Coresets for \( k \)-means clustering

Melanie Schmidt, TU Dortmund

Resource-aware Machine Learning - International Summer School

02.10.2014
Coresets and $k$-means
Coresets and $k$-means clustering
Coresets and $k$-means clustering
Coresets and \( k \)-means clustering
Coresets and \( k \)-means clustering
Introduction

Techniques

BICO

Coresets and $k$-means

Coresets for $k$-means clustering
Coresets and $k$-means clustering
Whether a summary / compressed representation / coreset is good depends on the objective
Whether a summary / compressed representation / coreset is good depends on the objective.

A coreset represents input data:
- with regard to an objective function
- (e.g.) in order to solve an optimization problem
Whether a summary / compressed representation / coreset is good depends on the objective.

A coreset represents input data with regard to an objective function (e.g.) in order to solve an optimization problem.

Notice that:
- there is no common definition
- many approaches can be viewed as a coreset

Coresets for $k$-means clustering
The $k$-means Problem

Given a point set $P \subseteq \mathbb{R}^n$, compute a set $C \subseteq \mathbb{R}^n$ with $|C| = k$ centers which minimizes cost $(P, C) = \sum_{p \in P} \min_{c \in C} ||c - p||^2$, the sum of the squared distances.

$||\cdot||$ is the Euclidean norm.

Coresets for $k$-means clustering
The k-means Problem

- Given a point set $P \subseteq \mathbb{R}^n$,
The \( k \)-means Problem

- Given a point set \( P \subseteq \mathbb{R}^n \),
- compute a set \( C \subseteq \mathbb{R}^n \) with \( |C| = k \) centers
The $k$-means Problem

- Given a point set $P \subseteq \mathbb{R}^n$,
- compute a set $C \subseteq \mathbb{R}^n$ with $|C| = k$ centers
- which minimizes $\text{cost}(P, C)$
  \[ = \sum_{p \in P} \min_{c \in C} ||c - p||^2, \]

the sum of the squared distances.
The $k$-means Problem

- Given a point set $P \subseteq \mathbb{R}^n$,
- compute a set $C \subseteq \mathbb{R}^n$ with $|C| = k$ centers
- which minimizes $\text{cost}(P, C)$
  \[
  = \sum_{p \in P} \min_{c \in C} ||c - p||^2,
  \]
- the sum of the squared distances.

Coresets and $k$-means clustering
Coresets and $k$-means

The $k$-means Problem

- Given a point set $P \subseteq \mathbb{R}^n$,
- compute a set $C \subseteq \mathbb{R}^n$ with $|C| = k$ centers
- which minimizes cost$(P, C)$

$$
\text{cost}(P, C) = \sum_{p \in P} \min_{c \in C} ||c - p||^2,
$$

the sum of the squared distances.

$|| \cdot ||$ is the Euclidean norm
What $k$-means cannot cluster

Coresets for $k$-means clustering
What $k$-means cannot cluster

Coresets for $k$-means clustering
What \( k \)-means cannot cluster

In these cases, other objective functions might be better suited.
Coreset (idea)

- compute a smaller **weighted point set**
- that preserves the *k*-means objective,
- i.e., the **sum of the weighted squared distances is similar**
- for all sets of *k* centers
Coresets and $k$-means

Coreset (idea)
- compute a smaller weighted point set
- that preserves the $k$-means objective,
- i.e., the sum of the weighted squared distances is similar
- for all sets of $k$ centers

Why for all centers?
- coreset and input should look alike for $k$-means
Coreset (idea)
- compute a smaller weighted point set
- that preserves the $k$-means objective,
- i.e., the sum of the weighted squared distances is similar
- for all sets of $k$ centers

Why for all centers?
- coreset and input should look alike for $k$-means
- assume optimizing over the possible centers
- if the cost is underestimated for certain center sets, then they might be mistakenly assumed to be optimal
Small summary of the data that preserves the cost function

**Coresets (Har-Peled, Mazumdar)**

Given a set of points $P \in \mathbb{R}^n$, a weighted set $S$ is a $(k, \epsilon)$-coreset if for all sets $C \subset \mathbb{R}^n$ of $k$ centers it holds that

$$|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)$$

where $\text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) ||p - c||^2$. 
Small summary of the data that preserves the cost function

**Coresets (Har-Peled, Mazumdar)**

Given a set of points $P \in \mathbb{R}^n$, a weighted set $S$ is a $(k, \epsilon)$-coreset if for all sets $C \subset \mathbb{R}^n$ of $k$ centers it holds that

$$|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)$$

where $\text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) ||p - c||^2$. 
Small summary of the data that preserves the cost function

**Coresets (Har-Peled, Mazumdar)**

Given a set of points $P \in \mathbb{R}^n$, a weighted set $S$ is a $(k, \epsilon)$-coreset if for all sets $C \subset \mathbb{R}^n$ of $k$ centers it holds that

$$|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)$$

where $\text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) ||p - c||^2$.}

- Diagram showing points and their weights for $k$-means clustering.
Small summary of the data that preserves the cost function

Coresets (Har-Peled, Mazumdar)

Given a set of points \( P \in \mathbb{R}^n \), a weighted set \( S \) is a \((k, \epsilon)\)-coreset if for all sets \( C \subset \mathbb{R}^n \) of \( k \) centers it holds that

\[
|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)
\]

where \( \text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) |p - c|^2 \).

Coresets for \( k \)-means clustering
Small summary of the data that preserves the cost function

**Coresets (Har-Peled, Mazumdar)**

Given a set of points \( P \in \mathbb{R}^n \), a weighted set \( S \) is a \((k, \epsilon)\)-coreset if for all sets \( C \subset \mathbb{R}^n \) of \( k \) centers it holds that

\[
|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)
\]

where \( \text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) ||p - c||^2 \).
Small summary of the data that preserves the cost function

**Coresets (Har-Peled, Mazumdar)**

Given a set of points $P \in \mathbb{R}^n$, a weighted set $S$ is a $(k, \epsilon)$-coreset if for all sets $C \subset \mathbb{R}^n$ of $k$ centers it holds that

$$|\text{cost}_w(S, C) - \text{cost}(P, C)| \leq \epsilon \text{cost}(P, C)$$

where $\text{cost}_w(S, C) = \sum_{p \in S} \min_{c \in C} w(p) ||p - c||^2$. 

\[4 \quad 2 \quad 5 \]

\[2 \quad 5 \]
### Coreset constructions

- **’01:** Agarwal, Har-Peled and Varadarajan: Coreset concept
- **’02:** Bădoiu, Har-Peled and Indyk: First coreset construction for clustering problems
- **’04:** Har-Peled and Mazumdar, Coreset of size $O(k\epsilon^{-d} \log n)$, maintainable in data streams
- **’05:** Har-Peled and Kushal, Coreset of size $O(k^3\epsilon^{-(d+1)})$
- **’05:** Frahling and Sohler: Coreset of size $O(k\epsilon^{-d} \log n)$, insertion-deletion data streams
- **’06:** Chen: Coresets for metric and Euclidean $k$-median and $k$-means, polynomial in $d$, $\log n$ and $\epsilon^{-1}$
- **’07:** Feldman, Monemizadeh, Sohler: weak coreset poly($k$, $\epsilon^{-1}$)
- **’10:** Langberg, Schulman: $\tilde{O}(d^2k^3/\epsilon^2)$
- **’13:** Feldman, S., Sohler: $(k/\epsilon)^O(1)$
Outline

- Different techniques to construct coresets
- Interlude: Dimensionality reduction
- A practically efficient coreset construction
Technique 0: The magic formula for $k$-means

Zhang, Ramakrishnan, Livny, 1996

For every $P \subset \mathbb{R}^d$ and $z \in \mathbb{R}^d$,

$$\sum_{x \in P} ||x - z||^2 = \sum_{x \in P} ||x - \mu(P)||^2 + |P| \cdot ||\mu(P) - z||^2$$

where $\mu(P) = \sum_{x \in P} x / |P|$ is the centroid of $P$. 

\[\text{Diagram showing the formula applying to a set of points.}\]
Technique 0: The magic formula for $k$-means

Implications

- centroid is always the optimal 1-means solution
- (much nicer situation than for 1-median!)
- centroid (plus constant) is an $(1, \varepsilon)$-coreset with no error
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

- move close points to the same position
- replace coinciding points by a weighted point
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

- move close points to the same position
- replace coinciding points by a weighted point
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

- move close points to the same position
- replace coinciding points by a weighted point

Goal

Overall squared movement small in comparison with cost
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

Let $\text{OPT}$ be the cost of an optimal $k$-means solution.

- move each point $x$ in $P$ to $\pi(x)$, obtain set $Q$
- Ensure that

$$\sum_{x \in P} ||x - \pi(x)||^2 \leq \frac{\varepsilon^2}{16} \cdot \text{OPT}$$

- Then $|\text{cost}(Q) - \text{cost}(P)| \leq \varepsilon \cdot \text{cost}(P)$
- $\Rightarrow \pi(P)$ is a coreset! (but a large one)
Technique 1: Bounded movement of points
Har-Peled, Mazumdar, 2004

Let \( \text{OPT} \) be the cost of an optimal \( k \)-means solution.

- move each point \( x \) in \( P \) to \( \pi(x) \), obtain set \( Q \)
- Ensure that
  \[
  \sum_{x \in P} \| x - \pi(x) \|^2 \leq \frac{\varepsilon^2}{16} \cdot \text{OPT}
  \]

- Then \( |\text{cost}(Q) - \text{cost}(P)| \leq \varepsilon \cdot \text{cost}(P) \)
- \( \Rightarrow \pi(P) \) is a coreset! (but a large one)

- Move points, obtain \( Q \), replace points by weighted points
- Notice: Sum of all movements must be small
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

First idea:
- Place a grid
- Move all points in the same cell to one point
Technique 1: Bounded movement of points

Har-Peled, Mazumdar, 2004

First idea:
- Place a grid
- Move all points in the same cell to one point

Problem:
- Requires a cell width of $\sqrt{\varepsilon^2 \text{OPT}}/(16dn)$
  $\Rightarrow \Omega((nd\varepsilon^{-2})^{d/2})$ cells
- far too large ‘coreset’
Technique 1: Bounded movement of points
Har-Peled, Mazumdar, 2004

Exponential grids:
- Partition $\mathbb{R}^d$ into cells
- Goal: Small cell diameter compared to optimal clustering cost of points in the cell
  $\Rightarrow$ Moving point within a cell is cheap enough
  \[
  \text{distance to center } \geq D + \text{cell diameter } \leq \varepsilon^2 / 16D
  \]
  $\Rightarrow$ movement $\leq \varepsilon^2 / 16$ cost

Closest center in optimal solution
Technique 1: Bounded movement of points
Har-Peled, Mazumdar, 2004

Idea
- Exponentially growing cells
- Diameter grows with distance

Construction
- An exponential grid per center
- $O(\log n)$ rings in each grid
- $O(\varepsilon^{-d})$ cells in each ring
- $O(k \log n \varepsilon^{-d})$ cells

Finally: Bicriteria approximation
Technique 1: Bounded movement of points
Har-Peled, Mazumdar, 2004

Idea
- Exponentially growing cells
- Diameter grows with distance

Construction
- An exponential grid per center
- $O(\log n)$ rings in each grid
- $O(\varepsilon^{-d})$ cells in each ring

Finally: Bicriteria approximation

There exists a $(k, \varepsilon)$-coreset of size $O(k \log^d n/\varepsilon^d)$. 
Frahling, Sohler, 2005

**Idea**
- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
- Thus, smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for $k$-means

**Idea**

- Distribute error more evenly among cells.
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
- $\Rightarrow$ smaller heavy cells contain more points.
- Place a coreset point in every heavy cell that has no heavy child cells.

**Frahling, Sohler, 2005**
Constructing coresets for $k$-means

Frahling, Sohler, 2005

Idea

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
- smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells

There exists a coreset of size $O(k \log n \varepsilon^{-d})$. 
Constructing coresets for $k$-means

Frahling, Sohler, 2005

Idea

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta OPT$
- Smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for $k$-means

**Idea**

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
  - smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells

**Coresets for $k$-means clustering**

**Frahling, Sohler, 2005**
Constructing coresets for $k$-means

Frahling, Sohler, 2005

Idea
- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
  \[ \Rightarrow \] smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for $k$-means

Frahling, Sohler, 2005

Idea
- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
- smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells
Constructing coresets for $k$-means

**Frahling, Sohler, 2005**

**Idea**

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
- Smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for $k$-means

Frahling, Sohler, 2005

Idea
- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
  ⇒ smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for $k$-means

**Frahling, Sohler, 2005**

**Idea**

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$

$\Rightarrow$ smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

There exists a coreset of size $O(k \log n \varepsilon^{-d})$. 
Constructing coresets for $k$-means

**Idea**

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$
  - Smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

**Frahling, Sohler, 2005**

There exists a coreset of size $O(k \log n \epsilon^{-d})$. 
Constructing coresets for $k$-means clustering

Frahling, Sohler, 2005

Idea

- Distribute error more evenly among cells
- A cell is $\delta$-heavy if its diameter times its number of points is $> \delta \text{OPT}$

$\Rightarrow$ smaller heavy cells contain more points
- place a coreset point in every heavy cell that has no heavy child cells

Coresets for $k$-means clustering
Constructing coresets for \(k\)-means

**Frahling, Sohler, 2005**

**Idea**
- Distribute error more evenly among cells
- A cell is \(\delta\)-heavy if its diameter times its number of points is \(> \delta \text{OPT}\)
  \[\Rightarrow\] smaller heavy cells contain more points
- Place a coreset point in every heavy cell that has no heavy child cells

There exists a coreset of size \(O(k \log n^{\varepsilon^{-d}})\).
Har-Peled, Kushal, 2005

Coreset for one-dimensional input

- Subdivide into $O(k^2/\varepsilon^2)$ intervals with $O((\varepsilon/k)^2 \text{OPT})$ cost
- Place two coreset points in each interval with correct mean
- Most of the intervals are clustered with one center
- These induce no error!
- Error for remaining $k - 1$ intervals can be bounded
Har-Peled, Kushal, 2005

Coreset for one-dimensional input

- Subdivide into $\mathcal{O}(k^2/\varepsilon^2)$ intervals with $\mathcal{O}((\varepsilon/k)^2 OPT)$ cost
- Place two coreset points in each interval with correct mean
- Most of the intervals are clustered with one center
- These induce no error!
- Error for remaining $k - 1$ intervals can be bounded

$$
\begin{array}{cccccccc}
2 & 2 & & 1 & & 1 & 1 & & 2 & 2
\end{array}
$$
Har-Peled, Kushal, 2005

Coreset for one-dimensional input

- Subdivide into $O(k^2/\varepsilon^2)$ intervals with $O((\varepsilon/k)^2 \text{OPT})$ cost
- Place two coreset points in each interval with correct mean
- Most of the intervals are clustered with one center
- These induce no error!
- Error for remaining $k - 1$ intervals can be bounded

---

2 2
Har-Peled, Kushal, 2005

**Multidimensional coreset**
- Again, centers of a bicriteria approximation
- Shoot $O(\varepsilon^{-(d-1)})$ rays from each center
- Project points to the rays
- Compute $O(k \cdot \varepsilon^{-(d-1)})$ one-dimensional coresets
Constructing coresets for $k$-means clustering

Har-Peled, Kushal, 2005

Multidimensional coreset

- Again, centers of a bicriteria approximation
- Shoot $\mathcal{O}(\varepsilon^{-(d-1)})$ rays from each center
- Project points to the rays
- Compute $\mathcal{O}(k \cdot \varepsilon^{-(d-1)})$ one-dimensional coresets

There exists a $(k, \varepsilon)$-coreset of size $\mathcal{O}(k^3/\varepsilon^{d} + 1)$.
Har-Peled, Kushal, 2005

**Multidimensional coreset**
- Again, centers of a bicriteria approximation
- Shoot $\mathcal{O}(\varepsilon^{-(d-1)})$ rays from each center
- Project points to the rays
- Compute $\mathcal{O}(k \cdot \varepsilon^{-(d-1)})$ one-dimensional coresets
Constructing coresets for $k$-means

Har-Peled, Kushal, 2005

Multidimensional coreset

- Again, centers of a bicriteria approximation
- Shoot $O(\varepsilon^{-(d-1)})$ rays from each center
- Project points to the rays
- Compute $O(k \cdot \varepsilon^{-(d-1)})$

There exists a $(k, \varepsilon)$-coreset of size $O(k^3 / \varepsilon^{d+1})$. 
Technique 2: Sampling

Sampling Algorithm
- Sample points from \( P \) uniformly at random
- The sampled points form the coreset

Around \( \mathcal{O}(k \cdot \log n \cdot n \cdot \text{diam}(P)/(\varepsilon^2 \cdot \text{OPT})) \) samples needed

Precise statements due to Haussler (1990), can be proven by Hoeffding’s inequality
Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with $diam(P') \approx \frac{\text{cost}(P')}{|P'|}$
- sample representatives from each subset
Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with $\text{diam}(P') \approx \frac{\text{cost}(P')}{|P'|}$
- sample representatives from each subset
 Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with $\text{diam}(P') \approx \text{cost}(P')/|P'|$
- sample representatives from each subset
Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with
  \[ \text{diam}(P') \approx \frac{\text{cost}(P')}{|P'|} \]
- sample representatives from each subset
Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with $\text{diam}(P') \approx \text{cost}(P')/|P'|$
- sample representatives from each subset

Coresets for $k$-means clustering
Technique 2: Sampling

Chen, 2006

- compute bicriteria approximation
- partition input points into subsets with
  \[ \text{diam}(P') \approx \frac{\text{cost}(P')}{|P'|} \]
- sample representatives from each subset
Technique 2: Sampling

Chen, 2006

- Succeeds with constant probability for each center set
- Discretization of the solution space necessary
Technique 2: Sampling

Chen, 2006

- Succeeds with constant probability for each center set
- Discretization of the solution space necessary

There exists a $(k, \varepsilon)$-coreset of size $\tilde{O}(dk^2 \log n/\varepsilon^2)$. 
Technique 2: Refined sampling strategies

Feldman, Monemizadeh, Sohler, 2007

Importance sampling
- Sample points with a probability proportional to their optimum cost
- Weight points accordingly
- For points with low optimum cost, sample uniformly

Improvement due to Langberg, Schulman, 2010.
Technique 2: Refined sampling strategies

Feldman, Monemizadeh, Sohler, 2007

Importance sampling
- Sample points with a probability proportional to their optimum cost
- Weight points accordingly
- For points with low optimum cost, sample uniformly

 Improvement due to Langberg, Schulman, 2010.

There exists a \((k, \varepsilon)\)-coreset of size \(O(d^2 k^3 \varepsilon^{-2})\).
Technique 2: Refined sampling strategies

Feldman, Langberg, 2011

Sensitivity based sampling

- The sensitivity of a point $x \in P$ is

$$\sup_{C \subseteq \mathbb{R}^d, |C| = k} \min_{c \in C} \|x - c\|^2 \sum_{y \in P} \min_{c \in C} \|y - c\|^2$$

- Maximum share of a point in the cost function

$\Rightarrow$ Sampling probabilities proportional to sensitivity
Technique 3: Pseudorandomness

Idea

- If a point set has little structure (it is pseudorandom), clustering it is similar for all centers
  - Clustering it with one center does not induce much error
  - Simulate clustering with one center by using the centroid

Partition the input into pseudorandom subsets
Technique 3: Pseudorandomness
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set
- Continuously subdivide sets until every set $S$ satisfies:
  - Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set
- Continuously subdivide sets until every set $S$ satisfies:
- Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set
- Continuously subdivide sets until every set $S$ satisfies:
- Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set
- Continuously subdivide sets until every set $S$ satisfies:
- Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set.
- Continuously subdivide sets until every set $S$ satisfies:
  - Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center.
Technique 3: Pseudorandomness

- Start with partitioning according to an optimal center set
- Continuously subdivide sets until every set $S$ satisfies:
  - Clustering $S$ with $k$ centers is at most a factor $(1 + \epsilon)$ cheaper than clustering $S$ with one center
  - ... or cost for 1-clustering is negligible $(\epsilon^2 \text{OPT})$
Technique 3: Pseudorandomness

- Sets on level 1 together cost $OPT$
- Sets on level $i$ cost $\frac{OPT}{(1+\epsilon)^i}$
- Sets on level $\log_{1+\epsilon} \epsilon^{-2}$ have negligible cost $(\epsilon^2 OPT)$
- $O \left( k^{\log_{1+\epsilon} \epsilon^{-2}} \right)$ coreset points $\rightarrow$ independent of $n$ and $d$
Technique 4: Dimensionality reduction

Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let \( P \) be a set of \( n \) points in \( \mathbb{R}^n \). Consider the best fit subspace

\[
V_k := \arg \min_{\dim(V) = k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.
\]

Solving the projected instance in \( V_k \) yields a 2-approximation.
Technique 4: Dimensionality reduction

Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\dim(V)=k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality reduction

Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\dim(V) = k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$  

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality reduction

Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\dim(V)=k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality reduction

Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\dim(V) = k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$  

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality reduction
Drineas, Frieze, Kannan, Vempala, Vinay, 1999

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\dim(V)=k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$ 

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality reduction

**Drineas, Frieze, Kannan, Vempala, Vinay, 1999**

Let $P$ be a set of $n$ points in $\mathbb{R}^n$. Consider the best fit subspace

$$V_k := \arg \min_{\text{dim}(V)=k} \sum_{p \in P} d(p, V)^2 \subset \mathbb{R}^n.$$ 

Solving the projected instance in $V_k$ yields a 2-approximation.
Technique 4: Dimensionality Reduction

Drineas et al.

Solving the instance projected to $V_k$ yields a 2-approximation.
Technique 4: Dimensionality Reduction

Drineas et.al.
Solving the instance projected to $V_k$ yields a 2-approximation.

Feldman, S., Sohler, 2013
Projecting to $V_{O(k/\epsilon^2)}$ instead yields a $(1 + \epsilon)$-approximation.

There exists a coreset of size $\tilde{O}(k^4 \epsilon^{-4})$. 
Processing Big Data

- Most coreset constructions need random access
- Undesirable / not possible for Big Data or streaming settings

Coresets for \( k \)-means clustering
Introduction

Techniques

BICO

A practically efficient coreset algorithm

Processing Big Data

- Most coreset constructions need random access
- Undesirable / not possible for Big Data or streaming settings

Conversion to a Streaming Algorithm: Merge & Reduce

- read data in blocks
- compute a coreset for each block $\rightarrow s$
- merge coresets in a tree fashion
- $\sim$ space $s \cdot \log n$

Coreset sizes increase, algorithm has additional overhead

Coresets for $k$-means clustering
A practically efficient coreset algorithm

**Streaming coreset algorithms (no Merge & Reduce)**

- Coreset construction due to Frahling and Sohler
- BICO (Fichtenberger, Gillé, S., Schwiegelshohn, Sohler)
Streaming coreset algorithms (no Merge & Reduce)

- Coreset construction due to Frahling and Sohler
- BICO (Fichtenberger, Gillé, S., Schwiegelshohn, Sohler)

BICO

- based on the datastructure of BIRCH
- works with Technique 1 (bounded movement of points)
- computes a coreset

http://ls2-www.cs.tu-dortmund.de/bico

BIRCH

- Zhang, Ramakrishnan, Livny, 1997
- SIGMOD Test of Time Award 2006
Introduction

Techniques

BICO

A practically efficient coreset algorithm

Coresets for $k$-means clustering
A practically efficient coreset algorithm

1. Find closest reference point
2. If node is not in range
3. Then create a new node
4. Else add to node if possible
5. If not, go one level down,
6. Find closest child, goto 2.
A practically efficient coreset algorithm

Threshold $T$
Radius $R_i$

1. Find closest reference point
2. If node is not in range
3. Then create a new node
4. Else add to node if possible
5. If not, go one level down,
6. Find closest child, goto 2.
A practically efficient coreset algorithm

1. Find closest reference point
2. If node is not in range
3. Then create a new node
4. Else add to node if possible
5. If not, go one level down,
6. Find closest child, goto 2.

Threshold $T$
Radius $R_i$
A practically efficient coreset algorithm

1. Find closest reference point
2. If node is not in range
3. Then create a new node
4. Else add to node if possible
5. If not, go one level down,
6. Find closest child, goto 2.

Threshold $T$
Radius $R_i$
A practically efficient coreset algorithm

1. Find closest reference point
2. If node is not in range
3. Then create a new node
4. Else add to node if possible
5. If not, go one level down,
6. Find closest child, goto 2.

Threshold $T$
Radius $R_i$
Algorithms for comparison

- StreamKM++ and BIRCH (author’s implementations)
- MacQueen’s k–means algorithm (ESMERALDA)

Data sets

<table>
<thead>
<tr>
<th></th>
<th>BigCross</th>
<th>CalTech128</th>
<th>Census</th>
<th>CoverType</th>
<th>Tower</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>$1 \cdot 10^7$</td>
<td>$3 \cdot 10^6$</td>
<td>$2 \cdot 10^6$</td>
<td>$6 \cdot 10^5$</td>
<td>$5 \cdot 10^6$</td>
</tr>
<tr>
<td>$d$</td>
<td>57</td>
<td>128</td>
<td>68</td>
<td>55</td>
<td>3</td>
</tr>
<tr>
<td>$nd$</td>
<td>$7 \cdot 10^8$</td>
<td>$4 \cdot 10^8$</td>
<td>$2 \cdot 10^8$</td>
<td>$3 \cdot 10^7$</td>
<td>$1 \cdot 10^7$</td>
</tr>
</tbody>
</table>

Diagrams

- 100 runs for every test instance
- Values shown in the diagrams are mean values
A practically efficient coreset algorithm

BigCross

Cost

Number of centers = k

Algorithm
- StreamKMPP
- BICO
- MacQueen
- BIRCH

Coresets for $k$-means clustering
A practically efficient coreset algorithm

Coresets for $k$-means clustering

**BigCross**

![Graph showing performance comparison of different algorithms for $k$-means clustering.](image)

- **Algorithm**
  - StreamKMPP
  - BICO
  - MacQueen
  - BIRCH

**Time [seconds]**

- **Number of centers = k**
  - 15
  - 20
  - 25
  - 30
  - 50
  - 100
A practically efficient coreset algorithm

Coresets for \(k\)-means clustering

Thank you for your attention!
A practically efficient coreset algorithm

Thank you for your attention!