Fast and Provably Good Seedings for k-Means

Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause
Fast and Provably Good Seedings for k-Means

Teaser
Fast and Provably Good Seedings for k-Means

Teaser

Up to 1'064x @ 1.32% speedup relative error compared to k-means++
Fast and Provably Good Seedings for k-Means

Teaser

UP TO 1'064x @ 1.32% SPEEDUP  RELATIVE ERROR
COMPARED TO K-MEANS++

+ THEORETICAL GUARANTEES
k-Means clustering
k-Means clustering

Most popular clustering approach (nonconvex)
k-Means clustering

Most popular clustering approach (nonconvex)

SEEDING
Find initial cluster centers
Fast and Provably Good Seedings for k-Means

**k-Means clustering**

Most popular clustering approach (*nonconvex*)

- **SEEDING**
  - Find initial cluster centers

- **FINE-TUNING**
  - Iteratively improve solution
k-Means clustering

Most popular clustering approach (nonconvex)

MANY LOCAL MINIMA MAY EXIST

SEEDING
Find initial cluster centers

FINE-TUNING
Iteratively improve solution
Fast and Provably Good Seedings for k-Means

**k-Means clustering**

Most popular clustering approach (*nonconvex*)

Many local minima may exist

---

**SEEDING**
Find initial cluster centers

**FINE-TUNING**
Iteratively improve solution

Ensures that local minimum is reached
k-Means clustering

Most popular clustering approach (nonconvex)

MANY LOCAL MINIMA MAY EXIST

SEEDING
Find initial cluster centers

FINE-TUNING
Iteratively improve solution

DETERMINES WHICH
LOCAL MINIMUM
IS REACHED

ENSURES THAT
LOCAL MINIMUM
IS REACHED
k-Means clustering

Most popular clustering approach (nonconvex)

MANY LOCAL MINIMA MAY EXIST

SEEDING
Find initial cluster centers

DETERTMINES WHICH
LOCAL MINIMUM
IS REACHED

FINE-TUNING
Iteratively improve solution

ENSURES THAT
LOCAL MINIMUM
IS REACHED

SEEDING IS CRITICAL!
k-Means algorithms

**SEEDING**
Find initial cluster centers

**FINE-TUNING**
Iteratively improve solution
Fast and Provably Good Seedings for k-Means

k-Means algorithms

SEEDING
Find initial cluster centers

k-Means++ seeding

FINE-TUNING
Iteratively improve solution
Fast and Provably Good Seedings for k-Means

**k-Means algorithms**

**SEEDING**
Find initial cluster centers

k-Means++ seeding

**FINE-TUNING**
Iteratively improve solution

Lloyd’s algorithm
Fast and Provably Good Seedings for k-Means

k-Means algorithms

SEEDING
Find initial cluster centers

FINE-TUNING
Iteratively improve solution

k-Means++ seeding SLOW
Lloyd’s algorithm SLOW
k-Means algorithms

SEEDING
Find initial cluster centers

k-Means++ seeding   SLOW

FINE-TUNING
Iteratively improve solution

Lloyd’s algorithm      SLOW
Mini-batch k-Means    FAST
**k-Means algorithms**

**SEEDING**
Find initial cluster centers

- k-Means++ seeding (SLOW)
- ??

**FINE-TUNING**
Iteratively improve solution

- Lloyd’s algorithm (SLOW)
- Mini-batch k-Means (FAST)

**NEED FOR FAST AND GOOD SEEDINGS**
Random seeding
Random seeding

Sample data points uniformly at random as cluster centers
Random seeding

Sample data points $\bullet$ uniformly at random as cluster centers $\star$
Fast and Provably Good Seedings for k-Means

k-Means++ seeding  [Arthur et al., 2007]
k-Means++ seeding [Arthur et al., 2007]

- Sample first center uniformly at random
**k-Means++ seeding** [Arthur et al., 2007]

- Sample first center uniformly at random
- for i=2, 3, ..., k:
  - sample point $x$ with
    - $p(x) \propto d(x, C)^2$

**D^2-SAMPLING**
**k-Means++ seeding**  [Arthur et al., 2007]

- Sample first center uniformly at random.
- For i = 2, 3, ..., k:
  - Sample point x with
    \[ p(x) \propto \frac{d(x, C)^2}{D^2} \]

where \( D^2 \)-SAMPLING.
**k-Means++ seeding** [Arthur et al., 2007]

- Sample first center uniformly at random
- for $i = 2, 3, \ldots, k$:
  - sample point $x$ with probability
    $$p(x) \propto d(x, C)^2$$

\[
D^2\text{-SAMPLING}
\]
k-Means++ seeding [Arthur et al., 2007]

- Sample first center uniformly at random

- for i=2, 3, ..., k:
  - sample point x with
    \[ p(x) \propto d(x, C)^2 \]

D^2-SAMPLING
Fast and Provably Good Seedings for k-Means

**k-Means++ seeding** [Arthur et al., 2007]

- Sample first center uniformly at random
- For $i = 2, 3, \ldots, k$:
  - Sample point $x$ with
    \[ p(x) \propto d(x, C)^2 \]

**D²-SAMPLING**
**k-Means++ seeding** [Arthur et al., 2007]

- Sample first center uniformly at random
- for $i = 2, 3, \ldots, k$:
  - sample point $x$ with $p(x) \propto d(x, C)^2$

*D²-Sampling*
Fast and Provably Good Seedings for k-Means

k-Means++ seeding [Arthur et al., 2007]
Fast and Provably Good Seedings for k-Means

**k-Means++ seeding** [Arthur et al., 2007]

PROVABLY GOOD

$$\mathbb{E}[\phi_{km++}] \leq 8(\log_2 k + 2)\phi_{OPT}$$

ALREADY AFTER SEEDING

[Arthur et al., 2007]
**k-Means++ seeding**  [Arthur et al., 2007]

**PROVABLY GOOD**

\[ \mathbb{E} \left[ \phi_{km++} \right] \leq 8 \log_2 k + 2 \phi_{OPT} \]

ALREADY AFTER SEEDING

[Arthur et al., 2007]

**BUT SLOW**

\[ \mathcal{O}(nkd) \]

HARD TO PARALLELISE

k sequential passes!
Fast and Provably Good Seedings for k-Means

Single round of $D^2$-sampling
Single round of $D^2$-sampling

- Sample each point with probability

$$p(x) = \frac{d(x, C)^2}{\sum_{x' \in X} d(x', C)^2}$$
Sample each point with probability

\[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in X} d(x', C)^2} \]

REQUIRES LINEAR PASS
Single round of $D^2$-sampling

ɐ Sample each point with probability

\[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in X} d(x', C)^2} \]

REQUIRES LINEAR PASS

❓ How can we efficiently approximate this step?
Markov chain Monte Carlo approach
Markov chain Monte Carlo approach

**Goal:** construct a Markov chain
Markov chain Monte Carlo approach

**Goal:** construct a Markov chain

- where data points are states
Markov chain Monte Carlo approach

**Goal:** construct a Markov chain

☑️ where data points are states
Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states
- whose stationary distribution is

\[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in \mathcal{X}} d(x', C)^2} \]
Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states
- whose stationary distribution is

\[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in \mathcal{X}} d(x', C)^2} \]
Markov chain Monte Carlo approach

**Goal:** construct a Markov chain

- where data points are states
- whose stationary distribution is
  \[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in X} d(x', C)^2} \]
- with a fast mixing time
Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states
- whose stationary distribution is
  \[ p(x) = \frac{d(x, C)^2}{\sum_{x' \in \mathcal{X}} d(x', C)^2} \]
- with a fast mixing time
Markov chain construction
Markov chain construction

Start with an arbitrary initial state $x_0$
Markov chain construction
Markov chain construction

Propose new candidate \( y_i \) according to "some" \( q(x) \)
Propose new candidate $y_i$ according to "some" $q(x)$
Markov chain construction

Propose new candidate $y_i$ according to "some" $q(x)$

Set $x_i = y_i$ with probability

$$\min \left( 1, \frac{d(y_j, C)^2}{d(x_{j-1}, C)^2} \frac{q(x_{j-1})}{q(y_j)} \right),$$

otherwise keep $x_i = x_{i-1}$.
Markov chain construction

- Propose new candidate $y_i$ according to "some" $q(x)$

- Set $x_i = y_i$ with probability

\[
\min \left( 1, \frac{d(y_j, C)^2}{d(x_{j-1}, C)^2} \frac{q(x_{j-1})}{q(y_j)} \right),
\]

otherwise keep $x_i = x_{i-1}$
Markov chain construction
Markov chain construction

Repeat $m$ times to create Markov chain of length $m$
Repeat $m$ times to create Markov chain of length $m$
Markov chain construction

\( \text{Repeat } m \text{ times to create Markov chain of length } m \)
Markov chain construction

Repeat m times to create Markov chain of length m
Markov chain construction

Repeat \( m \) times to create Markov chain of length \( m \)
Markov chain construction

Repeat $m$ times to create Markov chain of length $m$
Markov chain construction

Repeat \( m \) times to create Markov chain of length \( m \)
Markov chain construction

- Repeat $m$ times to create Markov chain of length $m$
- Return $x_m$ as cluster center
Markov chain construction

Repeat \( m \) times to create Markov chain of length \( m \)

Return \( x_m \) as cluster center

APPROXIMATE SINGLE STEP OF \( D^2 \) SAMPLING
Fast and Provably Good Seedings for k-Means

Algorithm
Algorithm

- Sample first center uniformly
Fast and Provably Good Seedings for k-Means

Algorithm

1. Sample first center uniformly
2. Compute proposal distribution $q(x)$
Sequentially construct \( k-1 \) independent Markov chains to obtain \( k-1 \) cluster centers.
Fast and Provably Good Seedings for k-Means

Algorithm

- Sample first center uniformly
- Compute proposal distribution \( q(x) \)
- Sequentially construct \( k-1 \) independent Markov chains to obtain \( k-1 \) cluster centers

\( \checkmark \) Approximation of k-Means++ seeding

EFFICIENT IF \( M \) IS SMALL ENOUGH
Fast and Provably Good Seedings for k-Means

K-MC² [Bachem et al., 2016]
Uniform proposal:

\[ q(x) = \frac{1}{n} \]
Fast and Provably Good Seedings for $k$-Means

$K\text{-MC}^2$ [Bachem et al., 2016]

Uniform proposal:

$$q(x) = \frac{1}{n}$$

⚠️ Misses small, far away clusters

NEVER PROPOSED
Uniform proposal:

\[ q(x) = \frac{1}{n} \]

- Misses small, far away clusters
- Requires assumptions on data or approach fails

\[ \text{NEVER PROPOSED} \]
Assumption Free K-MC² [This paper]
Nonuniform proposal:

\[ q(x) = \frac{1}{2n} + \frac{1}{2} \sum_{x' \in X} \frac{d(x, c_1)^2}{d(x', c_1)^2} \]
Nonuniform proposal:

\[ q(x) = \frac{1}{2n} + \frac{1}{2} \sum_{x' \in \mathcal{X}} \frac{d(x, c_1)^2}{d(x', c_1)^2} \]
Assumption Free K–MC$^2$ [This paper]

Nonuniform proposal:

$$q(x) = \frac{1}{2n} + \frac{1}{2} \sum_{x' \in X} d(x', c_1)^2$$

Provably good w/o assumptions

BIASED TOWARDS FAR AWAY POINTS
Assumption Free K-MC\(^2\)  

- Nonuniform proposal:
  \[ q(x) = \frac{1}{2n} + \frac{1}{2} \sum_{x' \in X} d(x', c_1)^2 \]

- Provably good w/o assumptions
- Works really well empirically
Nonuniform proposal:

\[ q(x) = \frac{1}{2n} + \frac{1}{2} \sum_{x' \in X} d(x', c_1)^2 \]

COMPUTED ONCE IN SINGLE LINEAR PASS

- Provably good w/o assumptions
- Works really well empirically

BIASED TOWARDS FAR AWAY POINTS
Main theoretical result
Main theoretical result

Choose an error tolerance $\epsilon > 0$
Main theoretical result

Choose an error tolerance $\epsilon > 0$

Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$
Main theoretical result

- Choose an error tolerance $\epsilon > 0$

  \[ m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon} \]  
  \text{INDEPENDENT OF DATA SET SIZE}
Main theoretical result

① Choose an error tolerance $\epsilon > 0$

② Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$

③ Expected solution quality:

$$E[\phi_{\text{AFK-MC}^2}] \leq 8(\log_2 k + 2) \phi_{\text{OPT}} + \epsilon \text{Var}(X')$$
Main theoretical result

Choose an error tolerance $\epsilon > 0$

Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$ Independent of data set size.

Expected solution quality:

$$\mathbb{E}[\phi_{AFK-MC}^2] \leq 8(\log_2 k + 2) \phi_{OPT} + \epsilon \text{Var}(\mathcal{X})$$

Same as k-means++
Main theoretical result

Choose an error tolerance $\epsilon > 0$

Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$ INDEPENDENT OF DATA SET SIZE

Expected solution quality:

$$\mathbb{E}[\phi_{\text{AFK-MC}}^2] \leq 8(\log_2 k + 2)\phi_{\text{OPT}} + \epsilon \text{Var}(X)$$

SAME AS K-MEANS++
Main theoretical result

Choose an error tolerance \( \epsilon > 0 \)

Run algorithm with

\[
m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}
\]

INDEPENDENT OF DATA SET SIZE

Expected solution quality:

\[
\mathbb{E}[\phi_{\text{AFK-MC}}^2] \leq 8(\log_2 k + 2)\phi_{\text{OPT}} + \epsilon \text{Var}(\mathcal{X})
\]

SAME AS K-MEANS++

Total runtime:

\[
\mathcal{O}\left(nd + \frac{1}{\epsilon}k^2d \log \frac{k}{\epsilon}\right)
\]
Main theoretical result

Choose an error tolerance $\epsilon > 0$

Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$

Expected solution quality:

$$\mathbb{E} [\phi_{\text{AFK-MC}^2}] \leq 8(\log_2 k + 2) \phi_{\text{OPT}} + \epsilon \text{Var}(\mathcal{X})$$

Total runtime: $O \left( nd + \frac{1}{\epsilon} k^2 d \log \frac{k}{\epsilon} \right)$
Experimental results
Experimental results

Quantization error

Markov chain length

CSN
Experimental results

Quantization error

- Markov chain length
- k-Means++

Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause

CSN
Experimental results

Quantization error vs Markov chain length

- Random
- K-MC^2
- k-Means++

CSN
Experimental results

Quantization error vs. Markov chain length

- Random
- K-MC
- AFK-MC
- CSN

Error decreases as the Markov chain length increases.
Experimental results

![Graph showing the relationship between quantization error and Markov chain length for CSN.]
Experimental results

- CSN
- KDD
- CODRNA
- MSYP
- SUSY
- WEB

Quantization error vs. Markov chain length for different datasets.
Experimental results

Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause

Quantization error

Markov chain length

CSN

KDD

CODRNA

MSYP

SUSY

WEB
Experimental results

M=100 IS SUFFICIENT IN PRACTICE
Experimental results

M=100 IS SUFFICIENT IN PRACTICE
Error vs time tradeoff
Error vs time tradeoff

Quantization error vs # distance evaluations

CSN

- k-Means++
Error vs time tradeoff

Quantization error

# distance evaluations

K-MC^2

k-Means++

CSN
Error vs time tradeoff

Quantization error vs 

# distance evaluations

K-MC²

AFK-MC²

k-Means++
Error vs time tradeoff

Quantization error vs # distance evaluations

- K-MC^2
- AFK-MC^2
- k-Means++

SUBSTANTIAL SPEEDUP
Code
Code

PYTHON IMPLEMENTATION

Available at olivierbachem.ch or with

pip install kmc2

FEATURES

☑️ drop-in replacement for k-means++
☑️ easy to use (2 lines)
☑️ compatible with scikit-learn
Fast and Provably Good Seedings for k-Means

Poster

# 175

TODAY
6 TO 9.30 PM
Appendix
Comparison to k-Means|| [Bachem et al., 2016]