empty set S
2. For k iterations, add the point $p \in V \setminus S$ that is farthest away from S.

Maximizing the minimum pairwise distance

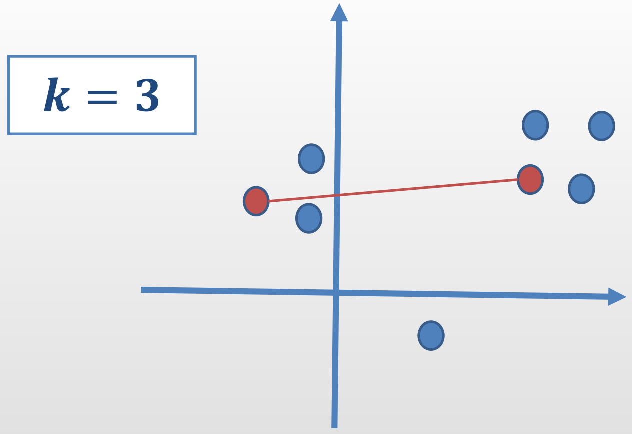
$k = 3$



Maximizing the minimum pairwise distance

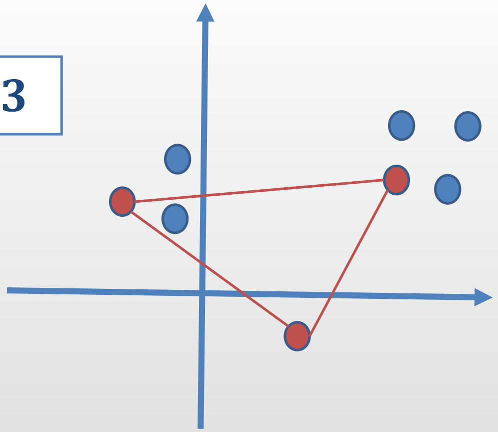


Maximizing the minimum pairwise distance



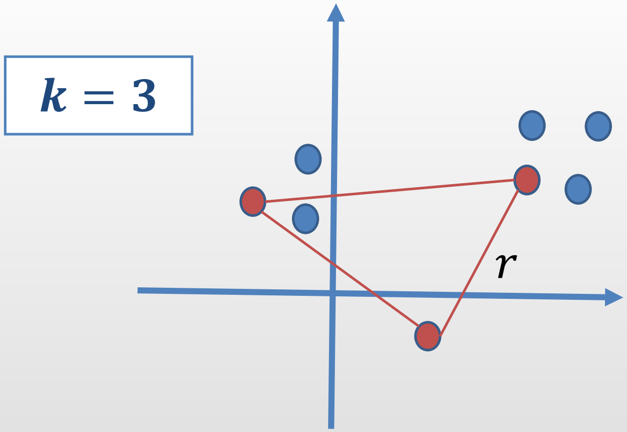
Maximizing the minimum pairwise distance

$\boldsymbol{k = 3}$



Observation

Let r be the diversity of S, i.e., $\min_{q_1, q_2 \in S} dist(q_1, q_2)$

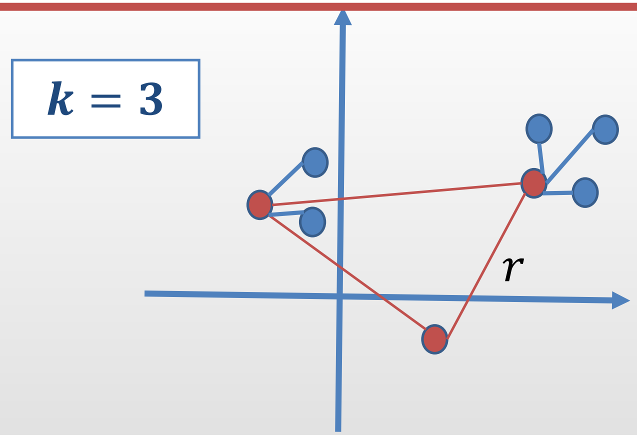


Observation

Let r be the diversity of S, i.e., $\min_{q_1,q_2 \in S} dist(q_1, q_2)$

Observation: For any point $p \in V$, we have $dist(p, S) \leq r$

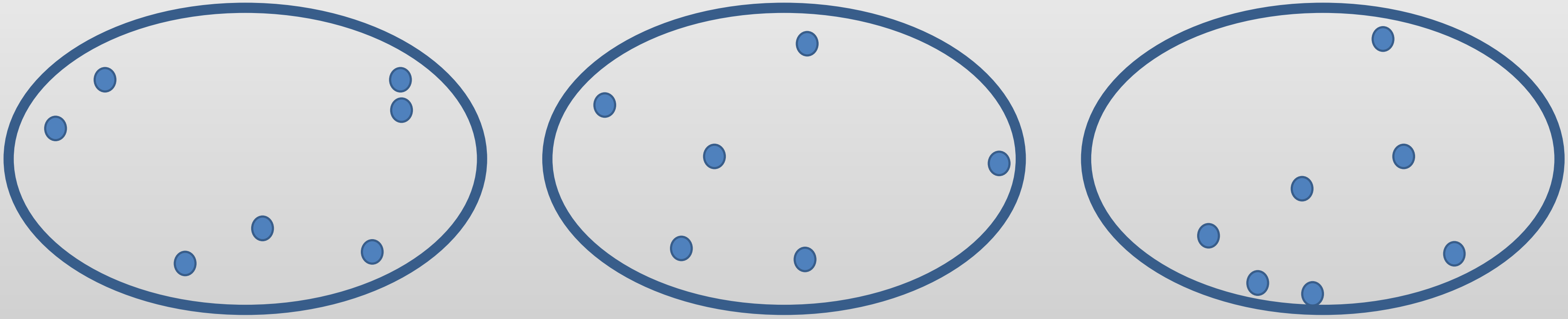
- $\exists q \in S$ such that $dist(p, q) \leq r$



Proof Idea

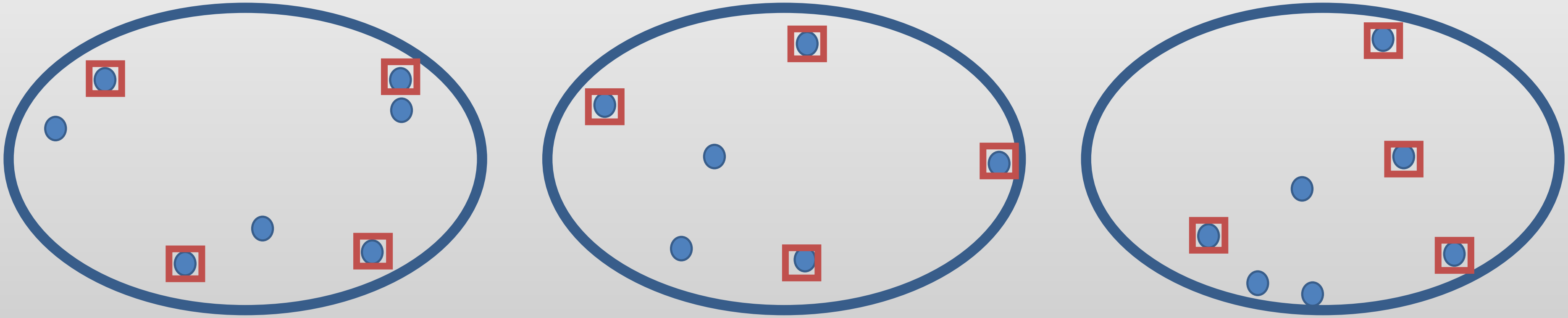
The Greedy Algorithm produces a composable core-set of size k with approximation factor $O(1)$

Let $V_1, \ldots, V_m$ be the set of points, $V = \bigcup_i V_i$



Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

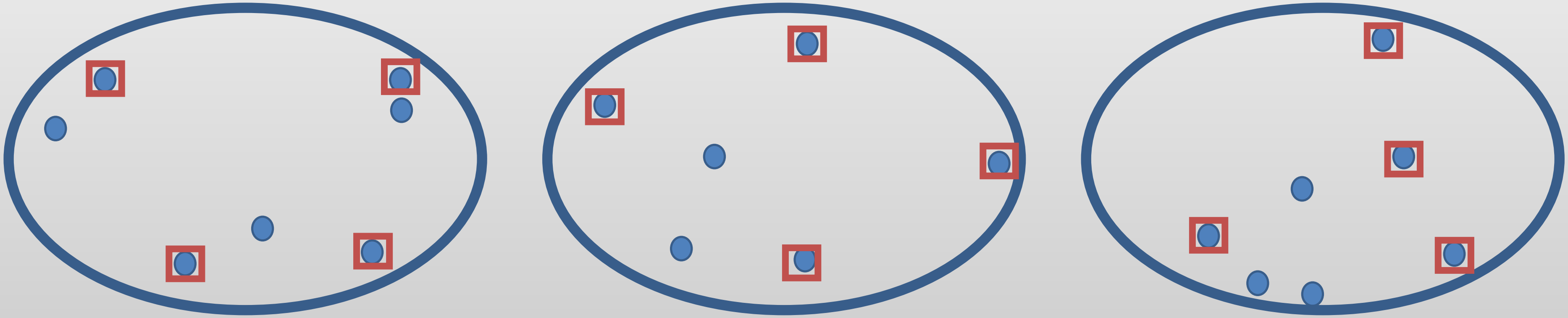
Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$



Let $V_1, \ldots, V_m$ be the set of points, $V = \bigcup_i V_i$

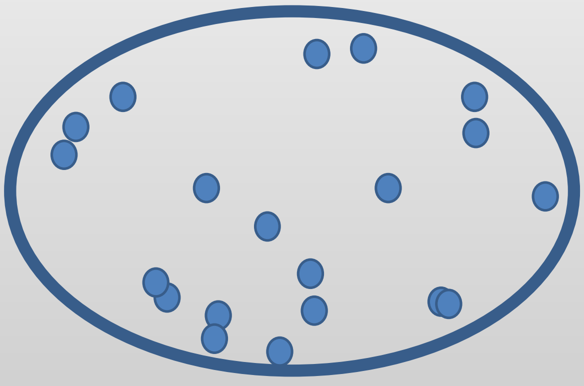
Let $S_1, \ldots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Goal: **there exists k points in S whose diversity is large in compare to the optimal set of k points in V**



Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

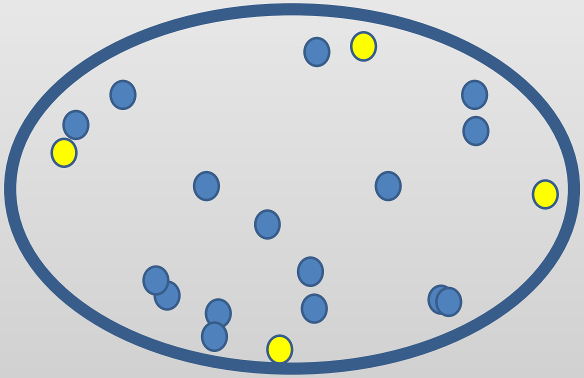
Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$



Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$

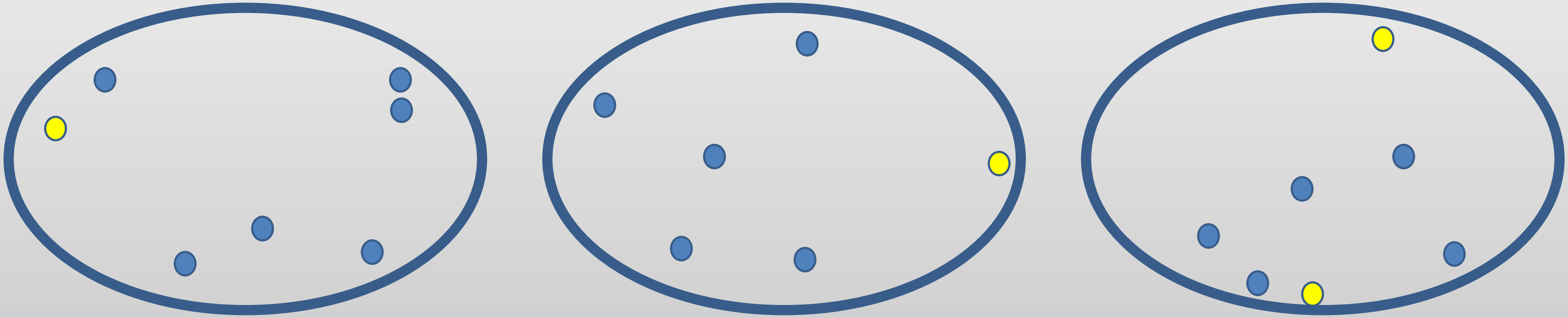
Let $Opt = \{o_1, \dots, o_k\}$ be the optimal solution



Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$

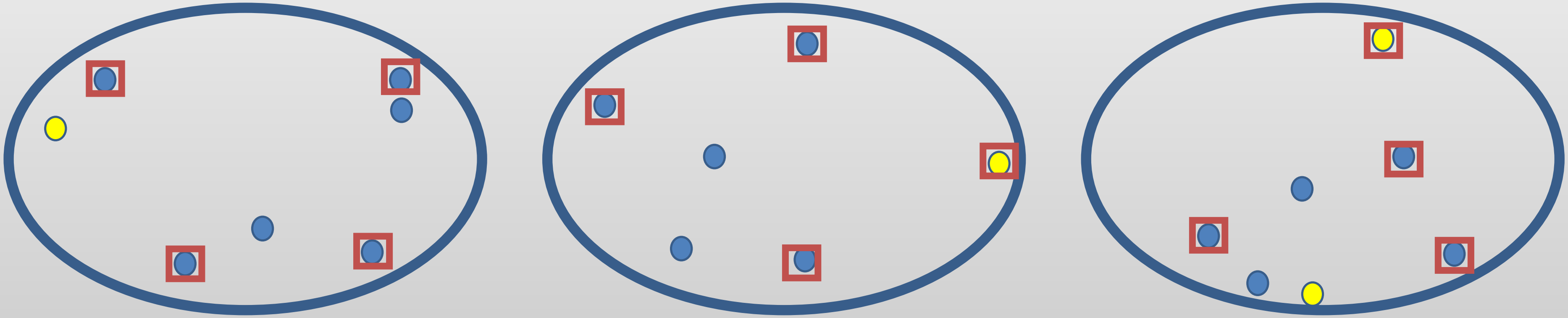
Let $Opt = \{o_1, \dots, o_k\}$ be the optimal solution



Let $V_1, \ldots, V_m$ be the set of points, $V = \bigcup_i V_i$

Let $S_1, \ldots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Let $Opt = \{o_1, \ldots, o_k\}$ be the optimal solution

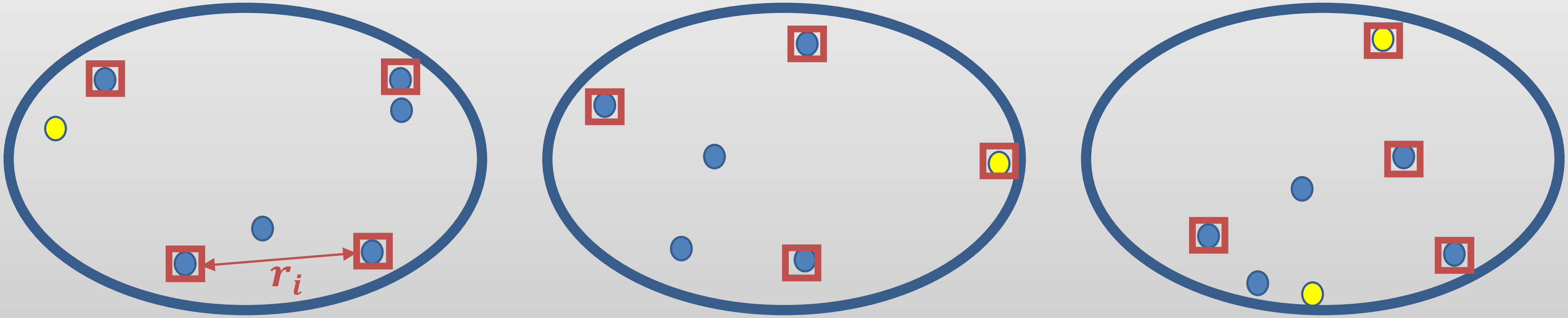


Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Let $Opt = \{o_1, \dots, o_k\}$ be the optimal solution

Case 1: one of S_i has good enough diversity (r_i is large)



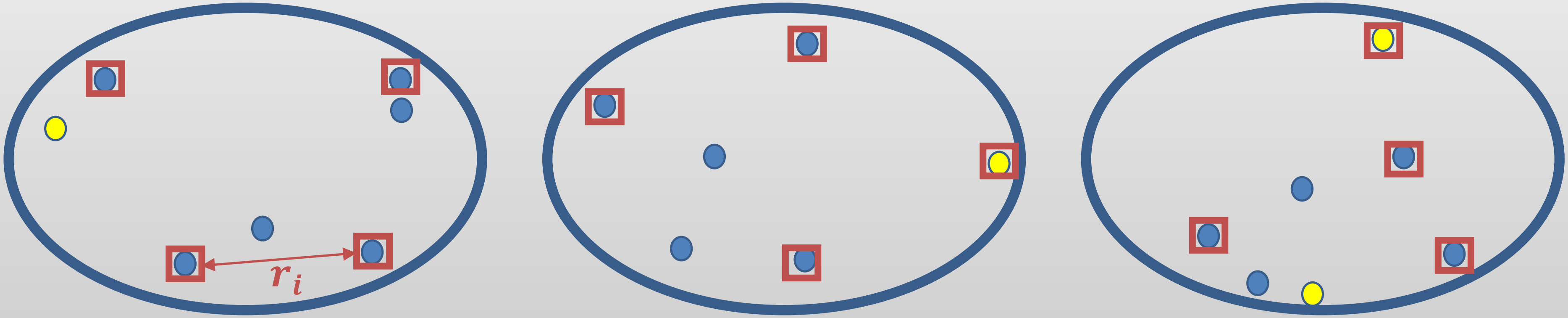
Let $V_1, \ldots, V_m$ be the set of points, $V = \cup_i V_i$

Let $S_1, \ldots, S_m$ be their core-sets, $S = \cup_i S_i$

Let $Opt = \{o_1, \ldots, o_k\}$ be the optimal solution

Case 1: one of S_i has good enough diversity (r_i is large)

➢ **S_i is a good solution**



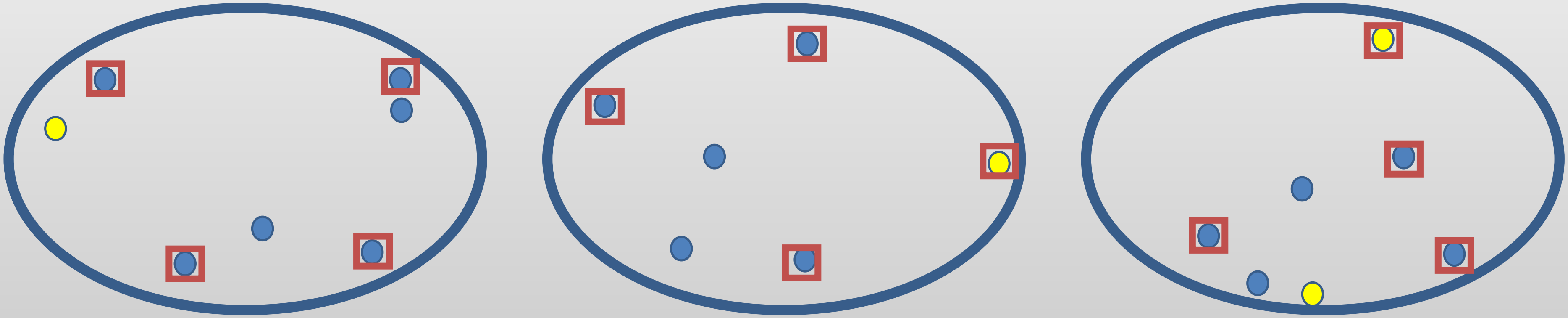
Let $V_1, \dots, V_m$ be the set of points, $V = \bigcup_i V_i$

Let $S_1, \dots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Let $Opt = \{o_1, \dots, o_k\}$ be the optimal solution

Case 1: one of S_i has good enough diversity (r_i is large)

Case 2: all r_i are small



Let $V_1, \ldots, V_m$ be the set of points, $V = \bigcup_i V_i$

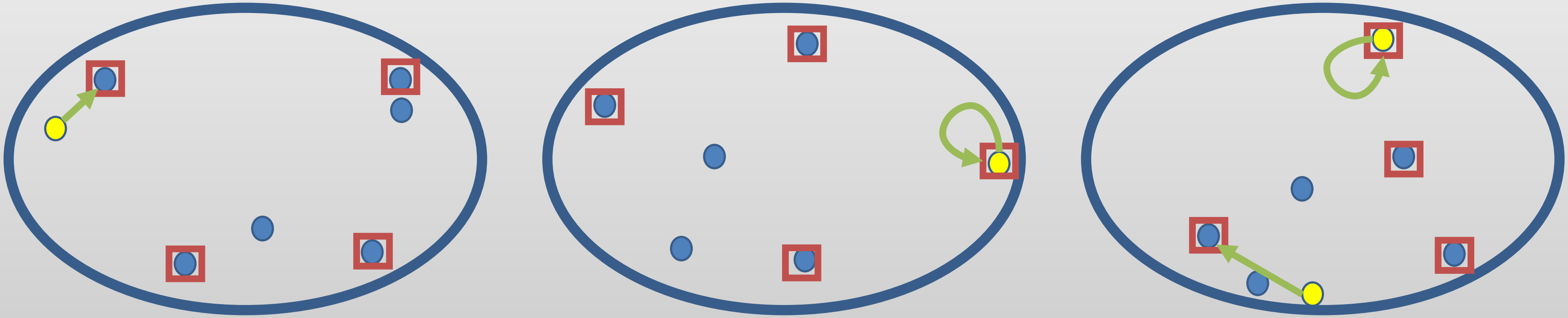
Let $S_1, \ldots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Let $Opt = \{o_1, \ldots, o_k\}$ be the optimal solution

Case 1: one of S_i has good enough diversity (r_i is large)

Case 2: all r_i are small

- Find a mapping μ from $Opt = \{o_1, \ldots, o_k\}$ to S



Let $V_1, \ldots, V_m$ be the set of points, $V = \bigcup_i V_i$

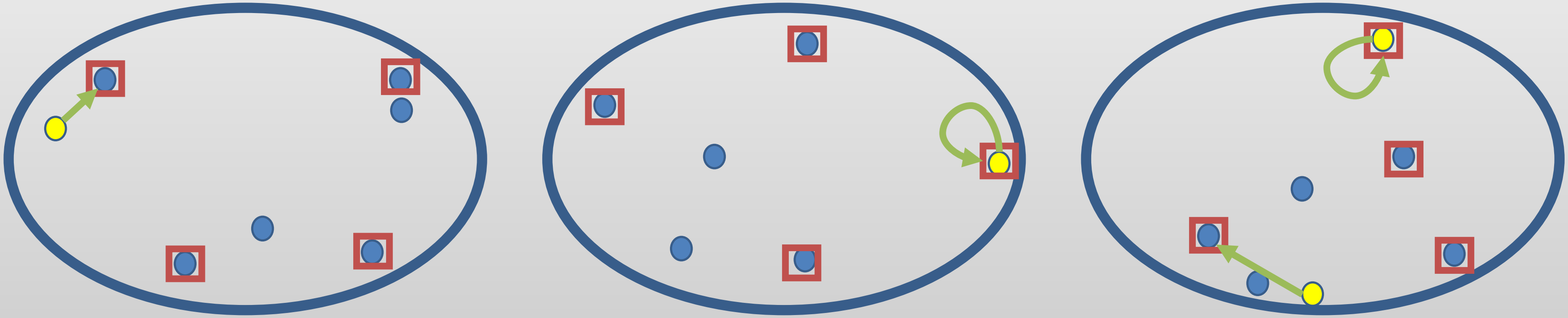
Let $S_1, \ldots, S_m$ be their core-sets, $S = \bigcup_i S_i$

Let $Opt = \{o_1, \ldots, o_k\}$ be the optimal solution

Case 1: one of S_i has good enough diversity (r_i is large)

Case 2: all r_i are small

- Find a mapping μ from $Opt = \{o_1, \ldots, o_k\}$ to S
- Replacing o_i with $\mu(o_i)$ has still large diversity



Maximizing the minimum pairwise distance

The Greedy Algorithm produces a composable core-set of size k with approximation factor $O(1)$.

Experimental Results

Real time recommendation of diverse related articles [AAIM WWW'13]

Experimental Results

Real time recommendation of diverse related articles [AAIM WWW'13]

Goal:

Recommend a few news articles based on what article the user is currently reading.

Experimental Results

Real time recommendation of diverse related articles [AAIM WWW’13]

Goal:

Recommend a few news articles based on what article the user is currently reading.

Data:

Aljazeera English opinion articles, and Reuters news articles

Experimental Results

Real time recommendation of diverse related articles [AAIM WWW'13]

Goal:

Recommend a few news articles based on what article the user is currently reading.

Data:

Aljazeera English opinion articles, and Reuters news articles

Composable Coresets

+

Nearest Neighbor Data Structure (LSH)

Experimental Results

Real time recommendation of diverse related articles [AAIM WWW'13]

Goal:

Recommend a few news articles based on what article the user is currently reading.

Data:

Aljazeera English opinion articles, and Reuters news articles

Results:

✓ The algorithm works well in practice, e.g.,

- In compare to k-nearest neighbor retrieval, we gain ~37% in diversity while losing only ~5% on relevance.
- Adding the coresets to the LSH improves the retrieve time by ~20x, with no meaningful loss on diversity.

Results

Composable Core-sets for Diversity Maximization:

Diversity Notion	Coreset Size	Approx.	Reference	Offline
Min Pairwise Distance	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Ravi et al 94]
Sum of Pairwise distances	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta}(\mathbf{1})$ [Hassin et al 97]
…				…
Volume	$\tilde{O}(k)$	$\tilde{O}(k)^{k/2}$	[IMOR'18]	$\boldsymbol{O}(\boldsymbol{c}^{\boldsymbol{k}}), \boldsymbol{\Omega}(\boldsymbol{c}^{\boldsymbol{k}})$ [Nik'15],[CIM'13]

Diversity: Volume of the points

Joint work with P. Indyk, S. Oveis Gharan, A. Rezaei

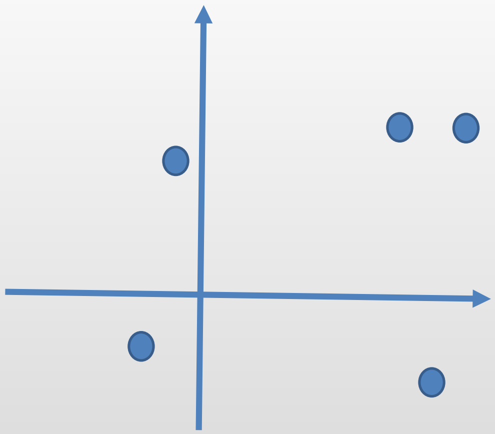
- Background on diversity maximization and how to model diversity
- Notion of composable core-sets
- Algorithms for finding core-sets for diversity maximization

1. Maximizing the minimum pairwise distance
2. **Maximizing the volume**

Volume (Determinant) Maximization Problem

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

$\boxed{\boldsymbol{k = 2}}$



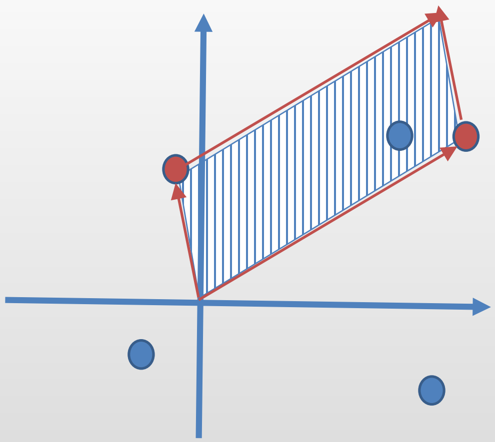
Volume (Determinant) Maximization Problem

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Output: a subset $S \subset V$ of size k with the maximum volume

- Parallelepiped spanned by the points in S

$\boldsymbol{k = 2}$



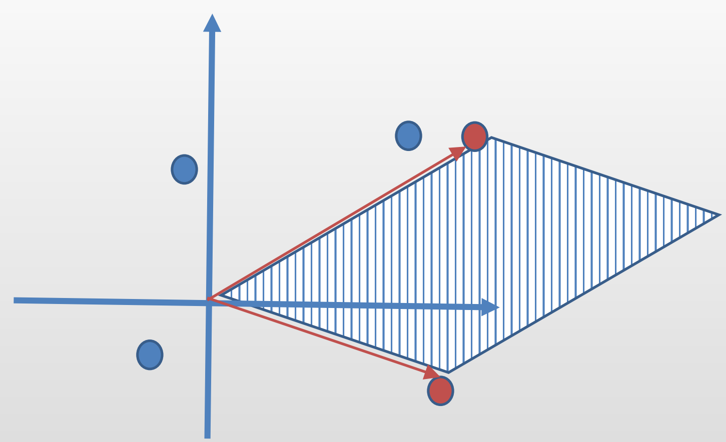
Volume (Determinant) Maximization Problem

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Output: a subset $S \subset V$ of size k with the maximum volume

- Parallelepiped spanned by the points in S

$\boxed{\boldsymbol{k = 2}}$



What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]

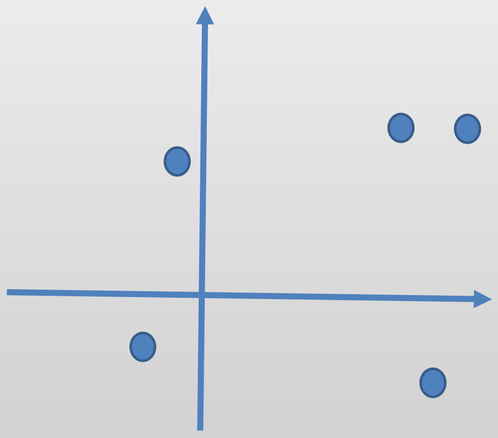
What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]

What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

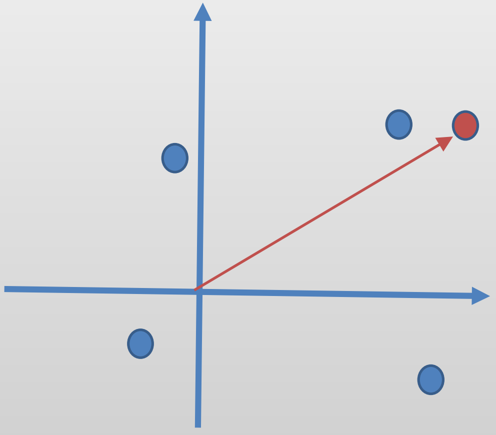
$\boxed{\boldsymbol{k = 2}}$



What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

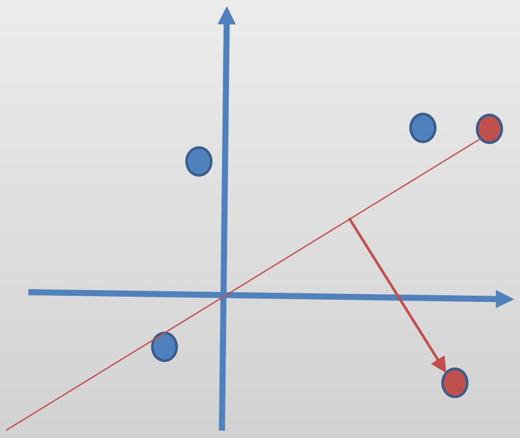
$\boldsymbol{k = 2}$



What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

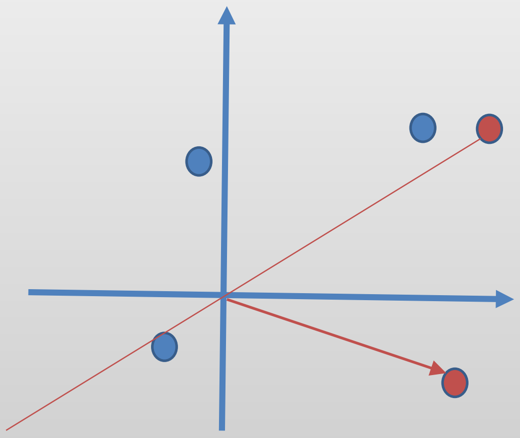
$\boldsymbol{k = 2}$



What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

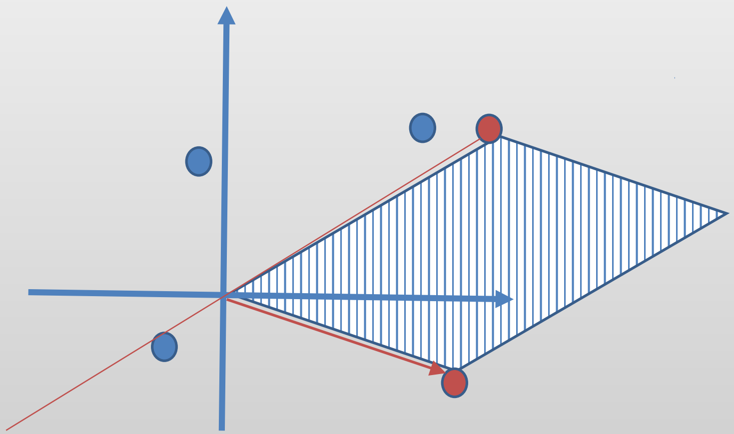
$\boxed{k = 2}$



What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

$\boldsymbol{k = 2}$

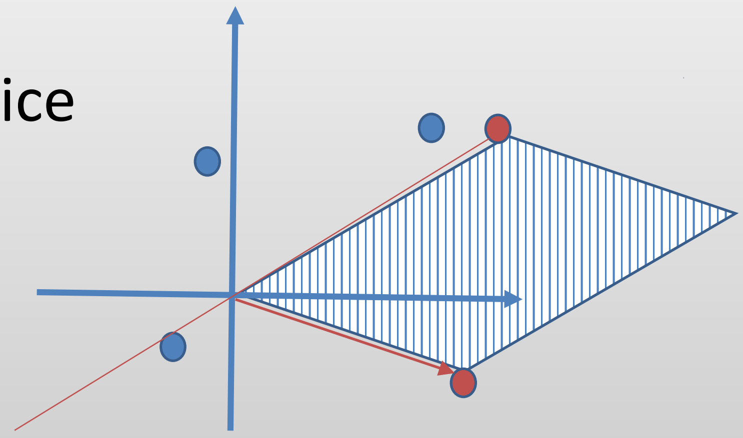


What is known?

- Hard to approximate within a factor of 2^{ck} [CMI'13]
- Best algorithm: $2^{O(k)}$-approximation [Nik'15]
- **Greedy** is a popular algorithm: achieves approximation factor $k!$
  - $U \leftarrow \emptyset$
  - For k iterations,
    - Add to U the farthest point from the subspace spanned by U

- Greedy performs very well in practice

$\boldsymbol{k = 2}$



Determinantal Point Processes (DPP)

DPP: Popular probabilistic model, where given a set of vectors V, **samples** any k-subset S with probability proportional to this volume.

- Maximum a posteriori (MAP) decoding is volume maximization

Determinant (Volume) Maximization Problem

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Output: a subset $S \subset V$ of size k with the maximum volume

- Parallelepiped spanned by the points in S

$$\begin{bmatrix} v_1 \; v_2 \dots v_n \end{bmatrix}$$

$$\mathbb{V}$$

Equivalent Formulation:

Reuse V to denote the matrix where its columns are the vectors in V

Determinant (Volume) Maximization Problem

Input: a set of n vectors $\mathrm{V} \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Output: a subset $S \subset V$ of size k with the maximum volume

- Parallelepiped spanned by the points in S

$$\underset{V^T}{\begin{bmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{bmatrix}} \times \underset{V}{\begin{bmatrix} v_1 \; v_2 \dots v_n \end{bmatrix}} = \underset{M}{\begin{array}{cc} & j \\ i & \begin{bmatrix} & & \\ & v_i \cdot v_j & \\ & & \end{bmatrix} \end{array}}$$

Equivalent Formulation:

Reuse V to denote the matrix where its columns are the vectors in V

- Let M be the gram matrix $V^T V$

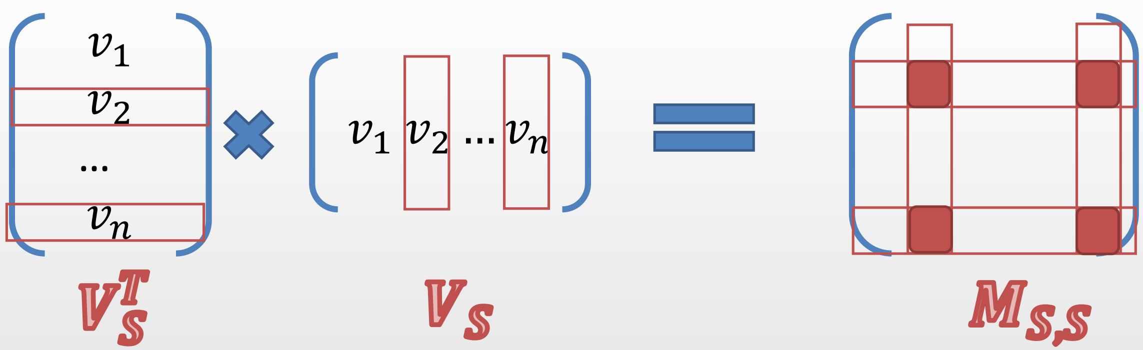
$$\boxed{M_{i,j} = v_i \cdot v_j}$$

Determinant (Volume) Maximization Problem

Input: a set of n vectors $V \subset \mathbb{R}^d$ and a parameter $k \leq d$,

Output: a subset $S \subset V$ of size k with the maximum volume

- Parallelepiped spanned by the points in S



Equivalent Formulation:

Reuse V to denote the matrix where its columns are the vectors in V

- Let M be the gram matrix $V^T V$
- Choose S such that $\det(M_{S,S})$ is maximized

$$\boxed{M_{i,j} = v_i \cdot v_j}$$

$$\boxed{\det(M_{S,S}) = Vol(S)^2}$$

Result: optimal composable core-set

Composable Core-sets for Volume Maximization:

Algorithm:

There exists a polynomial time algorithm for computing an $\tilde{O}(k)^{\frac{k}{2}}$ -composable core-set of size $\tilde{O}(k)$ for the volume maximization problem.

Lower bound:

Any composable core-set of size $k^{O(1)}$ for the volume maximization problem must have an approximation factor of $\Omega(k)^{\frac{k}{2}(1-o(1))}$.

Result: optimal composable core-set

Composable Core-sets for Volume Maximization:

Algorithm:

There exists a polynomial time algorithm for computing an $\tilde{O}(k)^{\frac{k}{2}}$ -composable core-set of size $\tilde{O}(k)$ for the volume maximization problem.

Lower bound:

Any composable core-set of size $k^{O(1)}$ for the volume maximization problem must have an approximation factor of $\Omega(k)^{\frac{k}{2}(1-o(1))}$.

- Note the gap with the approximation factor of the best offline algorithm: $2^{O(k)}$

In this Talk

Composable Core-sets for Volume Maximization:

Algorithm:
There exists a polynomial time algorithm for computing an $\boxed{\tilde{O}(k)^k}$ - composable core-set of size $\tilde{O}(k)$ for the volume maximization problem.

Lower bound:
Any composable core-set of size $k^{O(1)}$ for the volume maximization problem must have an approximation factor of $\Omega(k)^{\frac{k}{2}(1-o(1))}$.

Local Search Algorithm

The Local Search Algorithm produces a composable core-set of size k with approximation factor $O(k)^k$

Local Search Algorithm

The Local Search Algorithm produces a composable core-set of size k with approximation factor $O(k)^k$

In compare to the optimal core-set algorithm

- Approximation $\mathrm{O}(k)^k$ as opposed to $O(k \log k)^{\frac{k}{2}}$

Local Search Algorithm

The Local Search Algorithm produces a composable core-set of size k with approximation factor $O(k)^k$

In compare to the optimal core-set algorithm

- Approximation $\mathrm{O}(k)^k$ as opposed to $O(k\log k)^{\frac{k}{2}}$
- Size k as opposed to $O(k\log k)$
- Simpler to implement (similar to Greedy)
- Better performance in practice

The Local Search Algorithm

Input: a set V of n points and a parameter k

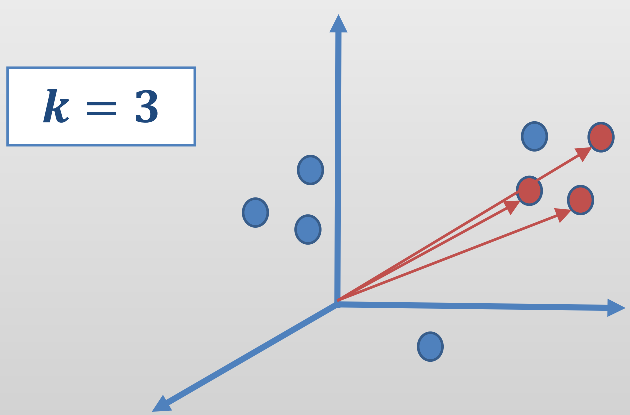
1. Start with an arbitrary subset of k points $S \subseteq V$
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$

2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

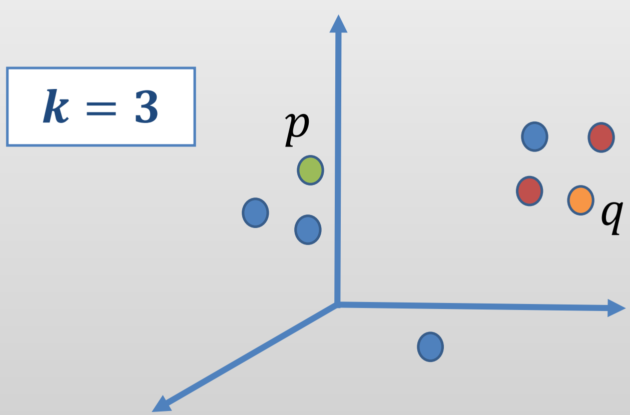


The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$

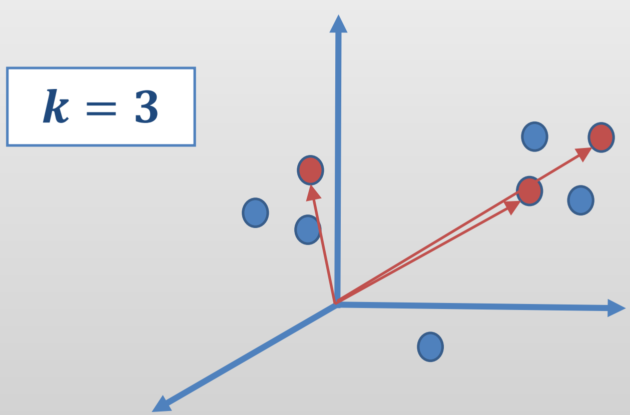
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$



The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

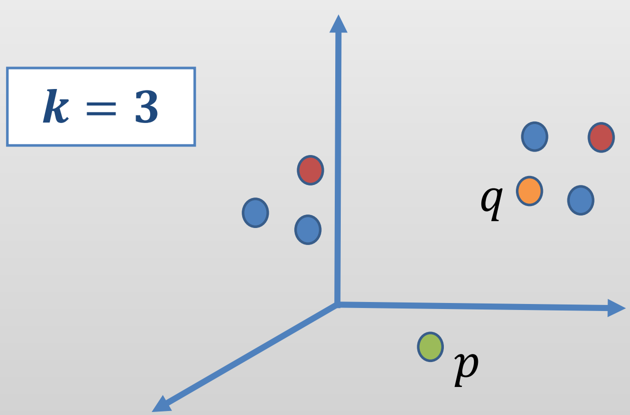


The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$

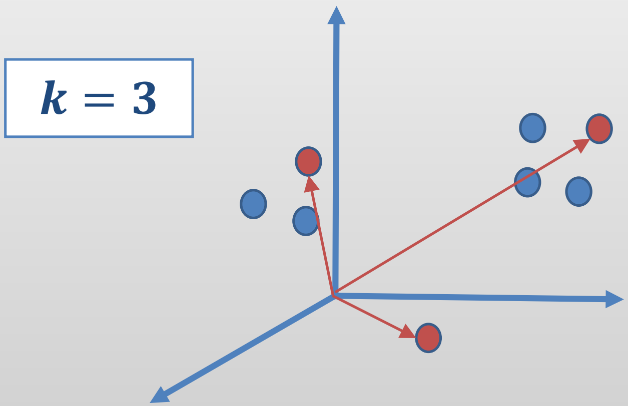
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$



The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q increases the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

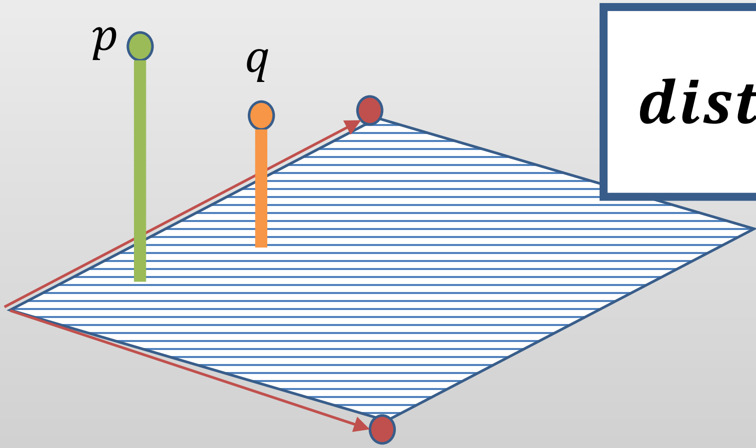


The Local Search Algorithm

Input: a set V of n points and a parameter k

1. Start with an arbitrary subset of k points $S \subseteq V$
2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q **increases the volume**, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

$$\boldsymbol{dist}\left(\boldsymbol{p}, \boldsymbol{H}_{\boldsymbol{S}\setminus\{\boldsymbol{q}\}}\right) > \boldsymbol{dist}\left(\boldsymbol{q}, \boldsymbol{H}_{\boldsymbol{S}\setminus\{\boldsymbol{q}\}}\right)$$



To bound the run time

Start with a crude approximation (Greedy algorithm)

Input: a set V of n points

1. Start with an **arbitrary** subset of k points $S \subseteq V$

2. While there exists a point $p \in V \setminus S$ and $q \in S$ s.t. replacing p with q **increases** the volume, then swap them, i.e., $S = S \cup \{p\} \setminus \{q\}$

If it increases by at least a factor of $(1 + \epsilon)$

Local Search algorithm preserves maximum distances to "all" subspaces of dimension $\boldsymbol{k-1}$

- V is the point set
- $S = LS(V)$ is the core-set produced by the local search algorithm.

Local Search algorithm preserves maximum distances to "all" subspaces of dimension $\boldsymbol{k-1}$

- V is the point set
- $S = LS(V)$ is the core-set produced by the local search algorithm.

Theorem:

For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{q \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{q \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.

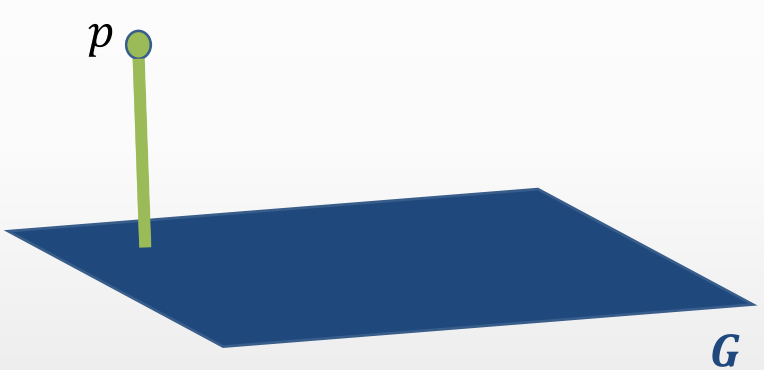
- Let $p \in V$ be a point

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.

- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.

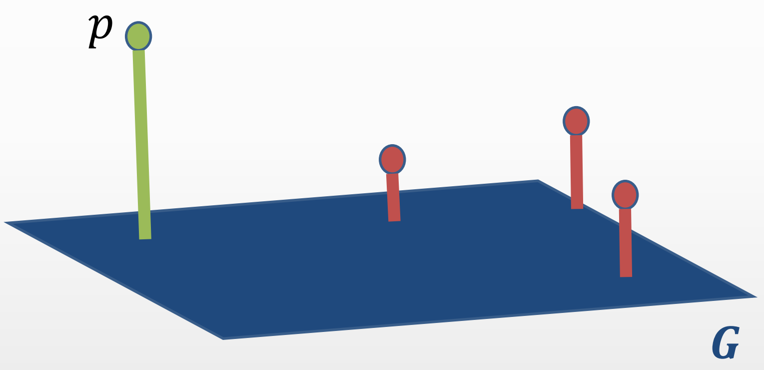


Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.

- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $\; d(q, G) \leq x$



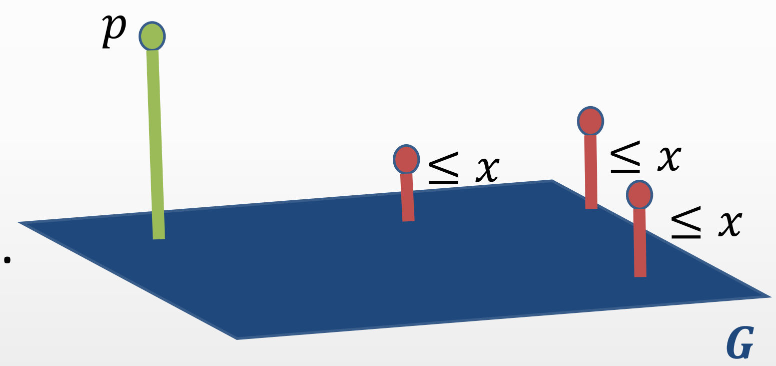
Theorem:

For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s\in S} dist(q,G) \geq \frac{1}{2k} \cdot \max_{p\in V} dist(p,G)$$

Proof.

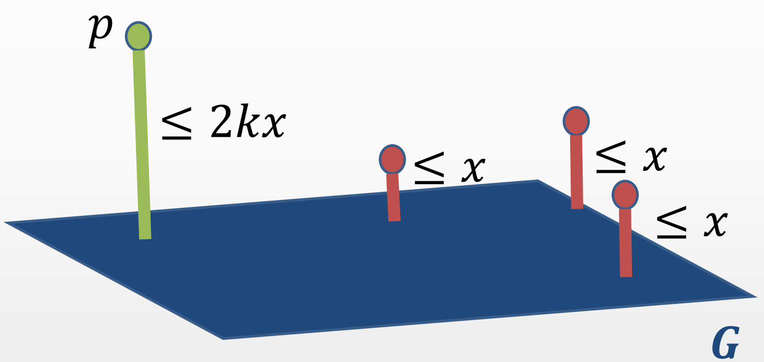
- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q,G) \leq x$



Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.

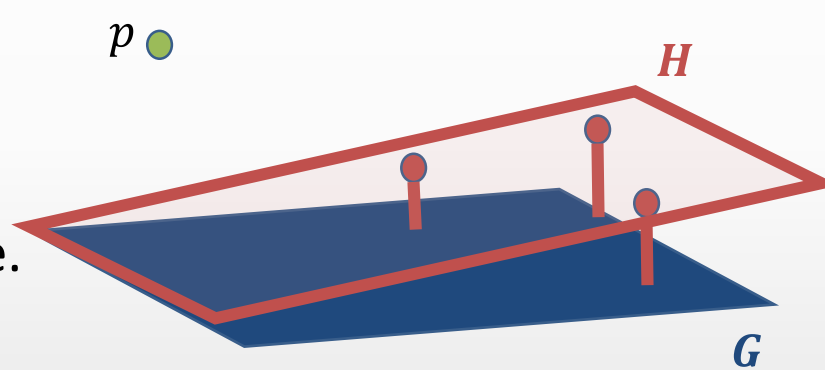


- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q, G) \leq x$
- **Goal:** $\boxed{\boldsymbol{d(p, G) \leq 2kx}}$

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s\in S} dist(q,G) \geq \frac{1}{2k} \cdot \max_{p\in V} dist(p,G)$$

Proof.

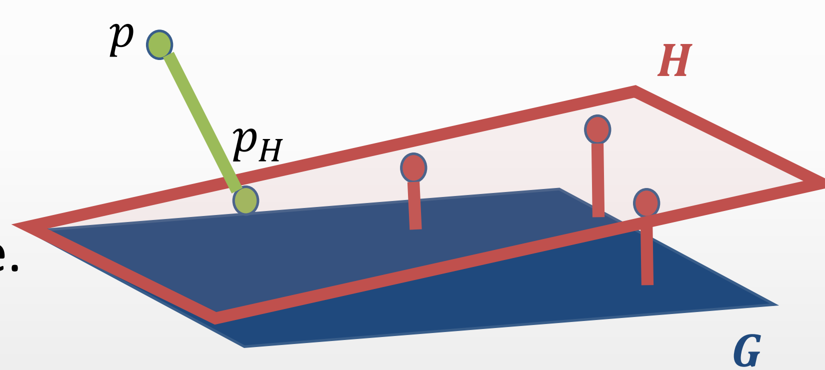


- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q,G) \leq x$
- **Goal:** $\boxed{\boldsymbol{d(p,G) \leq 2kx}}$
- $H := H_S$ be the subspace passing through S

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.



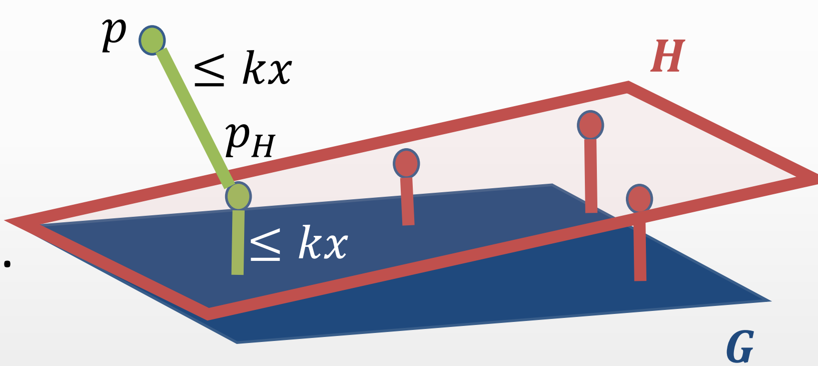
- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q, G) \leq x$
- **Goal:** $\boxed{\boldsymbol{d(p, G) \leq 2kx}}$
- $H := H_S$ be the subspace passing through S
- Let p_H be the projection of p onto G

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.

- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q, G) \leq x$
- **Goal:** $\boxed{\boldsymbol{d(p, G) \leq 2kx}}$
- $H := H_S$ be the subspace passing through S
- Let p_H be the projection of p onto G



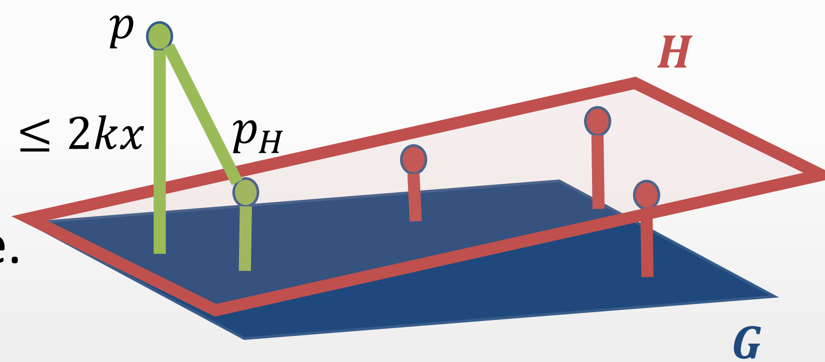
Lemma 1: $\boldsymbol{d(p, p_H) \leq kx}$

Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Theorem:
For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{s \in S} dist(q, G) \geq \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Proof.



- Let $p \in V$ be a point
- Let G be a $(k-1)$-dimensional subspace.
- Assume for any $q \in S$, $d(q, G) \leq x$
- **Goal:** $\boxed{\boldsymbol{d(p, G) \leq 2kx}}$
- $H := H_S$ be the subspace passing through S
- Let p_H be the projection of p onto G

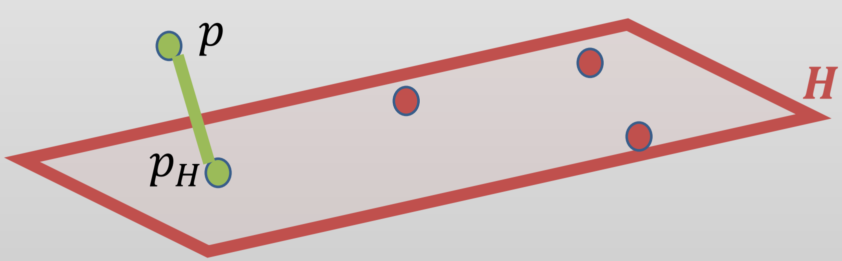
Lemma 1: $\boldsymbol{d(p, p_H) \leq kx}$

Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

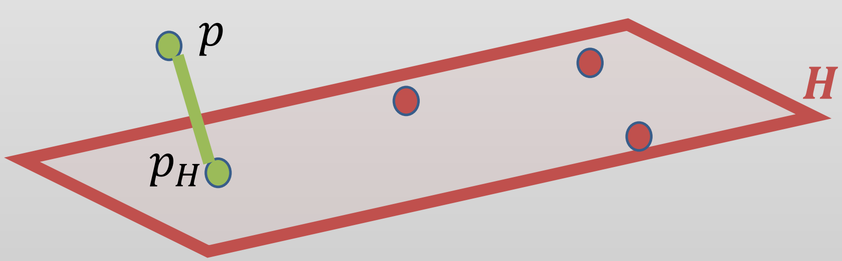


Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:
We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$



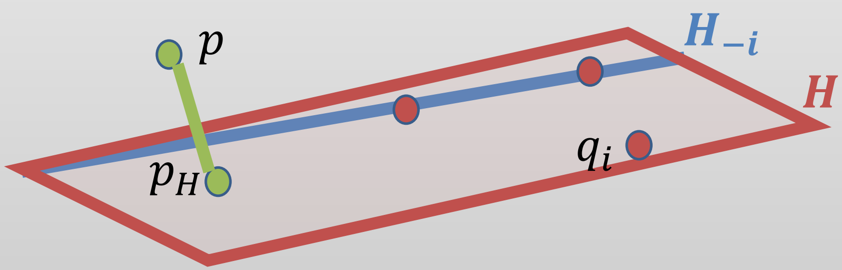
Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$
- Let $\boldsymbol{H_{-i} := H_{S \setminus \{q_i\}}}$



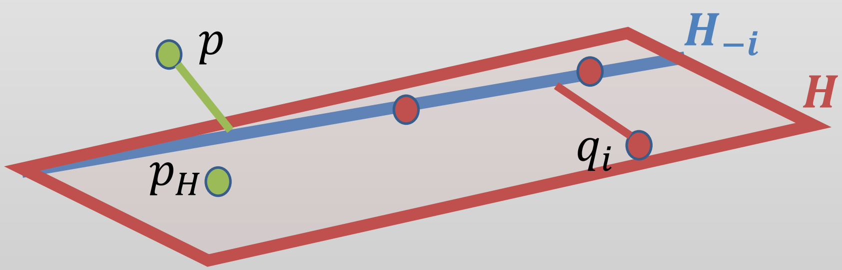
Lemma 2: $\boldsymbol{d(p_H, G) \le kx}$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \le 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$
- Let $\boldsymbol{H_{-i} := H_{S \setminus \{q_i\}}}$
- Since we did not choose p in LS, $\boldsymbol{dist(p, H_{-i}) \le dist(q_i, H_{-i})}$



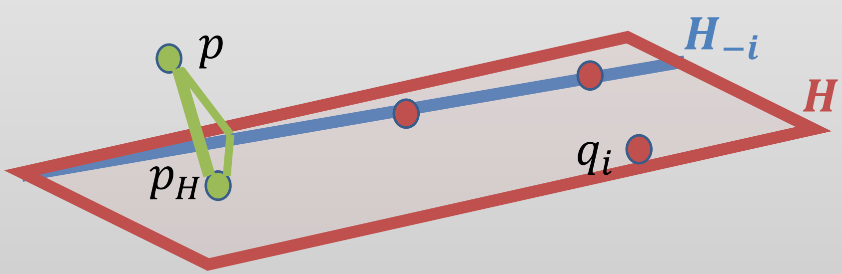
Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$
- Let $\boldsymbol{H_{-i} := H_{S \setminus \{q_i\}}}$
- Since we did not choose p in LS, $\boldsymbol{dist(p, H_{-i}) \leq dist(q_i, H_{-i})}$
- Since p_H is a projection of p onto H, $\boldsymbol{dist(p_H, H_{-i}) \leq dist(p, H_{-i})}$
-

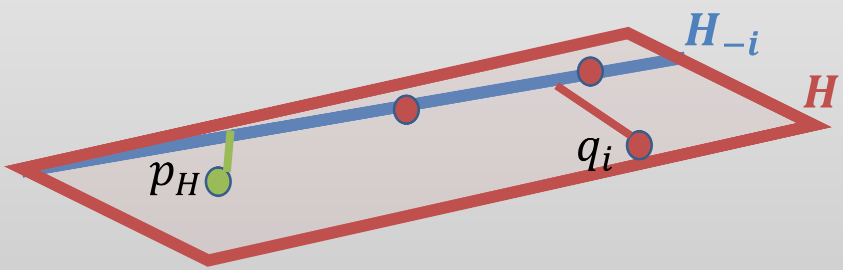


Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:
We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$
- Let $\boldsymbol{H_{-i} := H_{S \setminus \{q_i\}}}$
- Since we did not choose p in LS, $\boldsymbol{dist(p, H_{-i}) \leq dist(q_i, H_{-i})}$
- Since p_H is a projection of p onto H, $\boldsymbol{dist(p_H, H_{-i}) \leq dist(p, H_{-i})}$
- Thus $\boldsymbol{dist(p_H, H_{-i}) \leq dist(q_i, H_{-i})}$



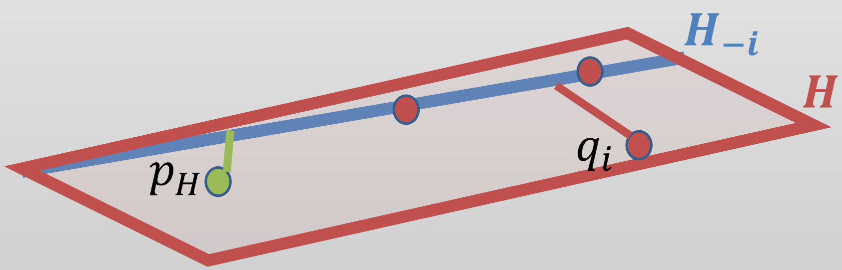
Lemma 2: $\boldsymbol{d(p_H, G) \leq kx}$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Proof.

- Since H has dimension k, we can write $\boldsymbol{p_H = \sum_{i=1}^{k} \alpha_i q_i}$
- Let $\boldsymbol{H_{-i} \coloneqq H_{S\setminus\{q_i\}}}$
- Since we did not choose p in LS, $\boldsymbol{dist(p, H_{-i}) \leq dist(q_i, H_{-i})}$
- Since p_H is a projection of p onto H, $\boldsymbol{dist(p_H, H_{-i}) \leq dist(p, H_{-i})}$
- Thus $\boldsymbol{dist(p_H, H_{-i}) \leq dist(q_i, H_{-i})}$
- **Thus** $\boldsymbol{|\alpha_i| \leq 1}$



Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Assumption: $dist(q_i, G) \leq x$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Assumption: $dist(q_i, G) \leq x$

Lemma2: $dist(p_H, G) \leq \sum_{i=1}^{k} \alpha_i dist(q_i, G) \leq k \cdot x \leq kx$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \le 1$

Assumption: $dist(q_i, G) \le x$

Lemma2: $dist(p_H, G) \le \sum_{i=1}^{k} \alpha_i dist(q_i, G) \le k \cdot x \le kx$

Lemma 1: $d(p, p_H) \le kx$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \leq 1$

Assumption: $dist(q_i, G) \leq x$

Lemma2: $dist(p_H, G) \leq \sum_{i=1}^{k} \alpha_i dist(q_i, G) \leq k \cdot x \leq kx$

Lemma 1: $d(p, p_H) \leq kx$

Goal: $d(p, G) \leq 2kx$

Claim:

We can write $p_H = \sum_{i=1}^{k} \alpha_i q_i$ s.t. all $|\alpha_i| \le 1$

Assumption: $dist(q_i, G) \le x$

Lemma2: $dist(p_H, G) \le \sum_{i=1}^{k} \alpha_i dist(q_i, G) \le k \cdot x \le kx$

Lemma 1: $d(p, p_H) \le kx$

Goal: $d(p, G) \le 2kx$

Theorem:

For any $(k-1)$-dimensional subspace G, the maximum distance of the point set to G is approximately preserved

$$\max_{q \in S} dist(q, G) \ge \frac{1}{2k} \cdot \max_{p \in V} dist(p, G)$$

Local Search produces a core-set for volume maximization

Let $V = \bigcup_i V_i$ be the union of the point sets

Let $S = \bigcup_i S_i$ be the union of core-sets

Local Search produces a core-set for volume maximization

Let $V = \bigcup_i V_i$ be the union of the point sets

Let $S = \bigcup_i S_i$ be the union of core-sets

Let $Opt = \{o_1, \ldots, o_k\} \subset V$ be the optimal subset of points maximizing the volume

Local Search produces a core-set for volume maximization

Let $V = \bigcup_i V_i$ be the union of the point sets

Let $S = \bigcup_i S_i$ be the union of core-sets

Let $Opt = \{o_1, \dots, o_k\} \subset V$ be the optimal subset of points maximizing the volume

$Sol \leftarrow Opt$

For $i = 1$ to k

- Let $q_i \in S$ be the point that is farthest away from $H_{Sol \setminus \{o_i\}}$
- $Sol \leftarrow Sol \cup \{q_i\} \setminus \{o_i\}$

Local Search produces a core-set for volume maximization

Let $V = \bigcup_i V_i$ be the union of the point sets

Let $S = \bigcup_i S_i$ be the union of core-sets

Let $Opt = \{o_1, \dots, o_k\} \subset V$ be the optimal subset of points maximizing the volume

$Sol \leftarrow Opt$

For $i = 1\ to\ k$

- Let $q_i \in S$ be the point that is fa
- $Sol \leftarrow Sol \cup \{q_i\} \setminus \{o_i\}$

Since local search preserve maximum distances to subspaces

➢ Lose a factor of at most $2k$ at each iteration

Local Search produces a core-set for volume maximization

Let $V = \bigcup_i V_i$ be the union of the point sets

Let $S = \bigcup_i S_i$ be the union of core-sets

Let $Opt = \{o_1, \ldots, o_k\} \subset V$ be the optimal subset of points maximizing the volume

$Sol \leftarrow Opt$

For $i = 1$ to k

- Let $q_i \in S$ be the point that is farthest away from $H_{Sol \setminus \{o_i\}}$
- $Sol \leftarrow Sol \cup \{q_i\} \setminus \{o_i\}$

➢ Lose a factor of at most $2k$ at each iteration

➢ Total approximation factor $(2k)^k$

In this Talk

Composable Core-sets for Volume Maximization:

Algorithm:
There exists a polynomial time algorithm for computing an $\boxed{O(k)^{k}}$ -composable core-set of size $\tilde{O}(k)$ for the volume maximization problem.

Lower bound:
Any composable core-set of size $k^{O(1)}$ for the volume maximization problem must have an approximation factor of $\Omega(k)^{\frac{k}{2}(1-o(1))}$.

Optimal core-set

Composable Core-sets for Volume Maximization:

Algorithm:

There exists a polynomial time algorithm for computing an $\boxed{\tilde{O}(k)^{\frac{k}{2}}}$-composable core-set of size $\tilde{O}(k)$ for the volume maximization problem.

Lower bound:

Any composable core-set of size $k^{O(1)}$ for the volume maximization problem must have an approximation factor of $\Omega(k)^{\frac{k}{2}(1-o(1))}$.

Experiments: comparison of core-sets for volume maximization

- Greedy algorithm
  - Widely used in Practice
  - We showed it achieves $O(C^{k^2})$
- Local Search algorithm
  - Performs better than Greedy but runs ~4 times slower.
  - Achieves $O(k)^k$
- The optimal core-set algorithm
  - Achieves $\tilde{O}(k)^{k/2}$
  - Performs worse than Local Search and runs slower.

Summary

- Different notions of diversity
- Notion of composable core-sets
- Algorithms that find composable core-sets for diversity maximization under different notions

Diversity Notion	Coreset Size	Approx.	Reference	Offline
Min Pairwise Distance	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta(1)}$ [Ravi et al 94]
Sum of Pairwise distances	k	$O(1)$	[IMMM'14]	$\boldsymbol{\theta(1)}$ [Hassin et al 97]
…				…
Volume	$\tilde{O}(k)$	$\tilde{O}(k)^{k/2}$	[IMOR'18]	$\boldsymbol{O(c^k), \Omega(c^k)}$ [Nik'15],[CIM'13]

Open Problems

- Characterizing problems that admit composable coresets
- Optimal algorithms for diversity maximization in other massive data models of computation (e.g. Streaming, MPC)
- Composable Core-sets for DPP sampling?

Open Problems

- Characterizing problems that admit composable coresets
- Optimal algorithms for diversity maximization in other massive data models of computation (e.g. Streaming, MPC)
- Composable Core-sets for DPP sampling?

THANK YOU!