

K -medoids for K -means seeding

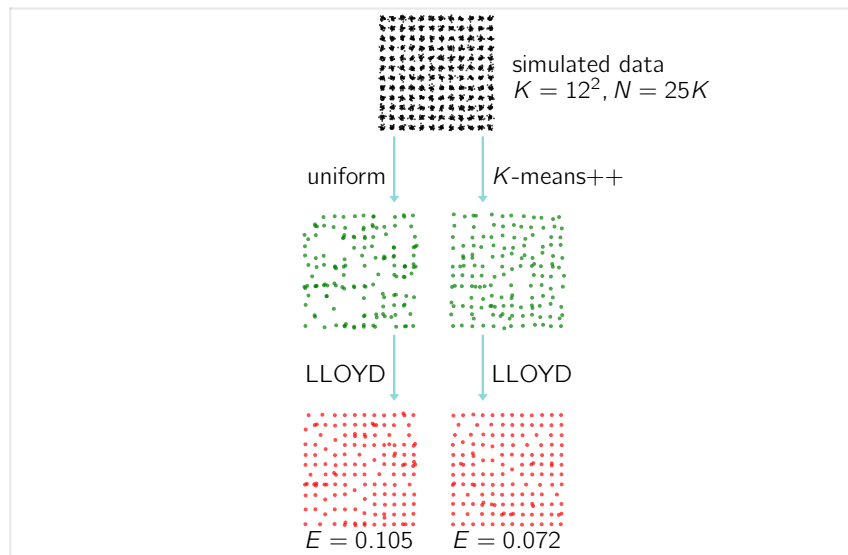
James Newling & François Fleuret

Machine Learning Group,
Idiap Research Institute

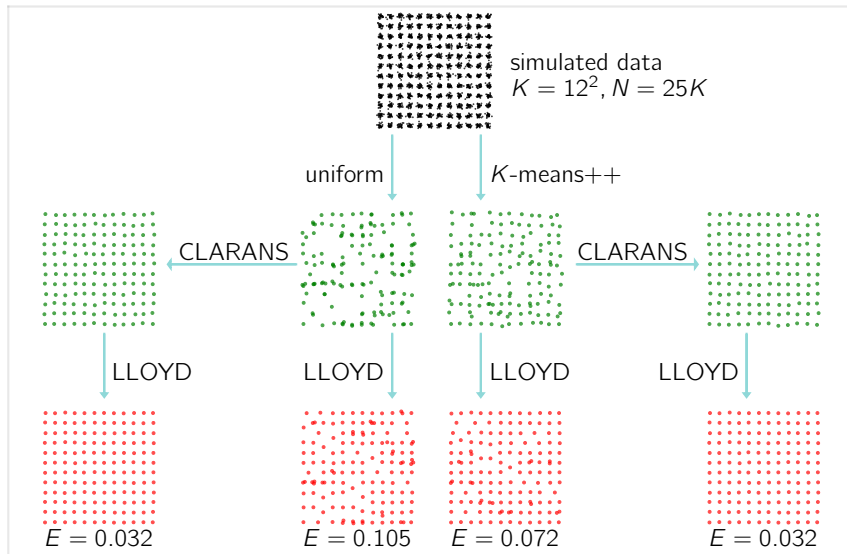
November 28th, 2017

The standard K -means pipeline

First step: Seeding. Second step: Lloyd's algorithm



The standard K -means pipeline (+CLARANS)



Talk Outline

- 1) K -medoids, K -means++ and LLOYD
- 2) The CLARANS algorithm of Ng and Han (1994), algorithmic complexity and improvements
- 3) Results

K -medoids problem

Input :

- N samples $\mathcal{X} = \{x(1), \dots, x(N)\}$
- Dissimilarity function $diss : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$

Task :

- Find K indices $c(1), \dots, c(K) \in \{1, \dots, N\}$ to minimize

$$E = \sum_{i=1}^N \min_{k=1:K} diss(x(i), x(c(k))).$$

- NP-hard.

K -means++ seeding

- Arthur and Vassilvitskii (2007)
- A K -medoids algorithm for

$$\text{diss}(x(i), x(i')) = \|x(i) - x(i')\|^2.$$

- 1: select $c(1)$ uniformly from $\{1, \dots, N\}$
- 2: **for** $k = 2 : K$ **do**
- 3: select $c(k) = i$ with prob $\sim \min_{k' < k} \text{diss}(x(i), x(c(k')))$
- 4: **end for**

- Provides $8 \ln K + 2$ approximation bound to optimal K -means solution in expectation

LLOYD for K -means:

- 1: $C(k) \leftarrow x(c(k))$ for $k \in \{1, \dots, K\}$
- 2: **while** not converged **do**
- 3: for $i = 1 \rightarrow N$ set $a(i) = \operatorname{argmin}_{k=1:K} \|x(i) - C(k)\|$
- 4: for $k = 1 \rightarrow K$ set $C(k) \leftarrow \frac{\sum_{i:a(i)=k} x(i)}{\|i : a(i) = k\|}$
- 5: **end while**

LLOYD for K -means:

- 1: $C(k) \leftarrow x(c(k))$ for $k \in \{1, \dots, K\}$
- 2: **while** not converged **do**
- 3: for $i = 1 \rightarrow N$ set $a(i) = \operatorname{argmin}_{k=1:K} \|x(i) - C(k)\|$
- 4: for $k = 1 \rightarrow K$ set $C(k) \leftarrow \frac{\sum_{i:a(i)=k} x(i)}{\|i : a(i) = k\|}$
- 5: **end while**

MEDLLOYD for K -medoids similar, with the constraint that centers are always samples,

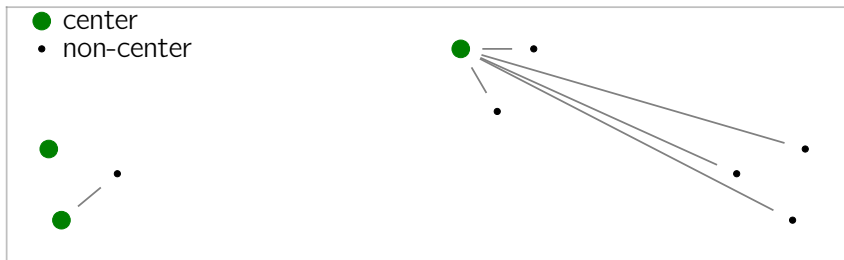
$$C(k) \leftarrow x \left(\operatorname{argmin}_{i:a(i)=k} \sum_{i':a(i')=k} \|x(i) - x(i')\|^2 \right)$$

A very simple K -medoids algorithm:

Randomly propose swaps between 1 center and 1 non-center, accept if E decreases.

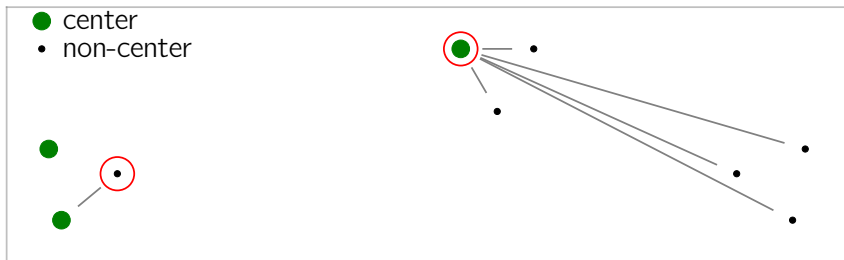
CLARANS

Demonstration



CLARANS

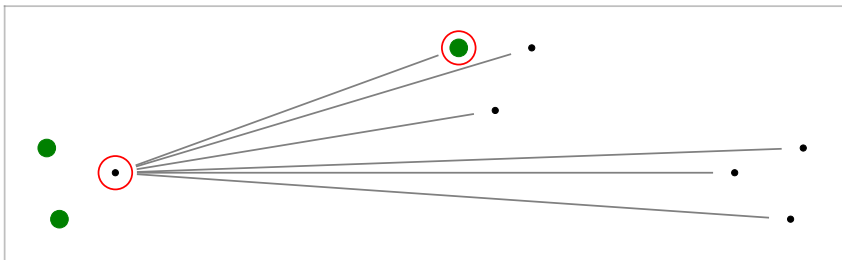
Demonstration



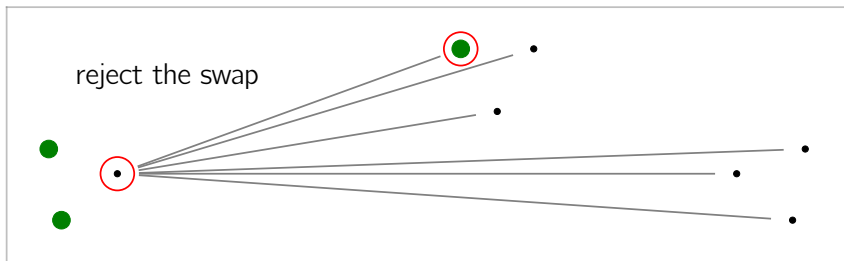
1) propose a swap

CLARANS

Demonstration



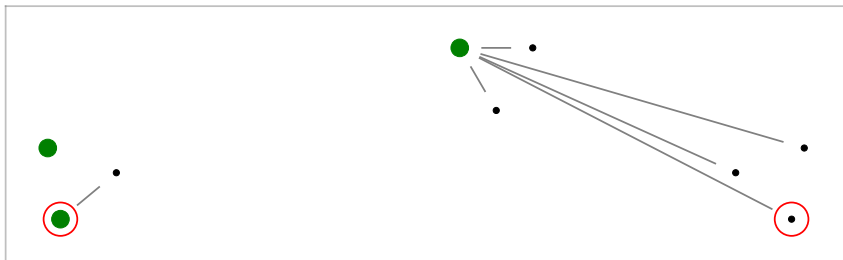
- 1) propose a swap
- 2) evaluate energy



- 1) propose a swap
- 2) evaluate energy
- 3) implement or reject

CLARANS

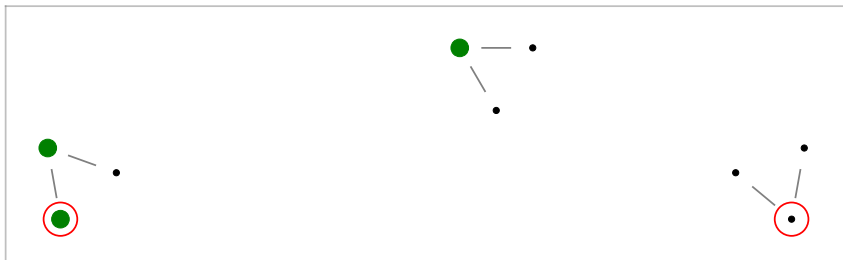
Demonstration



1) propose a swap

CLARANS

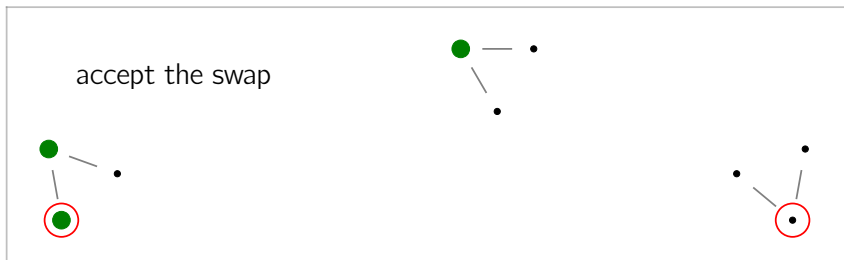
Demonstration



- 1) propose a swap
- 2) evaluate energy

CLARANS

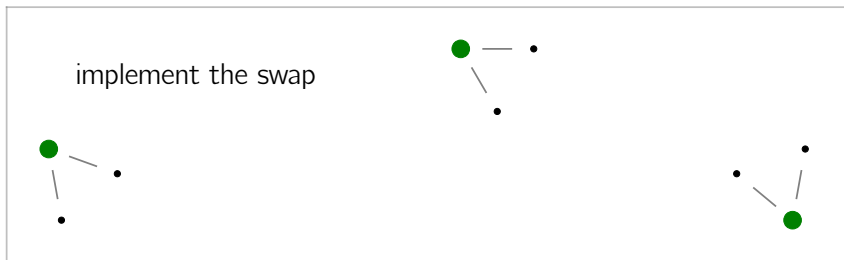
Demonstration



- 1) propose a swap
- 2) evaluate energy
- 3) implement or reject

CLARANS

Demonstration



- 1) propose a swap
- 2) evaluate energy
- 3) implement or reject

CLARANS

Advantages

Advantages of CLARANS over MEDLLOYD (and LLOYD) are,

- updates are *not local*
- assignments and centers change simultaneously

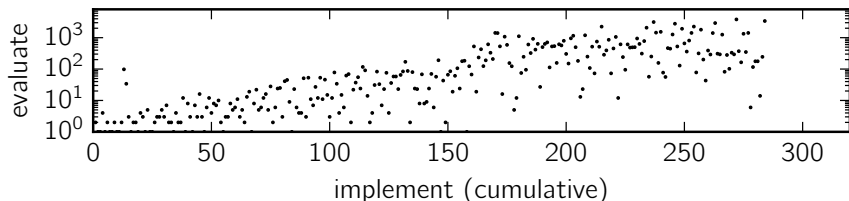
Easy to show that,

- $\{\text{local minima of CLARANS}\} \subseteq \{\text{local minima of MEDLLOYD}\}.$

Ng and Han use N^2 dissimilarity matrix, infeasible now

Useful to distinguish between **evaluate** and **implement** steps:

- 1: **while** stopping condition is false **do**
- 2: randomly select center and non-center
- 3: **evaluate** proposal energy
- 4: **if** proposal energy lower **then**
- 5: **implement** proposed swap
- 6: **end if**
- 7: **end while**



We present different levels of optimization:

1. For all samples, record distances to nearest and second nearest centers (d_1 and d_2 respectively).
2. Also record for all clusters maximum d_1 and d_2 , and inter-center distances. (Δ)

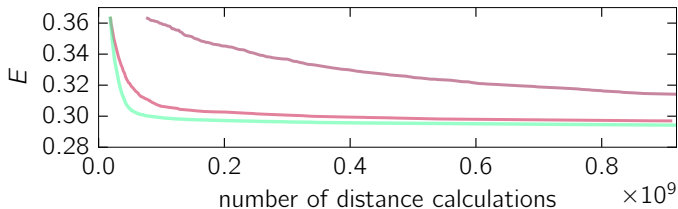
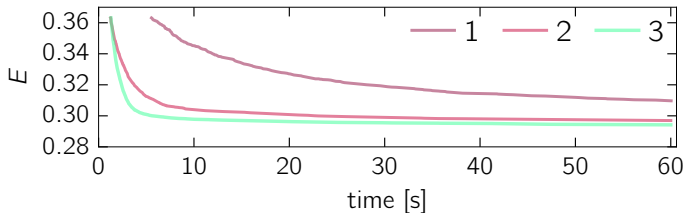
Assuming largest cluster is $O(N/K)$,

	1.	2.
evaluate	$O(N)$	$O(\frac{N}{K})$
implement	$O(N)$	$O(N)$

3. Terminate evaluation early if swap unpromising.

Empirical speed-up

$N = 10^6/2$, $K = 10^3/2$, data from 4-d Gaussian with I covariance .

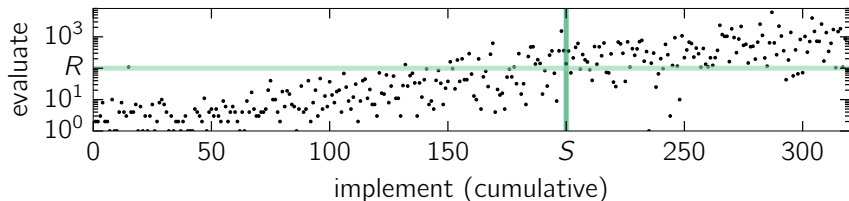


CLARANS for K -means seeding

Stopping criterion

Two possible stopping criteria

- Ng and Han stop after R consecutive swap rejections
- Can stop after S implementations (swap accepts)

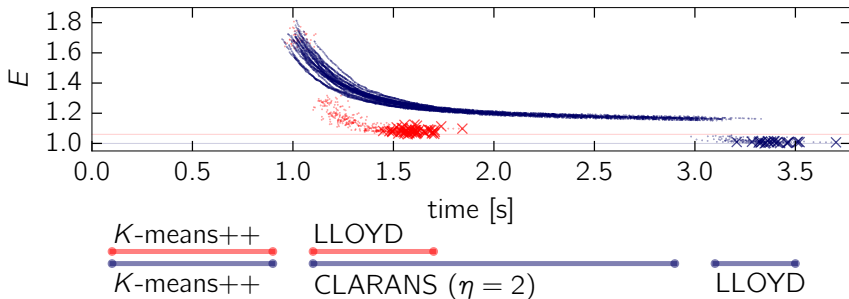


We prefer a time based criterion

- If first seeding (with K -means++) takes T_0 , stop after ηT_0 .

Experiment 1

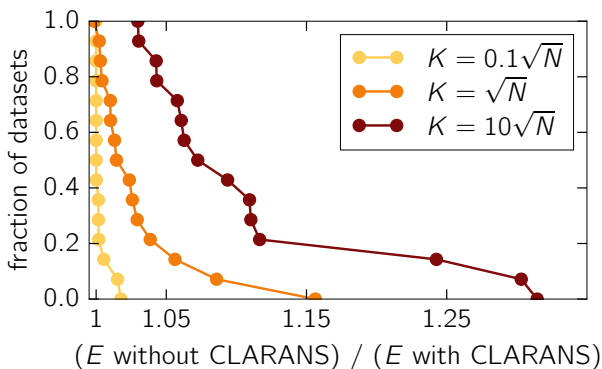
Subset of RNA dataset, $d = 8$, $N = 16 \times 10^4$, $K = 400$.



50 runs of K -means++ \rightarrow LLOYD, and several with CLARANS.
Number with CLARANS chosen so total times equal. Comparing best solutions, using CLARANS results in 6% lower E .

Results Summary

Summary of experiments. Each point is an experiment with same setup as previous slide, horizontal position is reduction in E obtained using CLARANS.

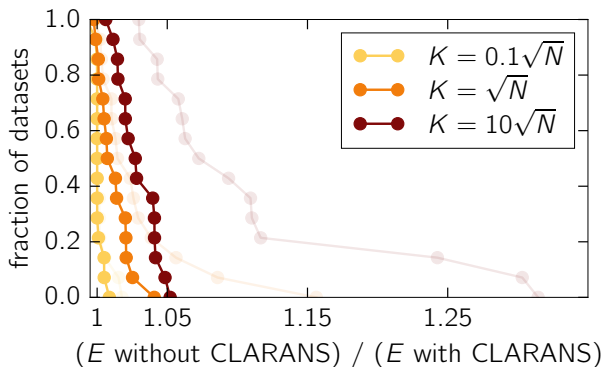


Dataset dimensions range in $d : 2 \rightarrow 90$, $N : 1484 \rightarrow 488565$.

From conclusion of *K*-means++ paper of Arthur and Vassilvitskii (2007):

"Also, experiments showed that k-means++ generally performed better if it selected several new centers during each iteration, and then greedily chose the one that decreased E as much as possible. Unfortunately, our proofs do not carry over to this scenario."

K -means++ revisited



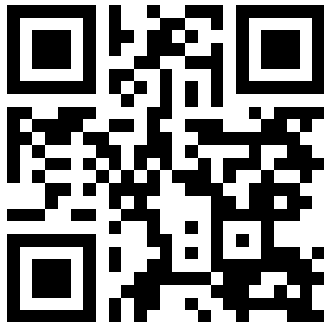
Comparison to version of K -means++ referred to in Conclusion of Arthur and Vassilvitskii (2007) (selecting from best of 5 new centers). Using CLARANS still improves results, but by less.

Clustering library: zentas

$(\triangle) : diss(x(i), x(i')) = \psi(dist(x(i), x(i')))$, where

- $\psi : \mathbb{R} \rightarrow \mathbb{R}$ is non-decreasing
- $dist : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ satisfies the \triangle -inequality

Our software **zentas** implements accelerated CLARANS for different metrics. Levenshtein for sequence data, $l_0, l_1, \dots, l_\infty$ for sparse/dense vectors. Also fast K -means++, LLOYD, many others.



Conclusion

We discussed CLARANS, and how

- to accelerate it
- it improves seeding for K -means
- it is a versatile clustering algorithm in its own right.

