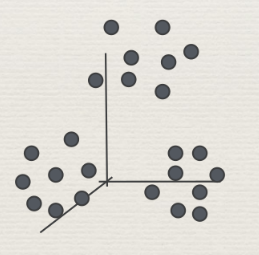
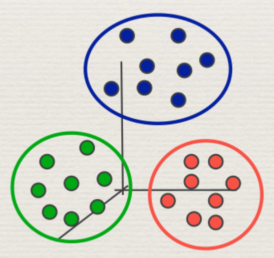
A Faster Sampling Algorithm for Spherical k-means

Rameshwar Pratap, **Anup Deshmukh**, Pratheeksha Nair and Tarun Dutt

What is Clustering

- Partitioning of unlabeled data objects(examples) into disjoint clusters such that
  - data objects within a cluster are very similar
  - data objects in different clusters are very different
- Discover new categories in an unsupervised manner(unlabeled data)





Applications of text clustering

- Pre-processing for fast search
- Postprocessing of search results
- Summarizing news articles along with headlines
- Collaborative filtering algorithms in Recommender Systems

Representation – The Problem of Sparsity

Data mining items	Linear Algebra Items	Neutral Items
Text	Linear	Analysis
Mining	Algebra	Application
Clustering	Matrix	
Classification		

	Research title 1	Research title 2	Research title 3	Research title 4
Term(s) 1	2	0	1	0
Term(s) 2	0	0	0	0
Term(s) 3	0	0	0	0
..	0	0	3	0
..	0	1	0	0
..	1	0	0	0
..	0	0	1	0
..	0	0	0	1
..	0	4	0	0
Term(s) 10	0	0	0	2

How to measure similarity between two documents? – Cosine Similarity

- Cosine similarity is appropriate for determining similarity between documents

$$\cos(D_i, D_j) = \frac{\langle D_i, D_j \rangle}{||D_i||.||D_j||}$$

$$D_1 = (2,3,5), D_2 = (3,7,1), Q = (0,0,2)$$

$$\cos(D_1, Q) = \frac{2*0+3*0+5*2}{\sqrt{4+9+25}\sqrt{0+0+4}} = 0.81$$

$$\cos(D_2, Q) = \frac{3*0+7*0+1*2}{\sqrt{9+49+25}\sqrt{0+0+4}} = 0.13$$

- D_1 is 6 times closer to Q than D_2

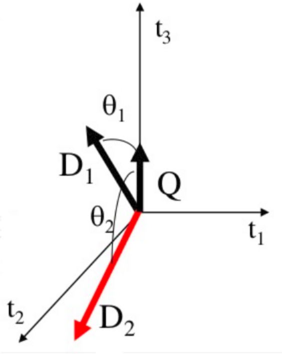


Figure: Cosine Similarity Illustration

Spherical k-means Clustering

Input

- A set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ of n documents (represented as vectors)
- Number of clusters k

- For a set $C = \{c_1, c_2, \ldots, c_k\}$ of cluster centers define:

$$\phi_{\mathcal{X}}(C) = 1.5|\mathcal{X}| - \sum_{x \in \mathcal{X}} \cos(x, C)$$

where $\cos(x, C)$ is cosine similarity between x and its closest center in C

Goal

To find a set C of cluster centers that minimizes the objective function $\phi_{\mathcal{X}}(C)$

Spherical k-means algorithm – SPKM

SPKM Algorithm

- Start with k arbitrary centers $\{c_1, c_2, , c_k\}$ (sampled uniformly at random from data points)
- Performs an EM-type local search till convergence

Spherical k-means algorithm – SPKM

SPKM Algorithm

- Start with k arbitrary centers $\{c_1, c_2, , c_k\}$ (sampled uniformly at random from data points)
- Performs an EM-type local search till convergence

Main advantage: Simplicity

Limitations of SPKM

- May take many iterations to converge
- Very sensitive to initialization
  - Random initialization likely to get two centers in the same cluster, when clusters are highly skewed
  - SPKM gets stuck in a local optimum

SPKM++

Preprocessing

Normalise every data point to unit norm

SPKM++ Algorithm

- **Main idea:** Spreads out the centers
- Choose first center, c_1, uniformly at random from the data set: $C \leftarrow c_1$
- Repeat for $2 \leq i \leq k$:
  - Choose c_i to be equal to a data point x' sampled from the following distribution:

$$\frac{1.5(1 - \cos(x', C))}{\phi_{\mathcal{X}}(C)} \propto (1.5 - \cos(x', C))$$

  - $C \leftarrow C \cup \{c_i\}$

Theorem: $O(\log k)$-approximation to optimum, right after initialization

Experimental comparison between SPKM and SPKM++

Data Set	No. of documents	No. of words in the vocab (dimension)	Max no. of words in a document(sparsity)
NIPS full papers	1500	12419	914
KOS blog entries	3430	6906	457
BBC	9635	2225	128
20News	1700	56916	734

Figure: Dataset description

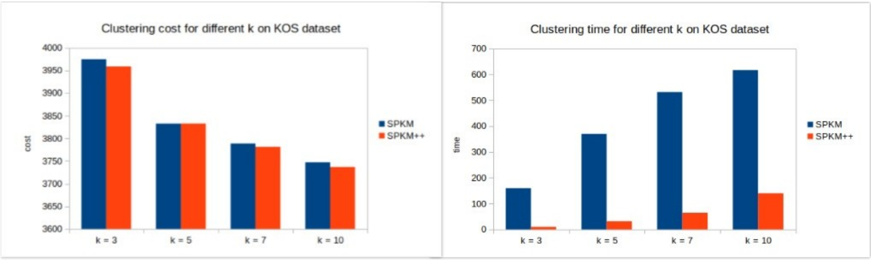


Figure: Comparison in terms of clustering cost and time

Limitations of SPKM++

- Needs k passes over the data for choosing k initial cluster center
- In large data applications, not only the data is massive, but also k is typically large (e.g., easily 1000) and hence is not scalable

SPKM-MCMC – Intuition

Key Idea

Approximate SPKM++ sampling *via* Markov chain sampling

Crux of the Analysis

- We show that for every iterations ($1 \leq i \leq k$) Markov chain distribution is close to the underlying SPKM++ distribution
- As a consequence of above, we show that expected clustering cost is also preserved

SPKM-MCMC – Algorithm and Correctness

Input: Data set $\mathcal{X}$, chain-length m, number of concept vectors k.
Result: A set of initial concept vectors $\mathbf{C} = \{\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}\}$.
  Preprocessing step:
$\mathbf{c_1} \leftarrow$ a vector sampled uniformly at random from $\mathcal{X}$.
for $x \in \mathcal{X}$ **do**
  $q(\mathbf{x}|\mathbf{c_1}) = \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\sum_{\mathbf{x}' \in \mathcal{X}} \mathrm{d}(\mathbf{x}', \mathbf{c_1})} + \frac{1}{2|\mathcal{X}|}$
end
  Main algorithm:
$\mathbf{C} \leftarrow \{\mathbf{c_1}\}$
for $i = 2, 3, \ldots, k$ **do**
  $x \leftarrow$ point sampled from $q(x)$
  $d_x \leftarrow \mathrm{d}(x, \mathbf{C})$
  for $j = 2, 3, \ldots, m$ **do**
    $y \leftarrow$ point sampled from $q(y)$
    $d_y \leftarrow \mathrm{d}(y, \mathbf{C})$
    if $\frac{d_y q(x)}{d_x q(y)} > \mathrm{Unif}(0, 1)$ **then**
      $x \leftarrow y, d_x \leftarrow d_y$
    end
  end
  $\mathbf{C} \leftarrow \mathbf{C} \cup \{x\}$
end

Algorithm 1: Markov chain based faster concept decomposition.

Theorem: $O(\log k) + \epsilon AV(\mathcal{X})$-approximation to optimum.
Where, $d(\mathbf{x}, \mathbf{C}) = \min_{\mathbf{c} \in \mathbf{C}}(1.5 - \langle \mathbf{x}, \mathbf{c} \rangle)$; $m = 1 + O(\frac{1}{\epsilon} \log \frac{k}{\epsilon})$;
$AV(\mathcal{X}) = |\mathcal{X}| - \sum_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{x}, FM(\mathcal{X}) \rangle$; $FM(\mathcal{X}) = \frac{\sum_{\mathbf{x} \in \mathcal{X}} \mathbf{x}}{||\sum_{\mathbf{x} \in \mathcal{X}} \mathbf{x}||}$

Experimental comparison between SPKM++ and SPKM-MCMC

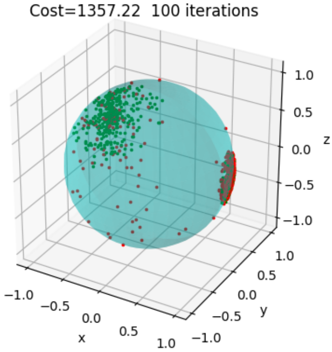
$k = 10$	KOS	BBC	NIPS	20News
SPKM++	0.00%	0.00%	0.00%	0.00%
SPKM-MCMC (m=5)	-0.03%	0.07%	0.08%	0.48%
SPKM-MCMC (m=30)	-0.07%	-0.03%	0.08%	0.03%
SPKM-MCMC (m=100)	-0.06%	-0.03%	0.09%	-0.14%
SPKM-MCMC (m=500)	-0.43%	0.06%	-0.13%	-0.08%

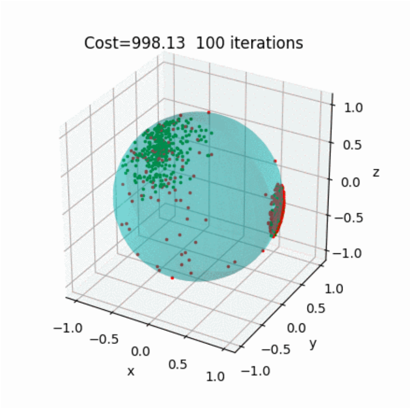
Figure: Comparison of clustering cost between two algorithms for different values of m

$k = 10$	KOS	BBC	NIPS	20News
SPKM++	1	1	1	1
SPKM-MCMC (m=5)	$\times 8.0$	$\times 7.5$	$\times 5.4$	$\times 4.8$
SPKM-MCMC (m=30)	$\times 7.6$	$\times 7.0$	$\times 5.0$	$\times 3.3$
SPKM-MCMC (m=100)	$\times 6.6$	$\times 5.7$	$\times 4.2$	$\times 1.8$
SPKM-MCMC (m=500)	$\times 4.0$	$\times 2.7$	$\times 2.2$	$\times 0.5$

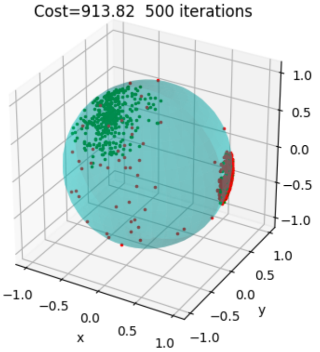
Figure: Comparison of seeding time between two algorithms for different values of m

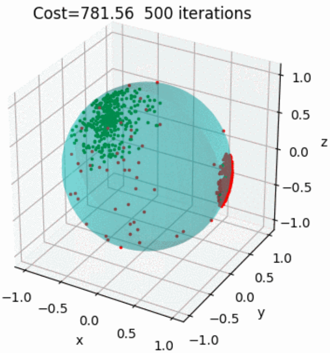
Comparing SPKM++ and SPKM-MCMC



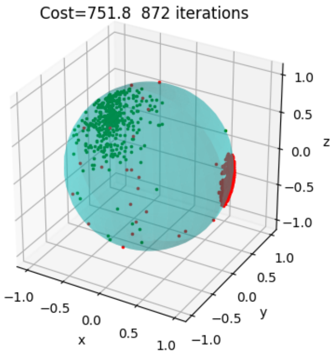


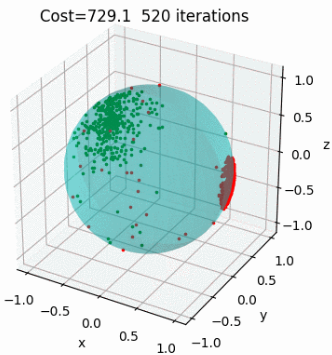
Comparing SPKM++ and SPKM-MCMC





Comparing SPKM++ and SPKM-MCMC





Conclusion

Practicality of SPKM++ algorithm

We conducted thorough experimental evaluations of SPKM++ (Endo and Miyamoto, 2015) on publicly available datasets to demonstrate its applicability, which was not addressed in their paper.We obtain improved clustering quality in addition to better running time with respect to vanilla SPKM (Dhillon and Modha, 2001)

SPKM-MCMC

We propose a Markov chain based algorithm(SPKM-MCMC) for initial seeding of k points. The theoretical guarantee on the clustering cost of our algorithm is close to SPKM++ while achieving a significant speed-up in the seeding time

Implementation

Link to our Github code: *https://github.com/nair-p/SPKM*

Questions/Comments?

Thank You!