Fast and Provably Good Seedings for k-Means

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

Teaser

Teaser

UP TO 1'064x @ 1.32%

SPEEDUP RELATIVE ERROR

COMPARED TO 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

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

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		FINE-TUNING
Find initial cluster centers	→	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		FINE-TUNING
Find initial cluster centers	→	Iteratively improve solution

DETERMINES **WHICH** LOCAL MINIMUM IS REACHED

ENSURES **THAT** LOCAL MINIMUM IS REACHED

SEEDING IS CRITICAL!

k-Means algorithms

SEEDING

Find initial cluster centers

→

FINE-TUNING

Iteratively improve solution

k-Means algorithms

SEEDING
Find initial cluster centers

→

FINE-TUNING
Iteratively improve solution

k-Means++ seeding

k-Means algorithms

SEEDING		FINE-TUNING
Find initial cluster centers	⇨	Iteratively improve solution
k-Means++ seeding		Lloyd's algorithm

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

FINE-TUNING

Iteratively improve solution

k-Means++ seeding SLOW

Lloyd's algorithm SLOW

Mini-batch k-Means FAST

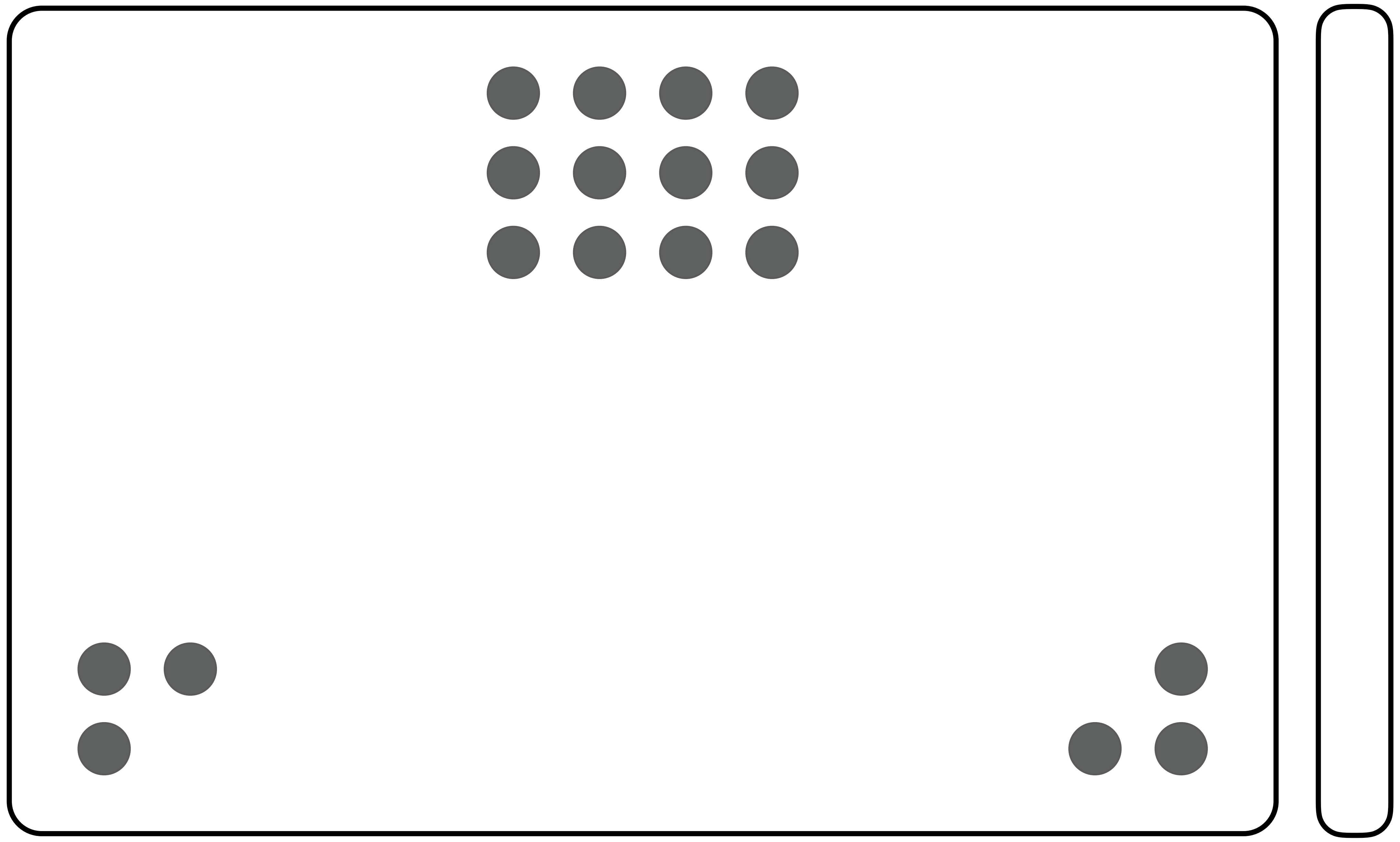
k-Means algorithms

SEEDING		FINE-TUNING
Find initial cluster centers	→	Iteratively improve solution

k-Means++ seeding	SLOW	Lloyd's algorithm	SLOW
??		Mini-batch k-Means	FAST

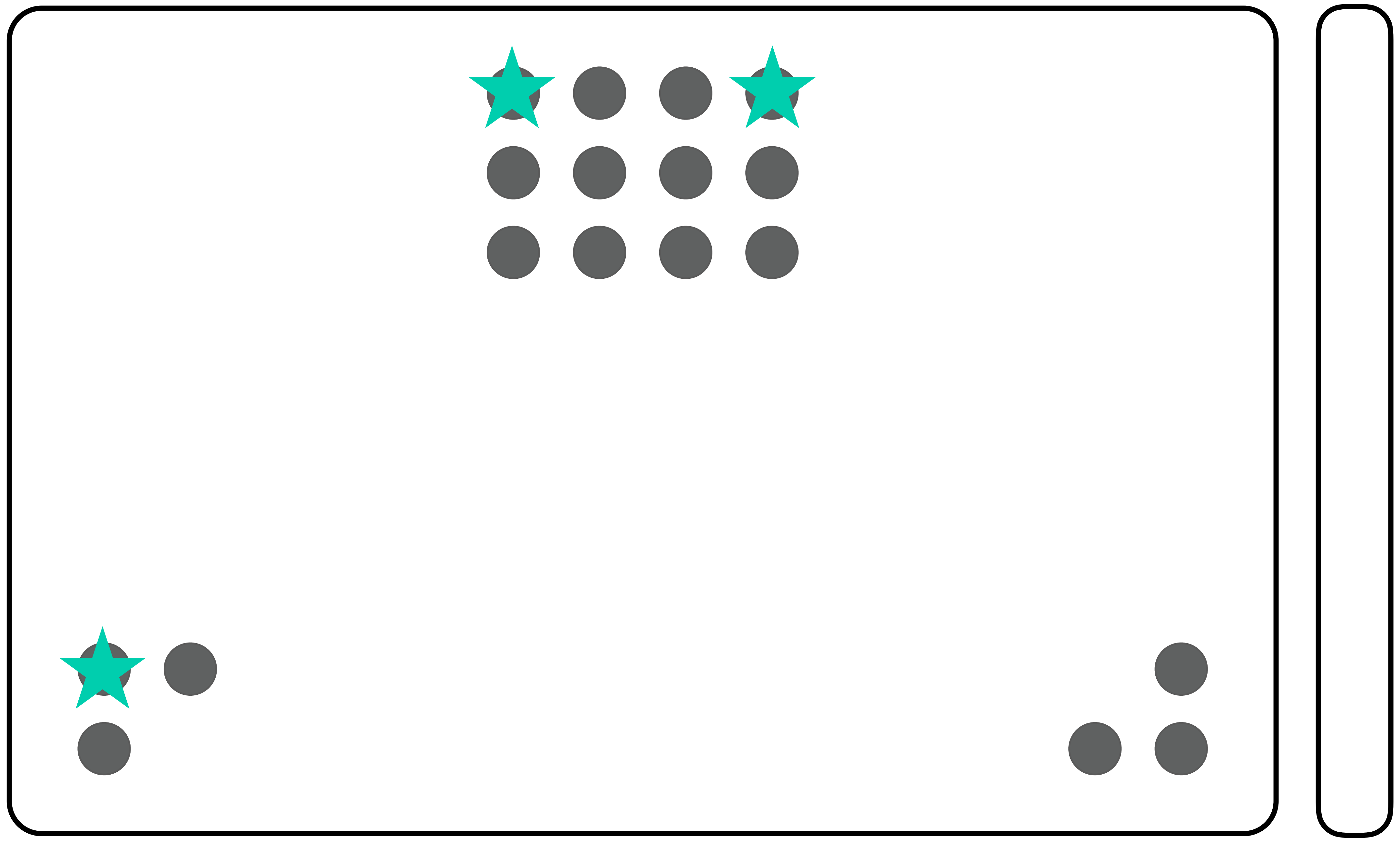
NEED FOR FAST AND GOOD SEEDINGS

Random seeding



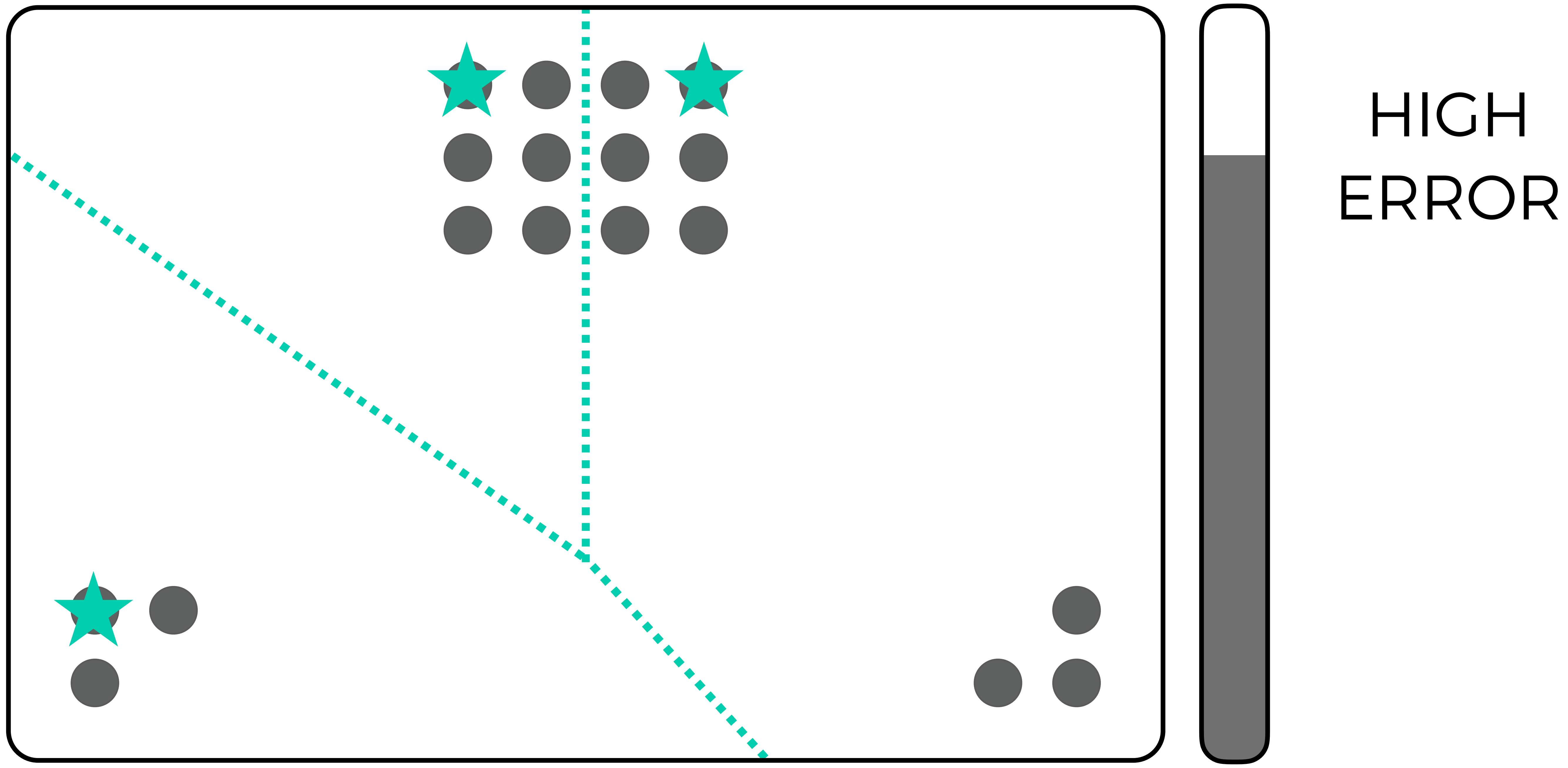
Random seeding

Sample data points ● uniformly at random as cluster centers ★

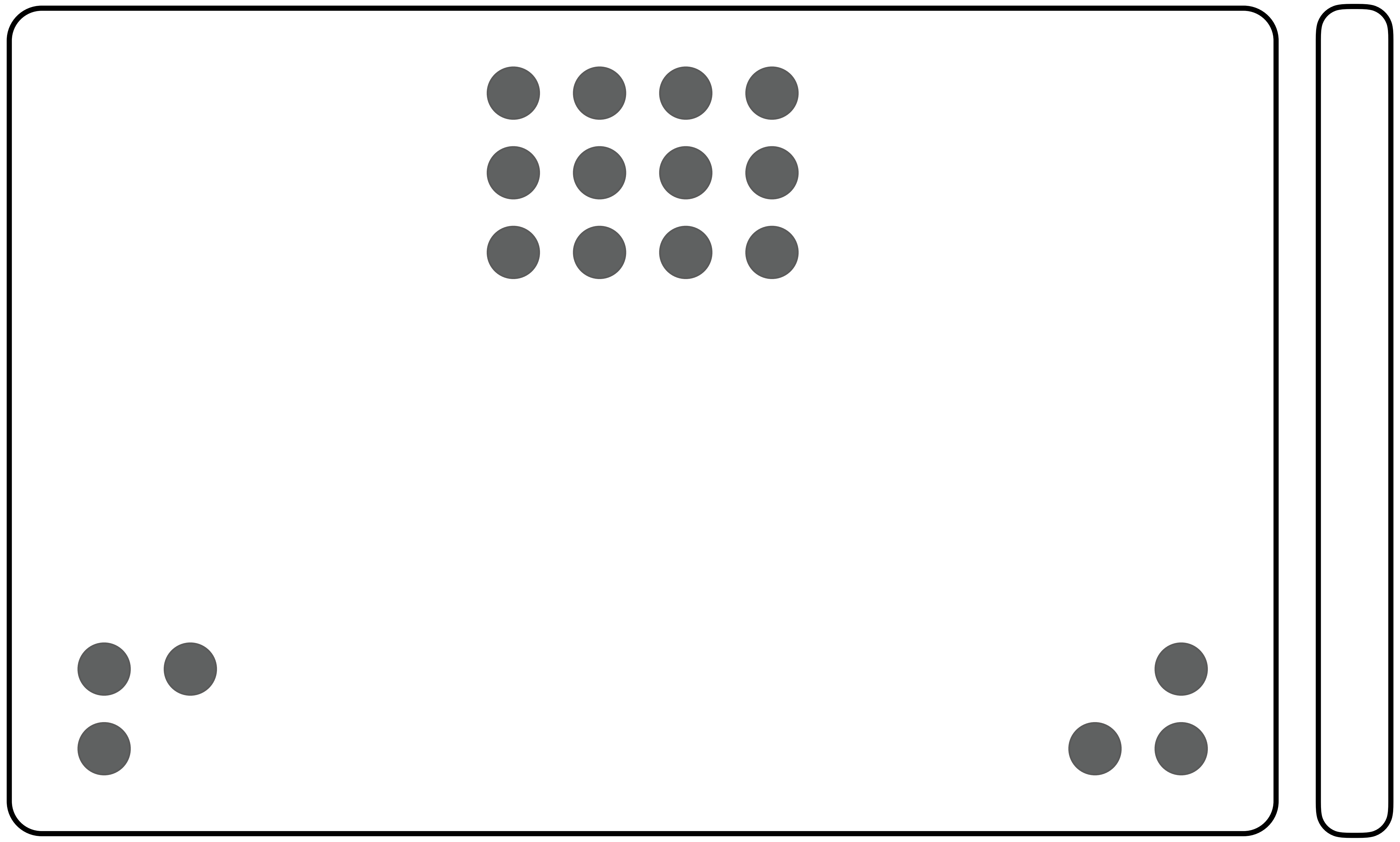


Random seeding

Sample data points uniformly at random as cluster centers

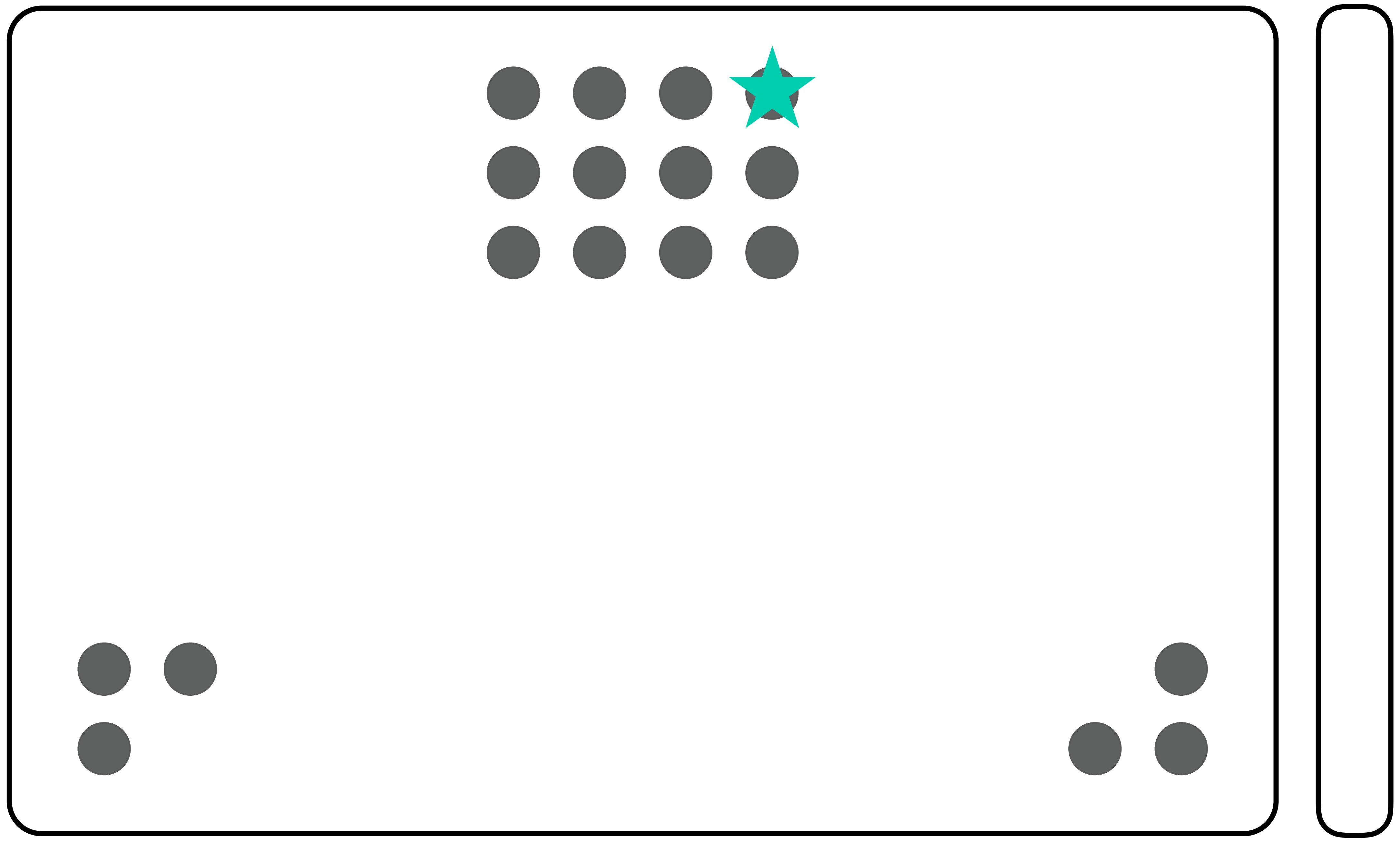


k-Means++ seeding [Arthur et al., 2007]



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

- Sample first center uniformly at random



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

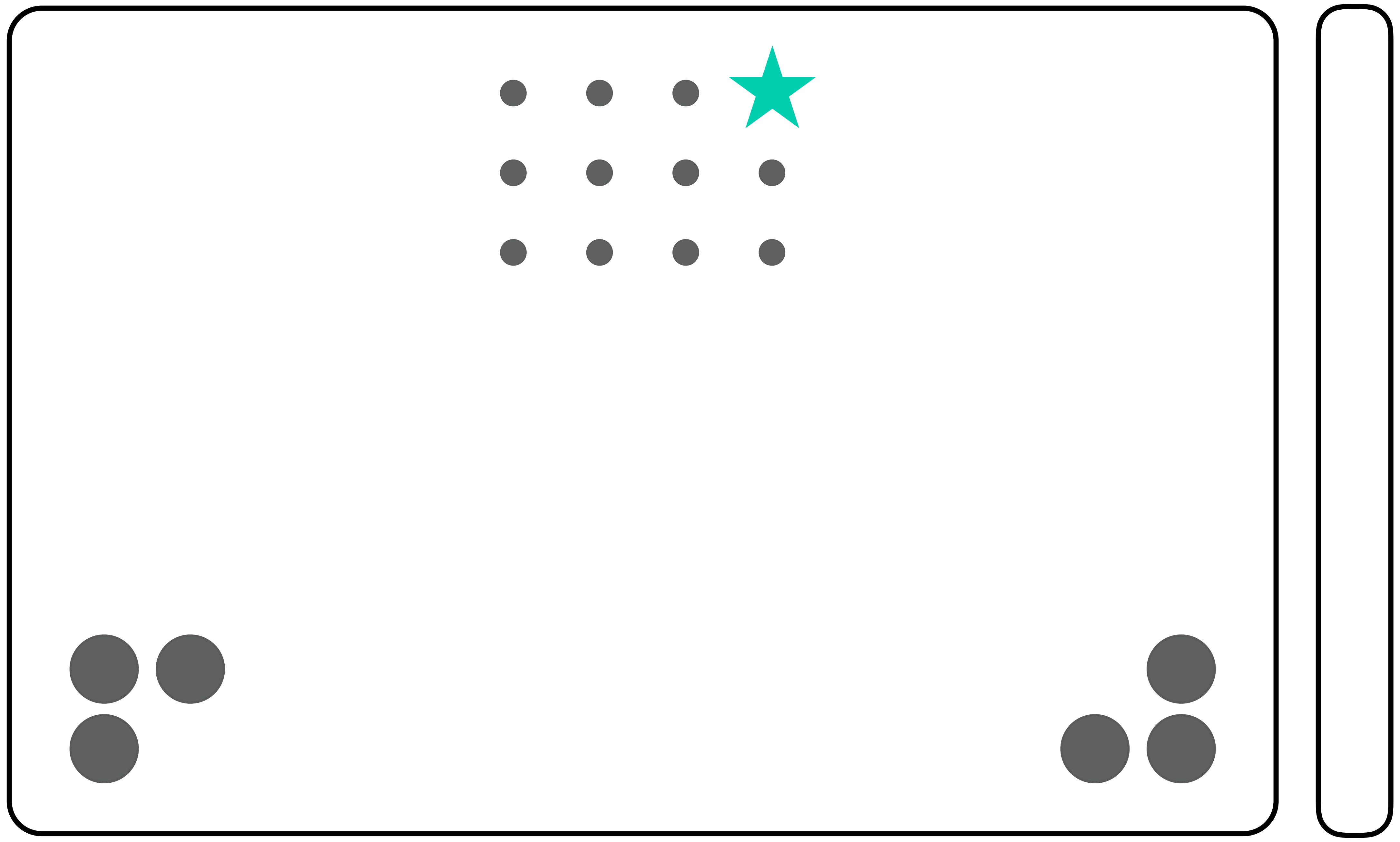
Sample first center uniformly at random

for i=2, 3, ..., k:

sample point x with

$$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

D²-SAMPLING



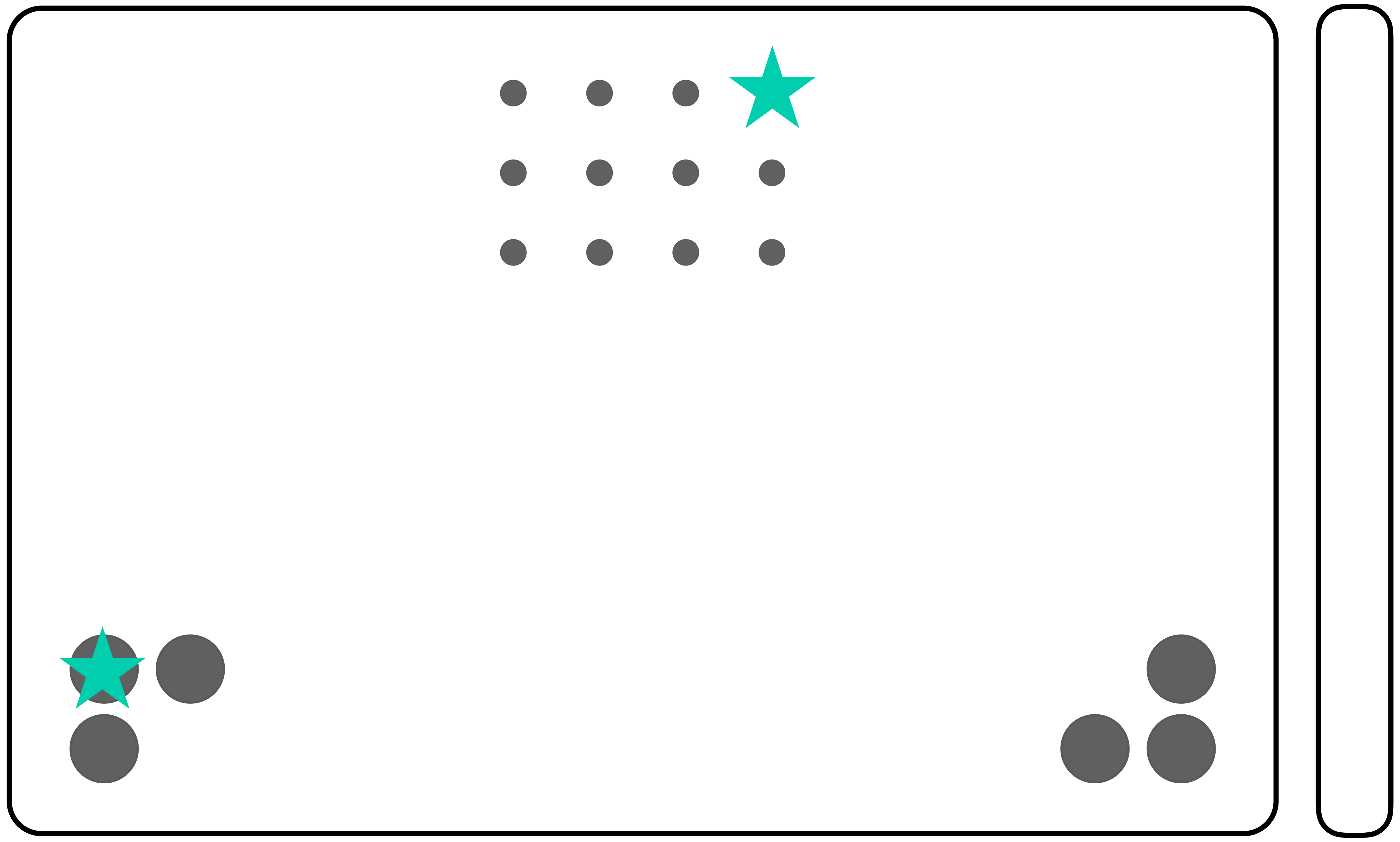
k-Means++ seeding [Arthur et al., 2007]

- Sample first center uniformly at random
- for i=2, 3, ..., k:

  sample point x with

  $$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

  D²-SAMPLING



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

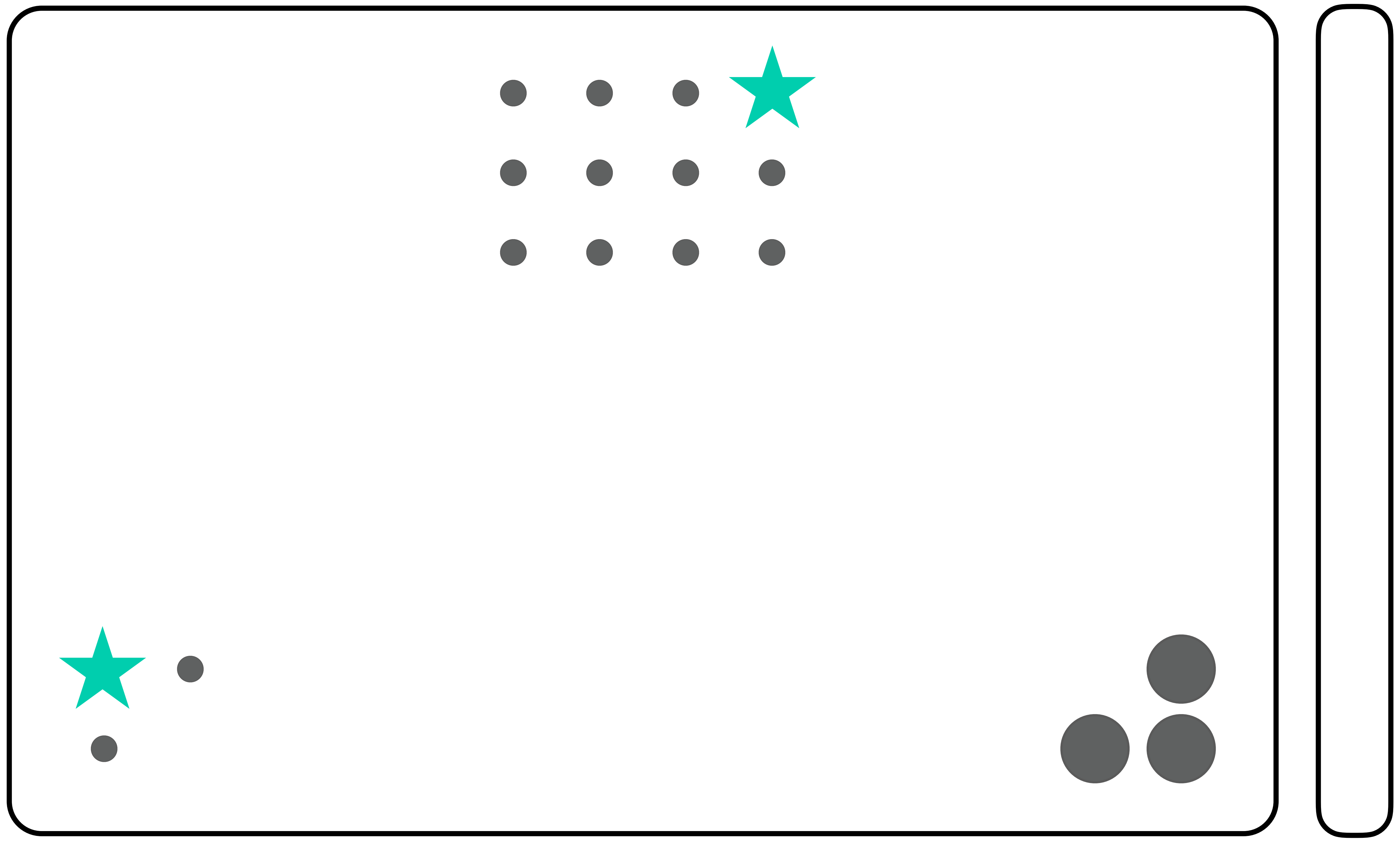
Sample first center uniformly at random

for i=2, 3, ..., k:

sample point x with

$$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

D²-SAMPLING



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

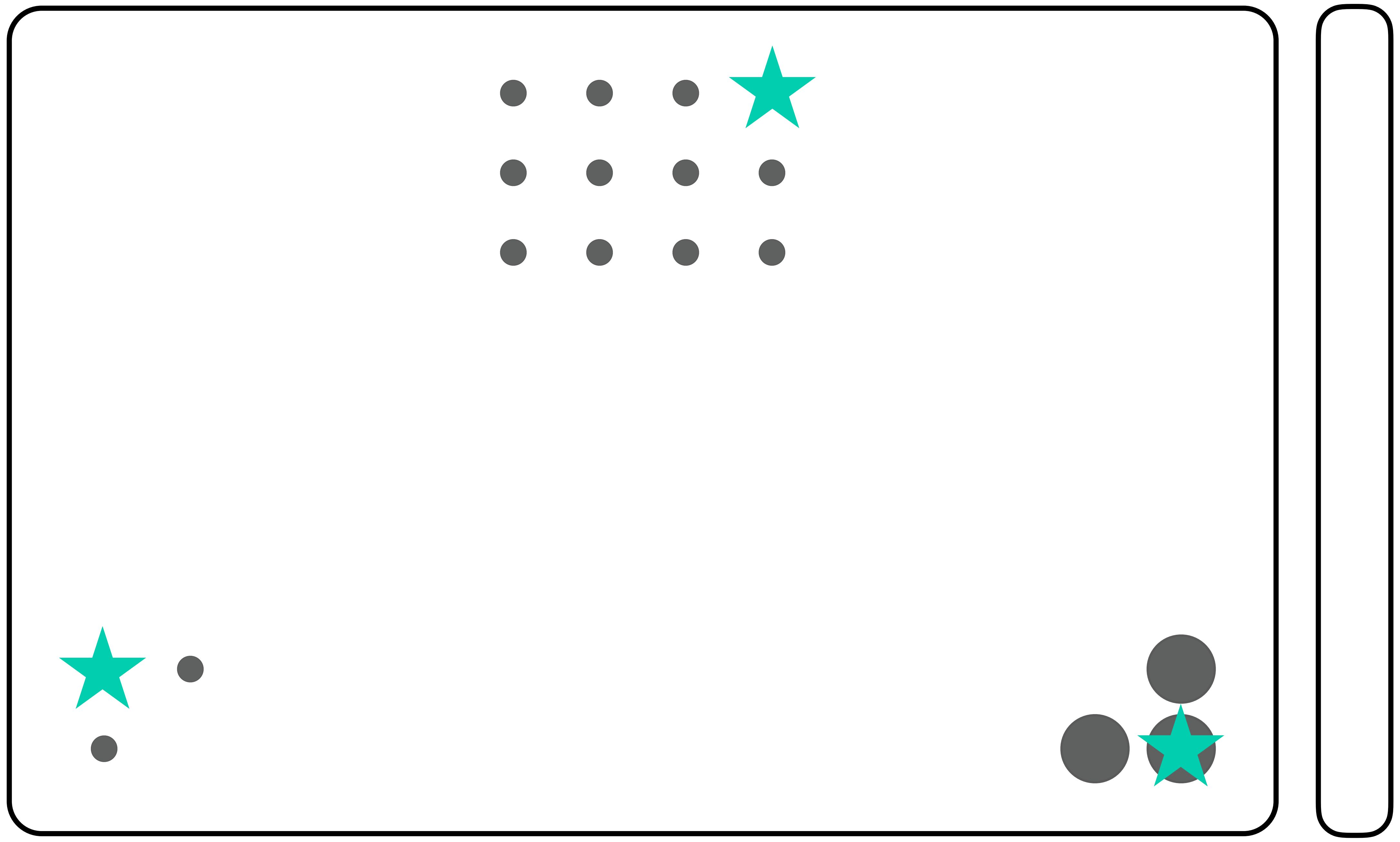
- Sample first center uniformly at random

- for i=2, 3, ..., k:

  sample point x with

  $$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

  D²-SAMPLING



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

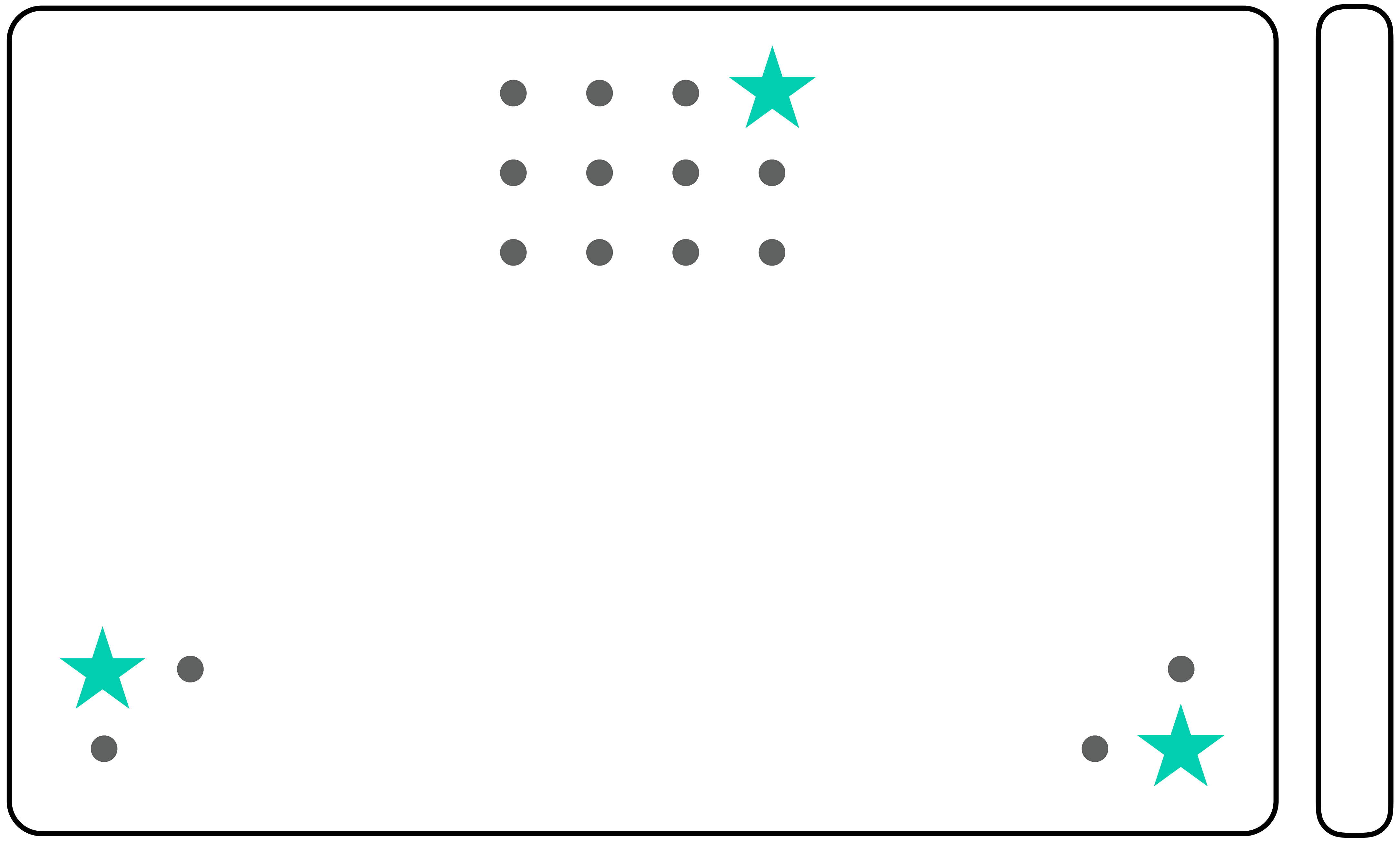
Sample first center uniformly at random

for i=2, 3, ..., k:

sample point x with

$$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

D^2-SAMPLING



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

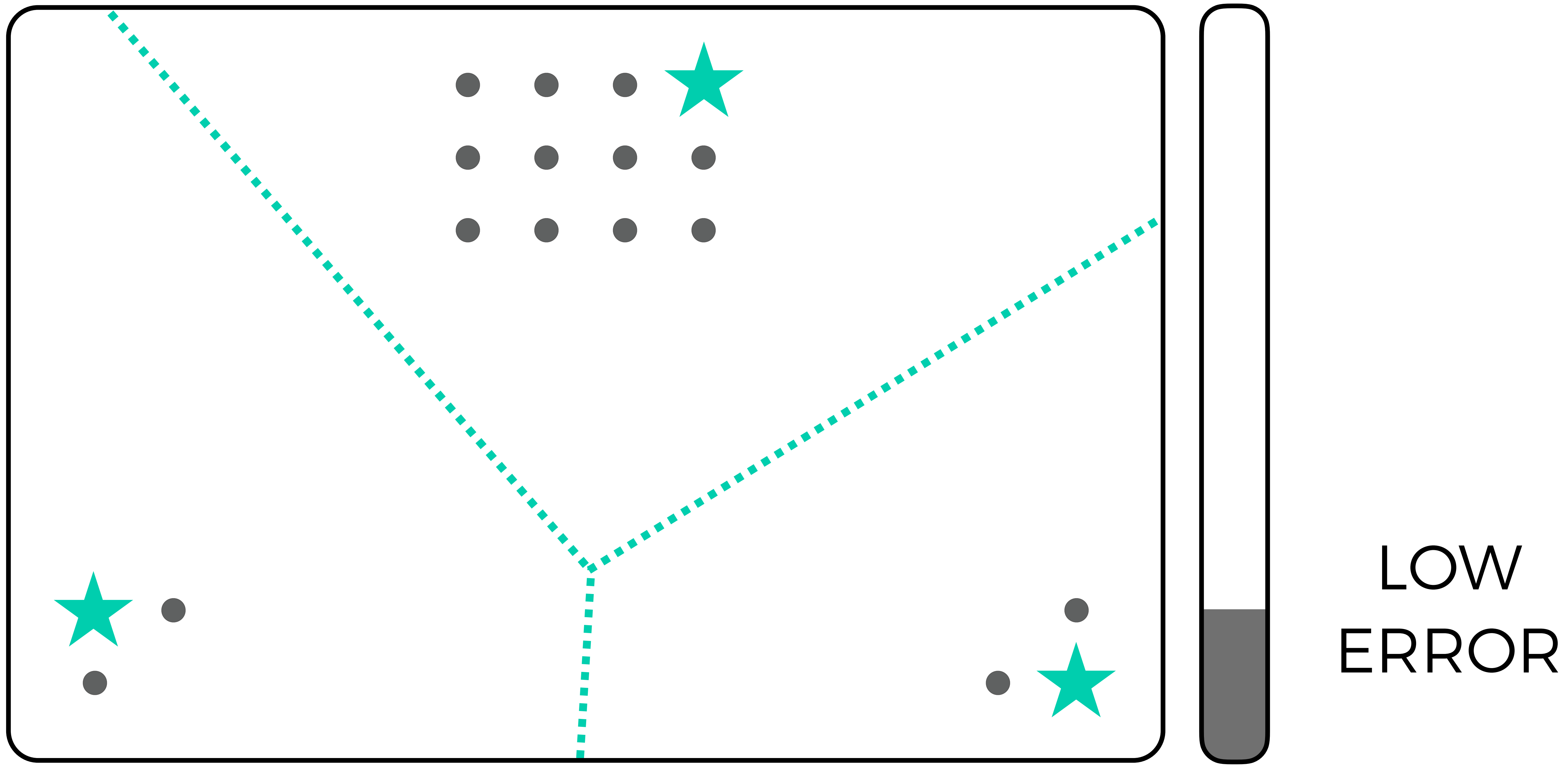
- Sample first center uniformly at random

- for i=2, 3, ..., k:

  sample point x with

  $$\boxed{p(x) \propto \mathrm{d}(x, C)^2}$$

  D²-SAMPLING



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

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

PROVABLY GOOD

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

ALREADY AFTER SEEDING

[Arthur et al., 2007]

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

PROVABLY GOOD

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

ALREADY AFTER SEEDING

[Arthur et al., 2007]

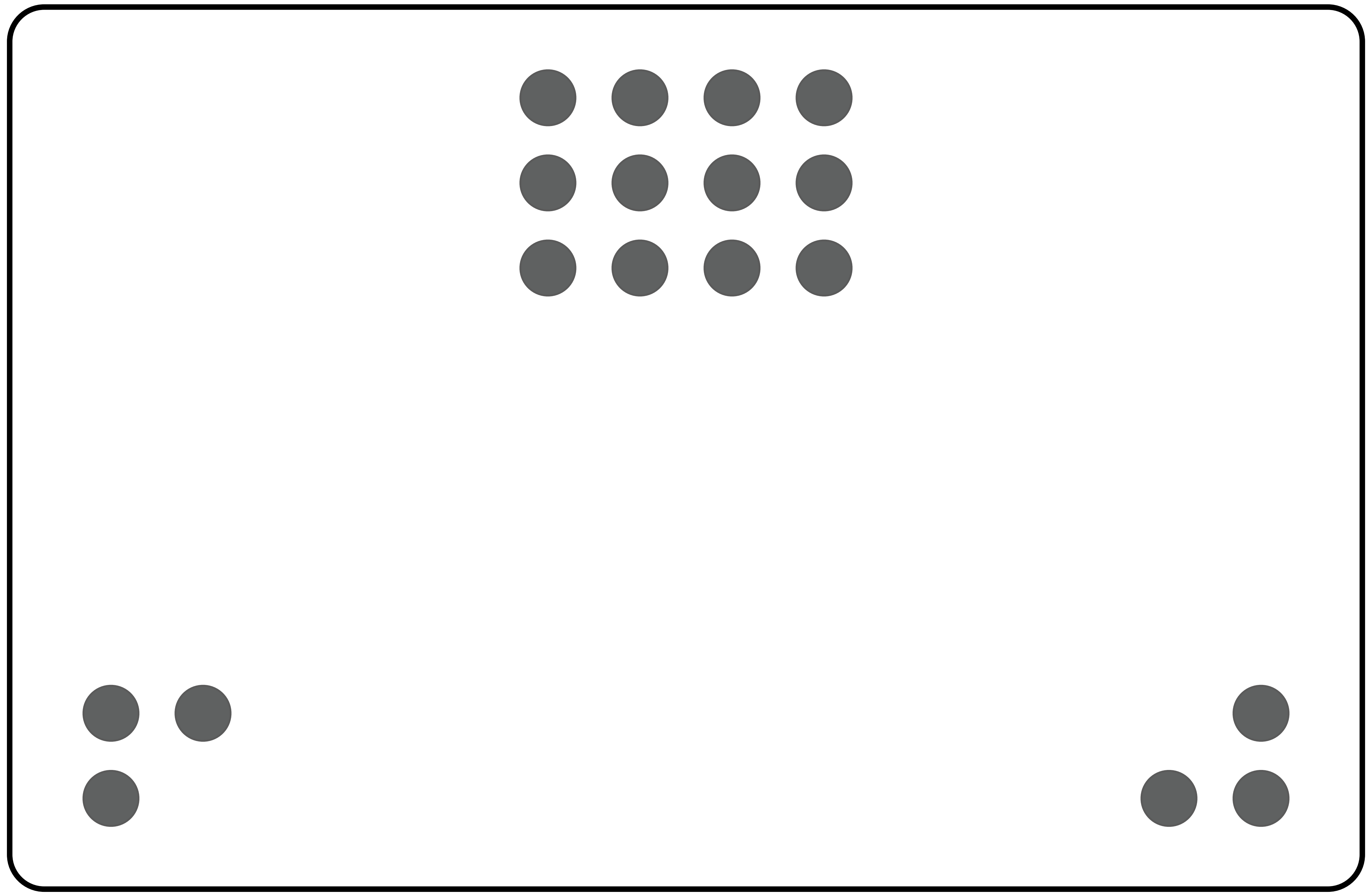
BUT SLOW

$$\boxed{\mathcal{O}(nkd)}$$

HARD TO PARALLELISE

k sequential passes!

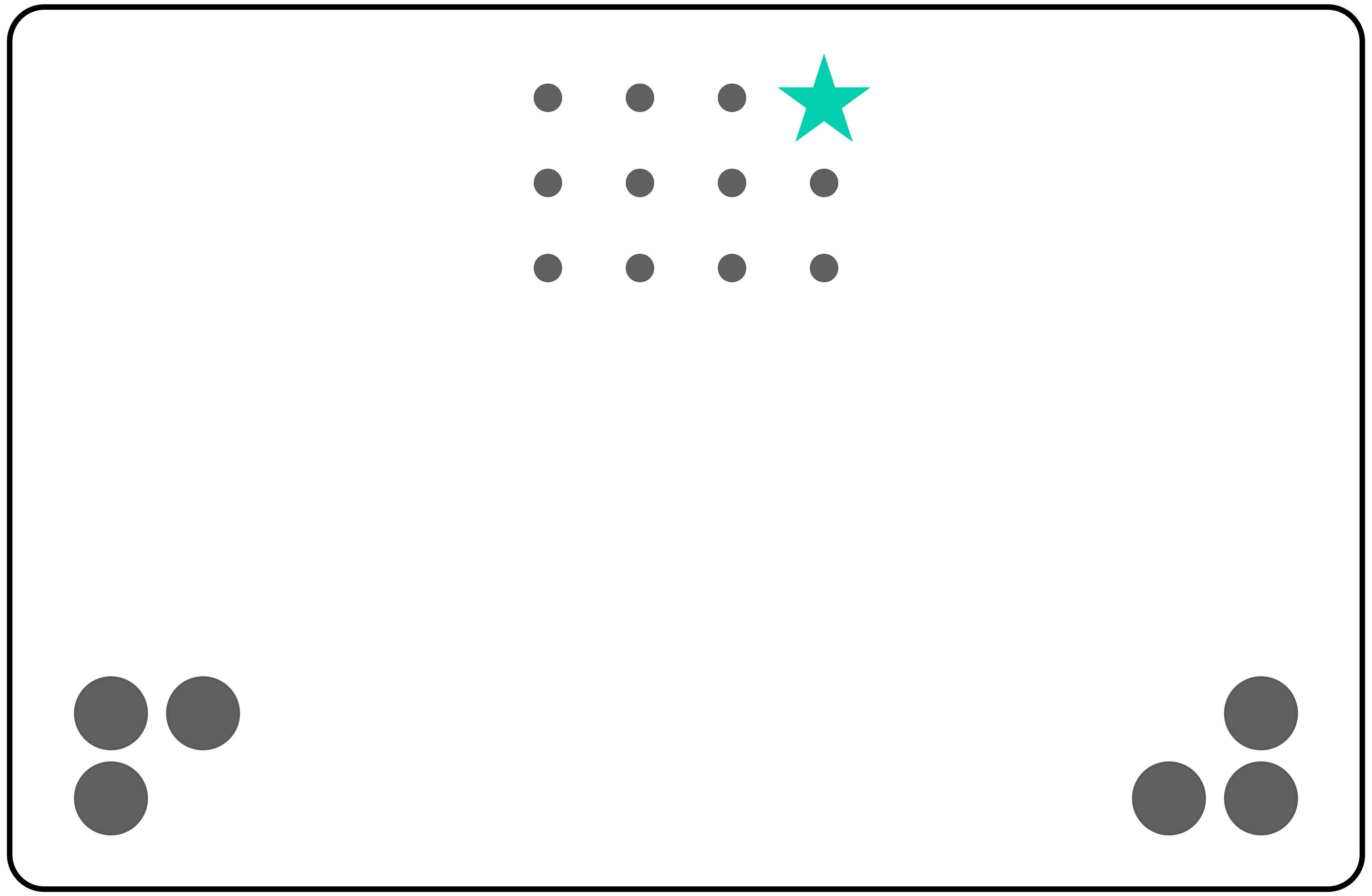
Single round of D^2-sampling



Single round of D²-sampling

- Sample each point with probability

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

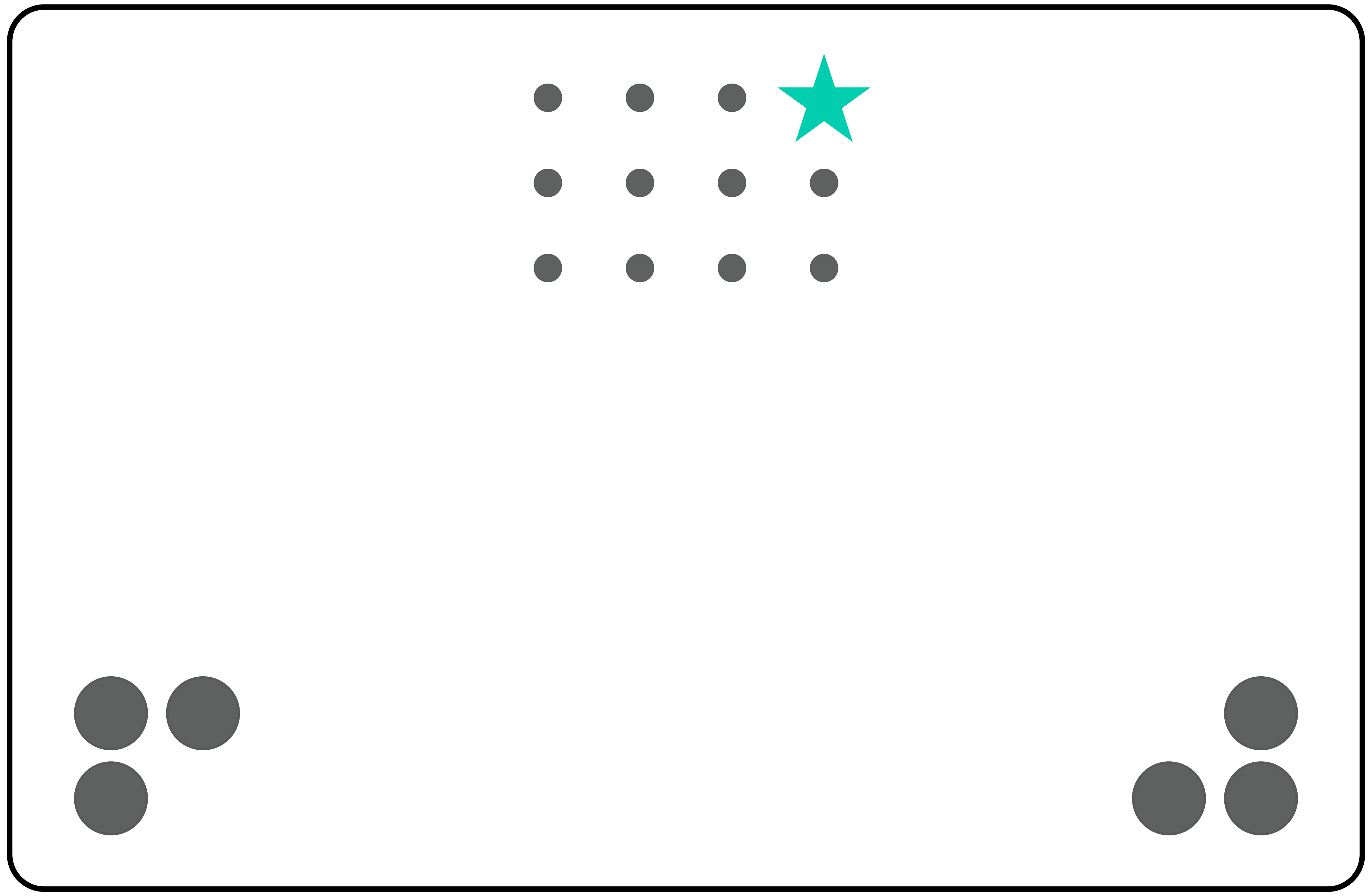


Single round of D^2-sampling

Sample each point with probability

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

REQUIRES LINEAR PASS



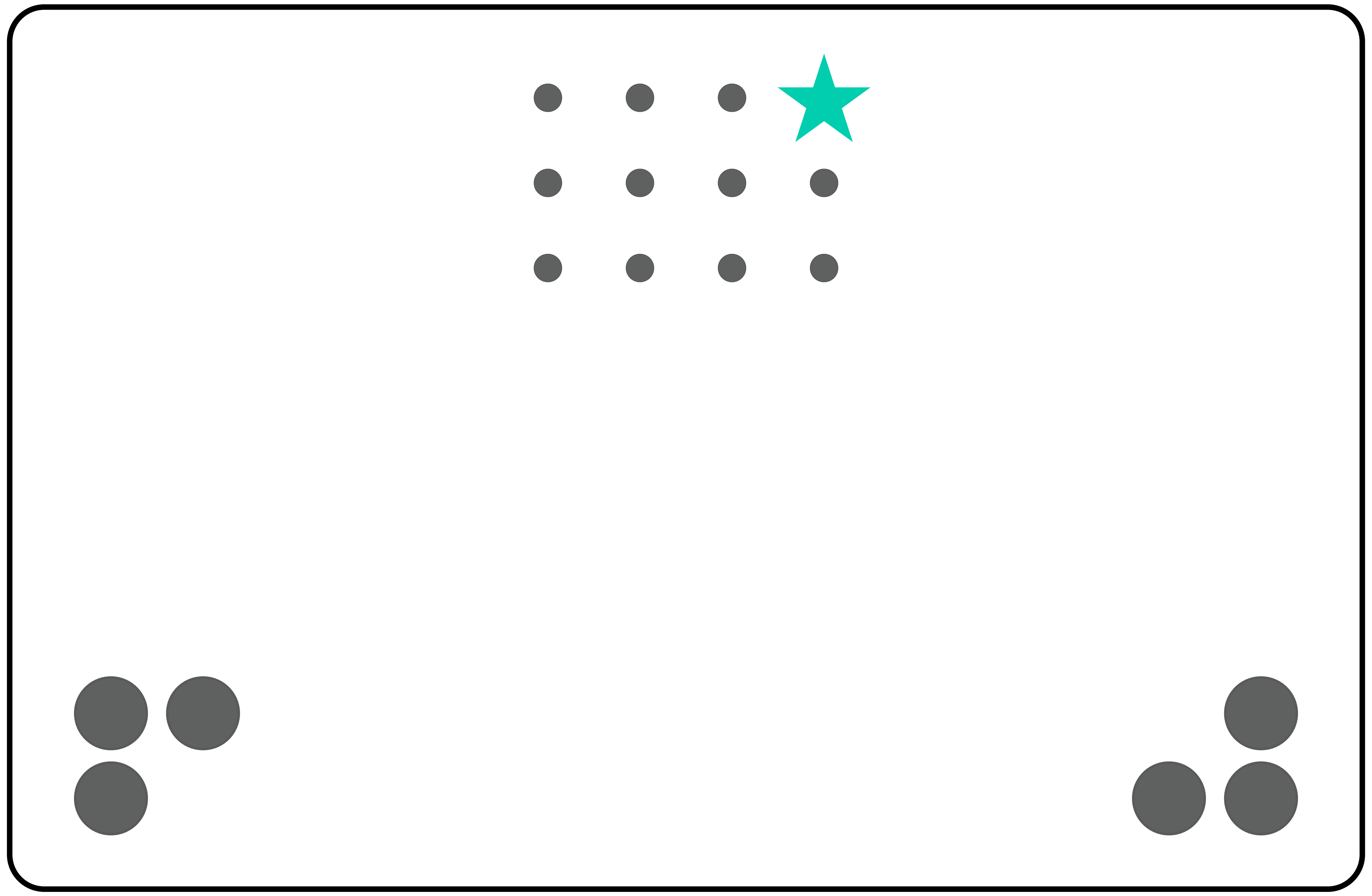
Single round of D²-sampling

- Sample each point with probability

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

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 → RETURN CENTER

Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states ⇨ RETURN CENTER
- whose stationary distribution is

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states ⇨ RETURN CENTER
- whose stationary distribution is ⇨ GUARANTEES **APPROXIMATION**

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

Markov chain Monte Carlo approach

Goal: construct a Markov chain

- where data points are states ⇒ RETURN CENTER
- whose stationary distribution is ⇒ GUARANTEES **APPROXIMATION**

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

- with a fast mixing time

Markov chain Monte Carlo approach

Goal: construct a Markov chain

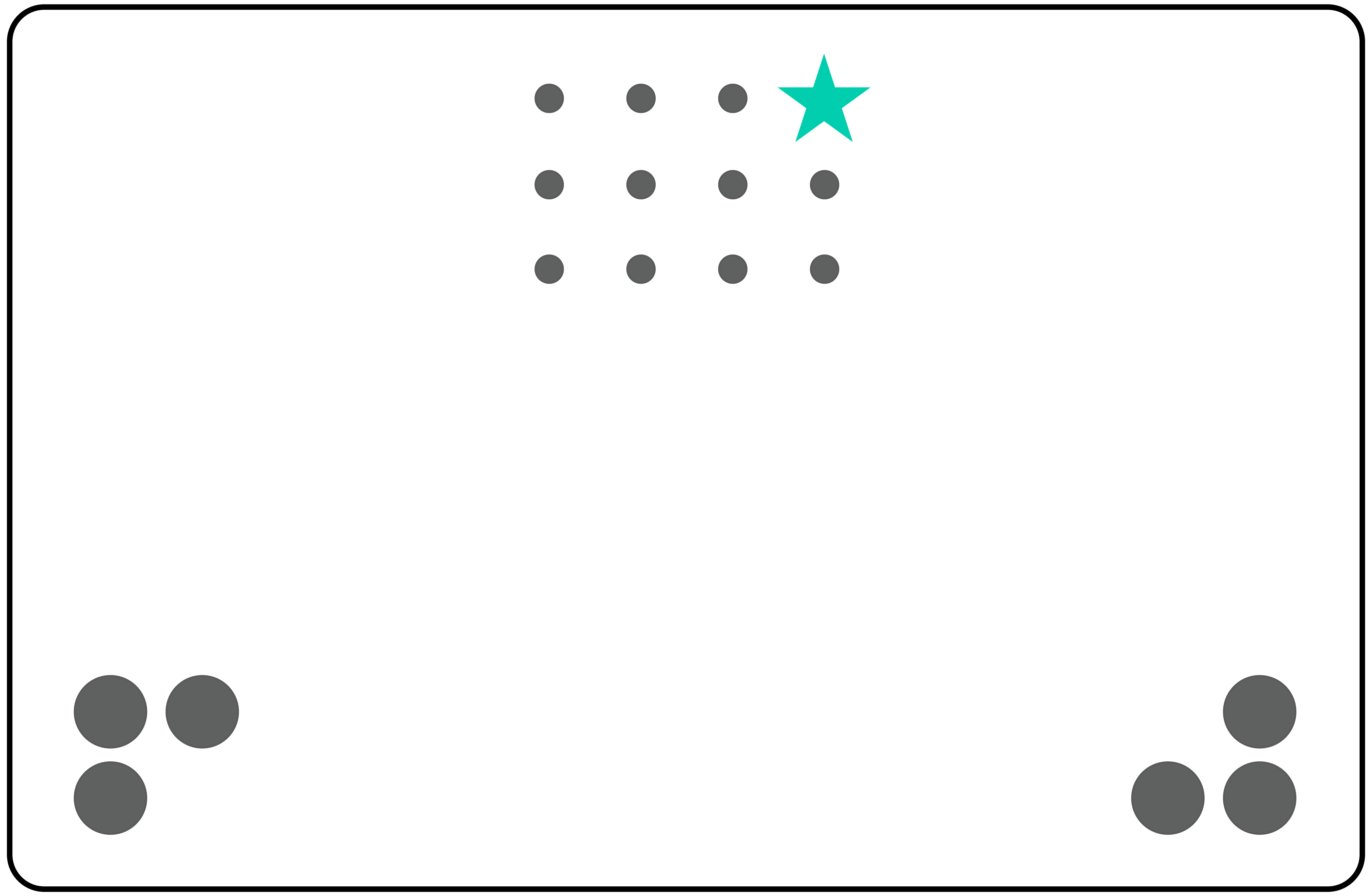
- where data points are states ⇨ RETURN CENTER
- whose stationary distribution is

$$p(x) = \frac{\mathrm{d}(x, C)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', C)^2}$$

⇨ GUARANTEES **APPROXIMATION**

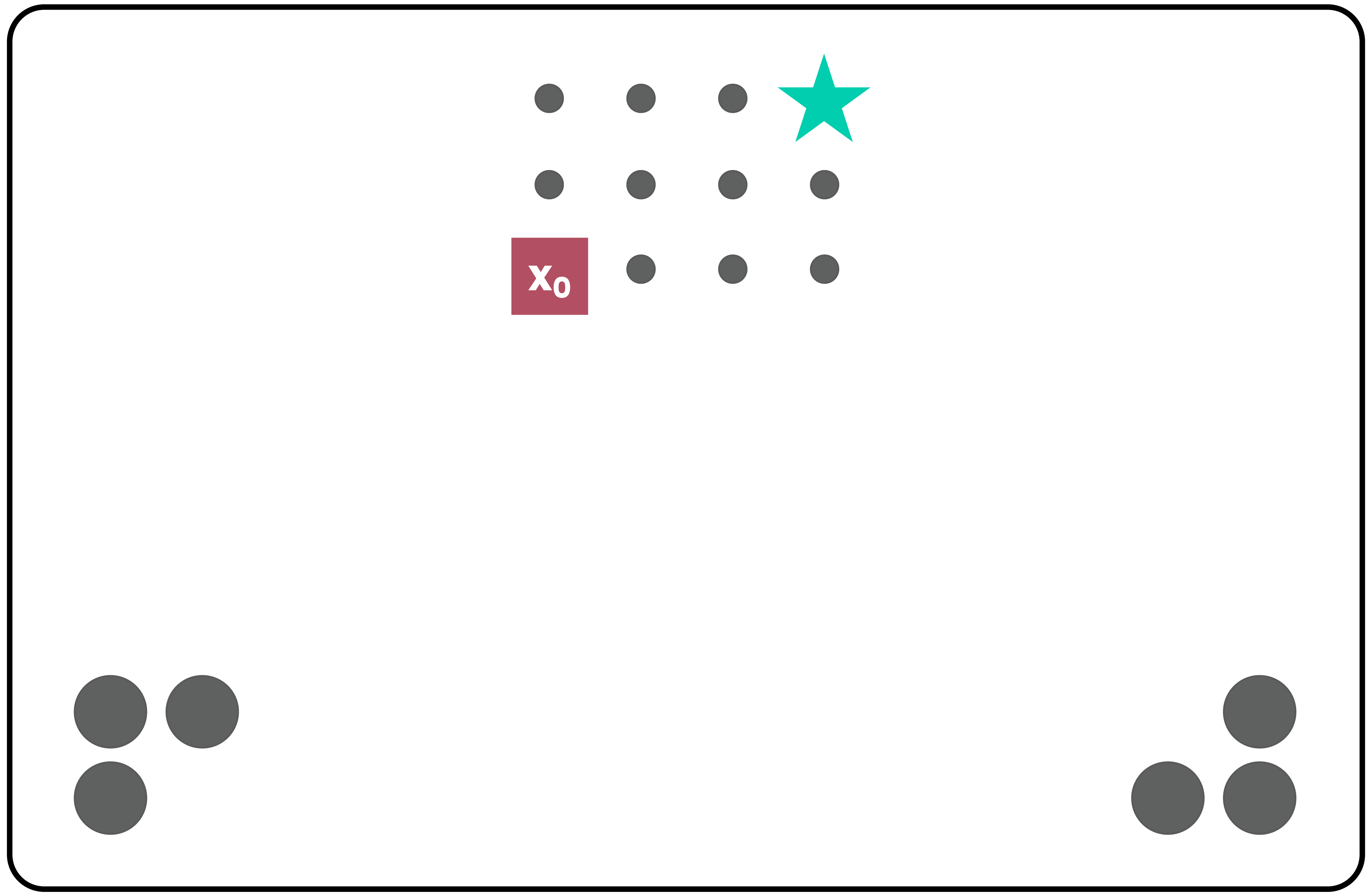
- with a fast mixing time ⇨ GUARANTEES **EFFICIENCY**

Markov chain construction

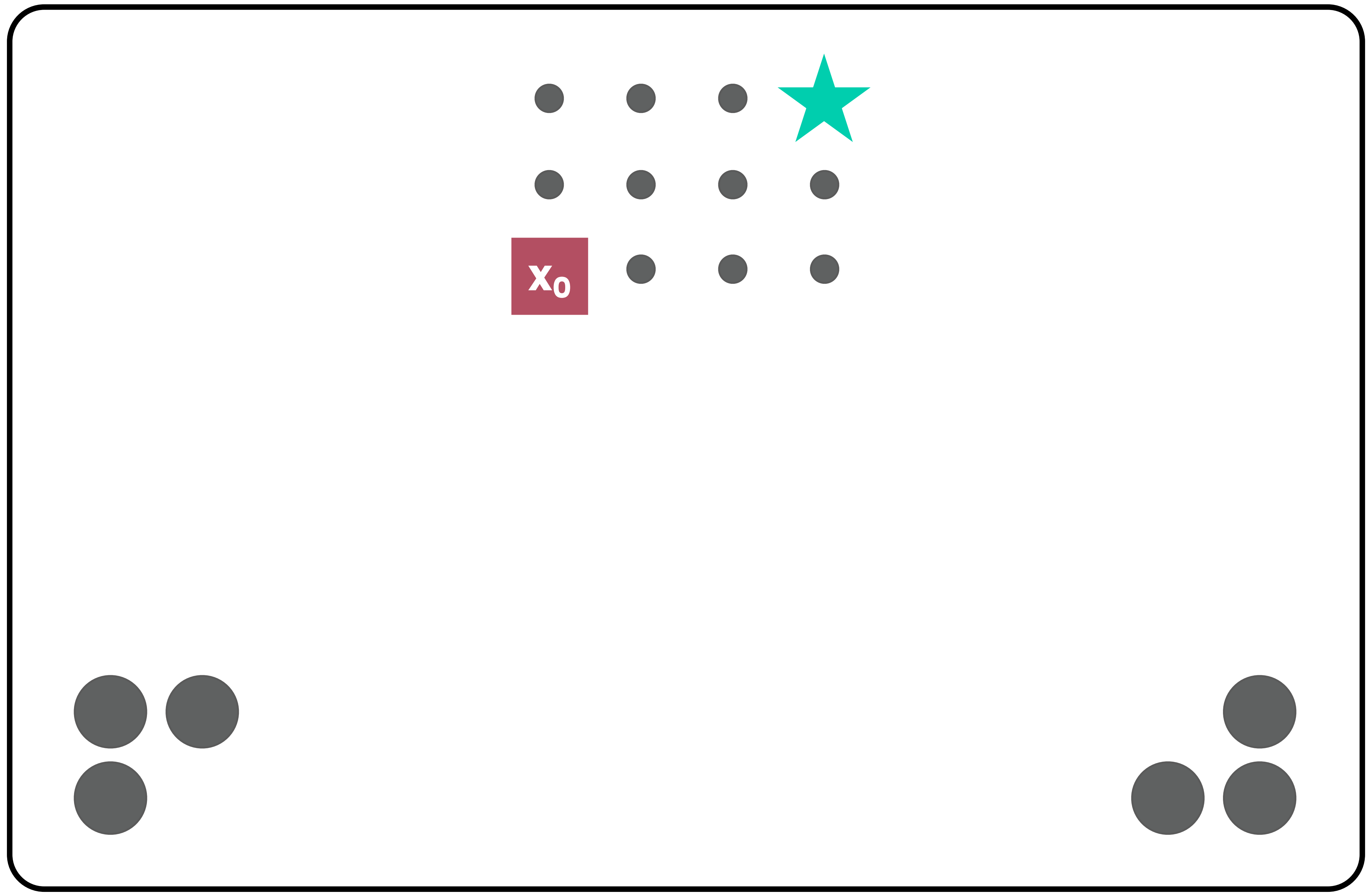


Markov chain construction

Start with an arbitrary initial state x_0

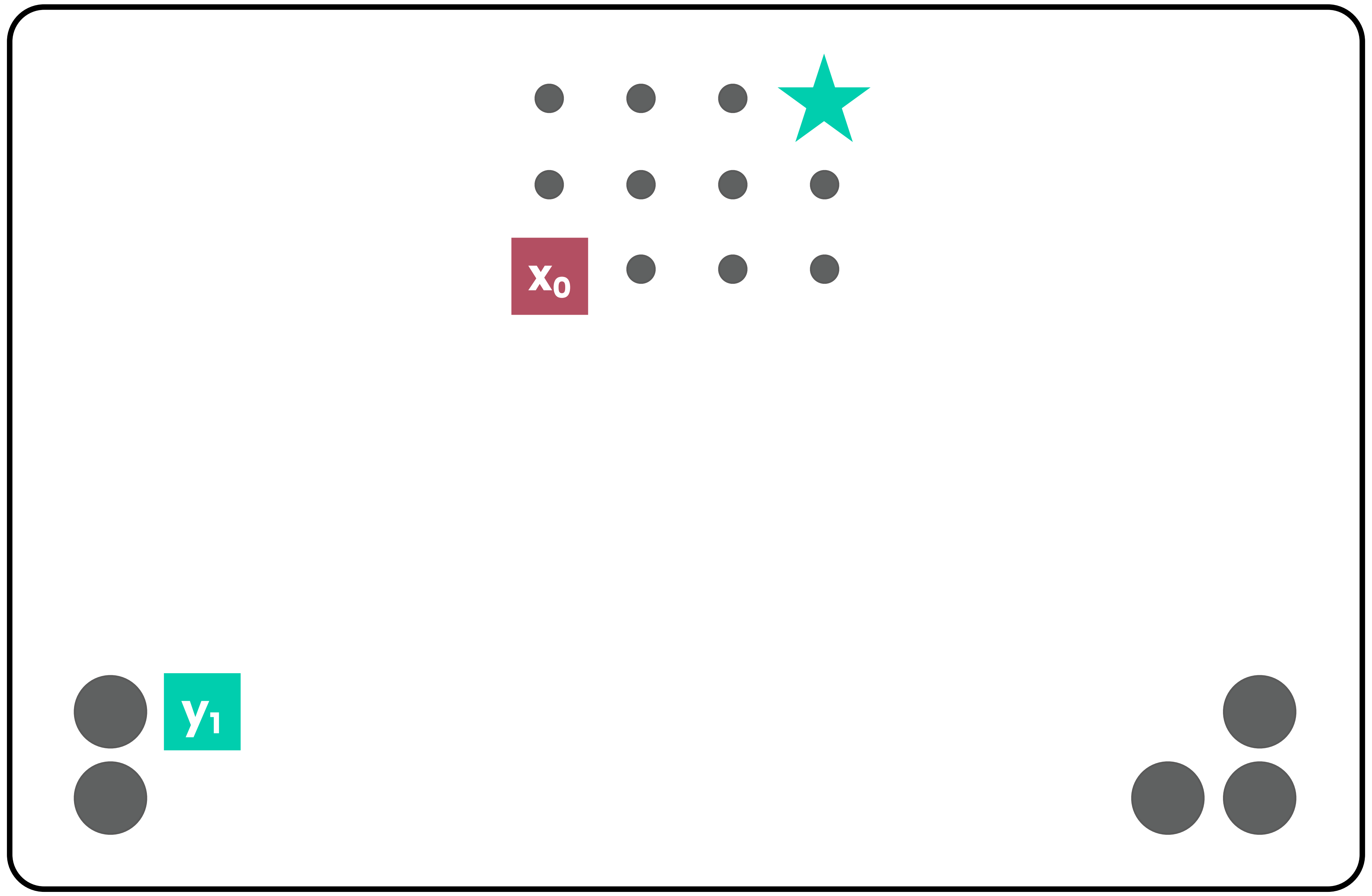


Markov chain construction



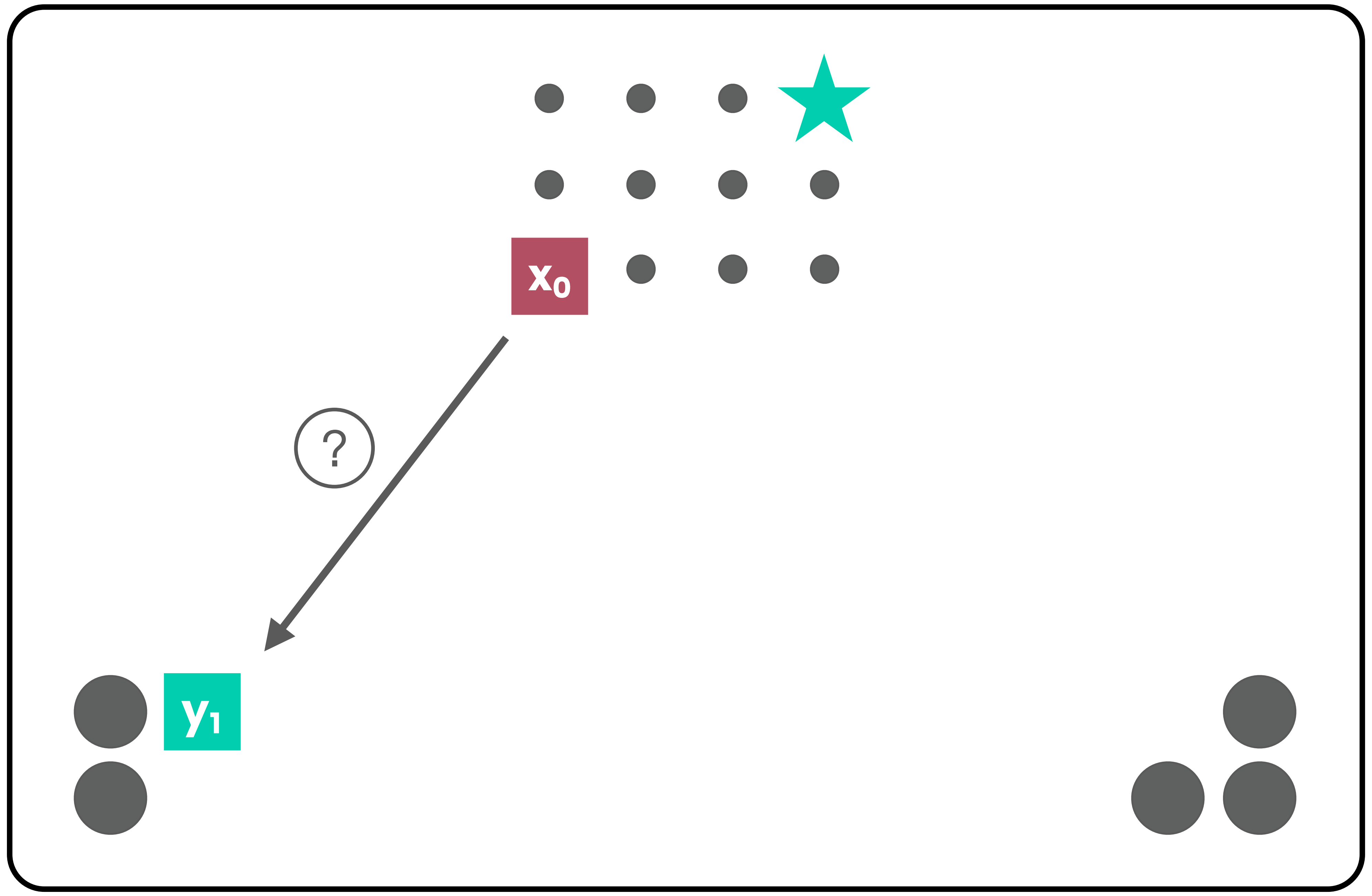
Markov chain construction

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



Markov chain construction

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



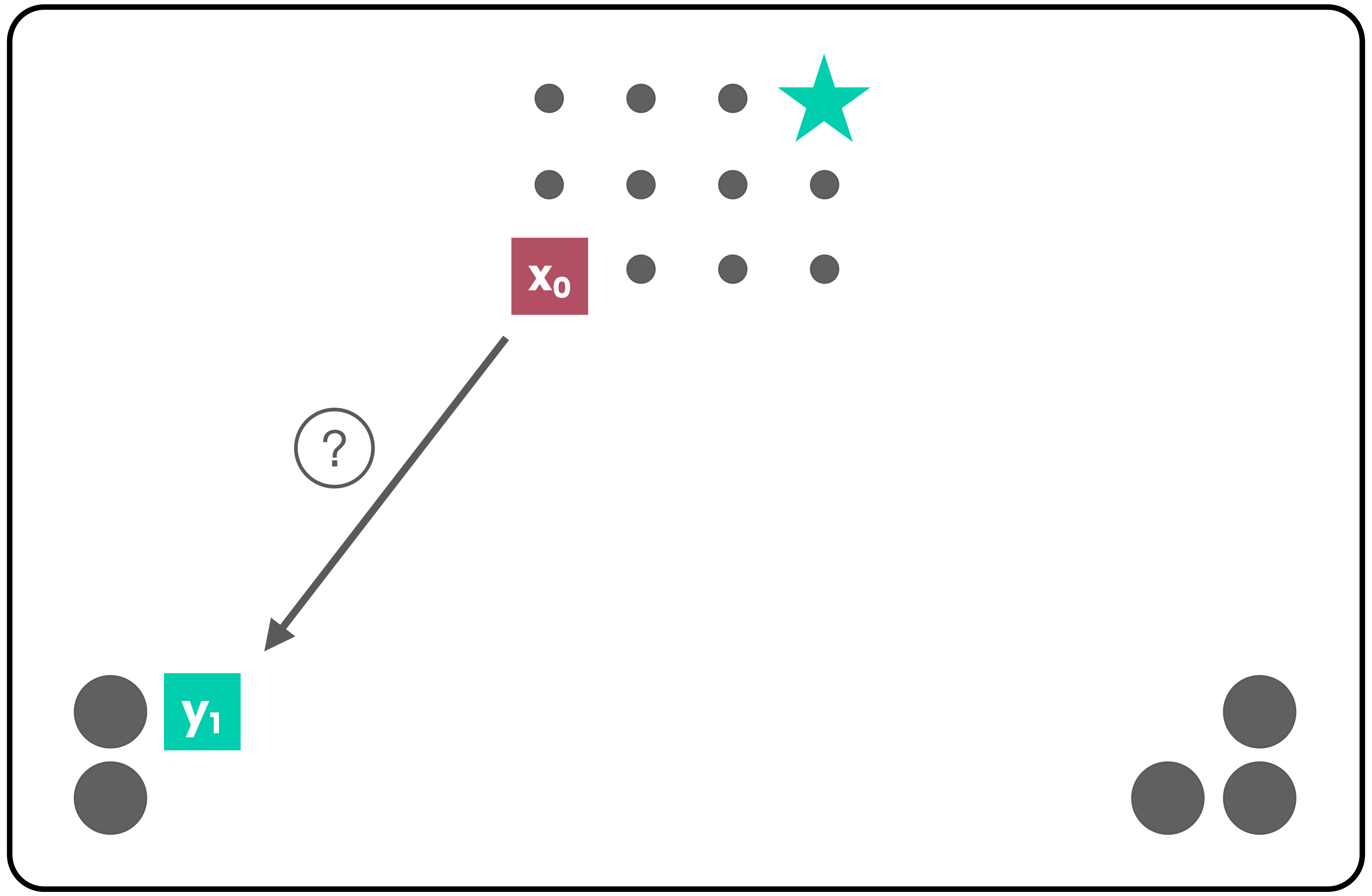
Markov chain construction

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

Set x_i = y_i with probability

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

otherwise keep x_i = x_{i-1}



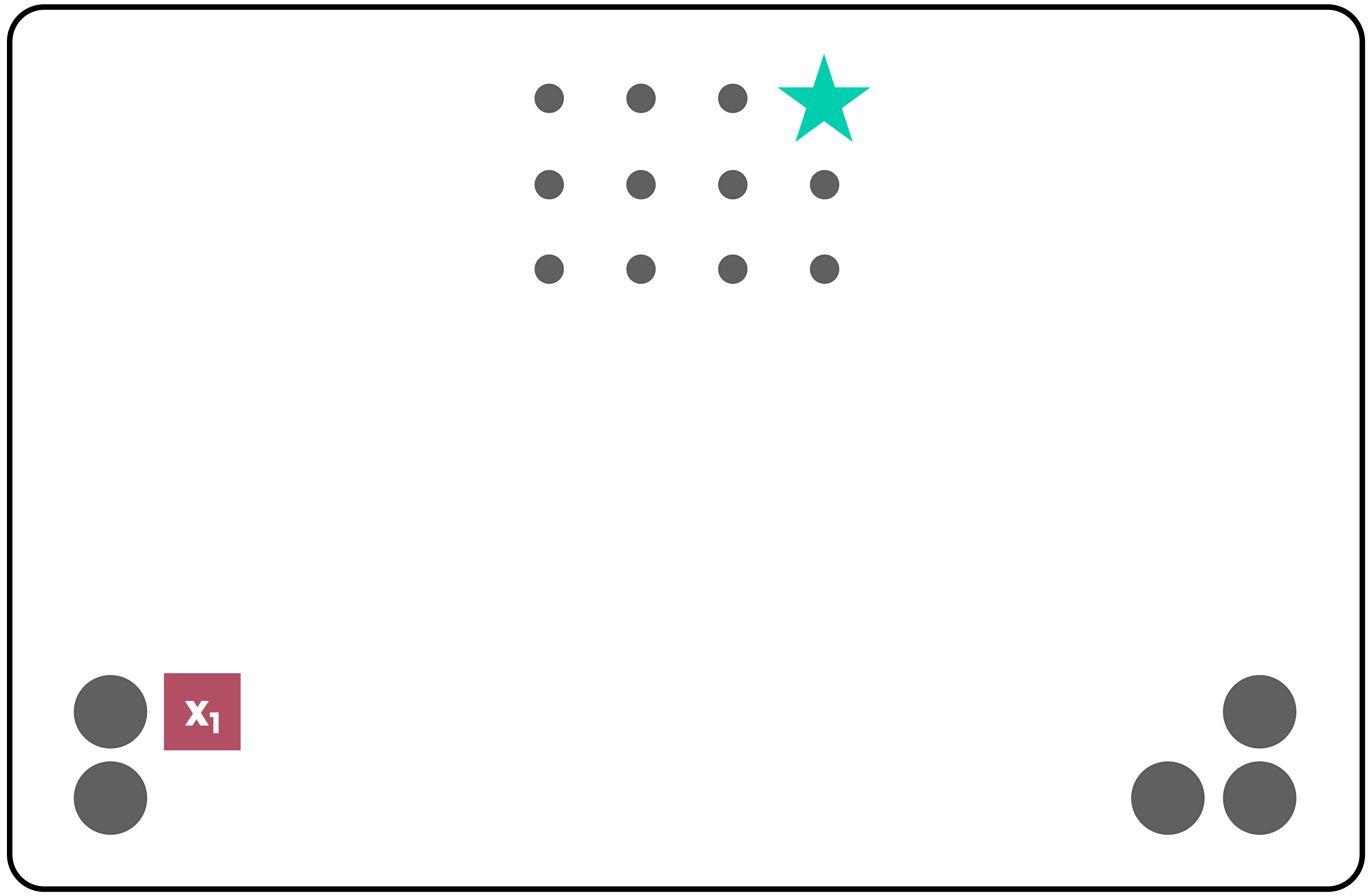
Markov chain construction

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

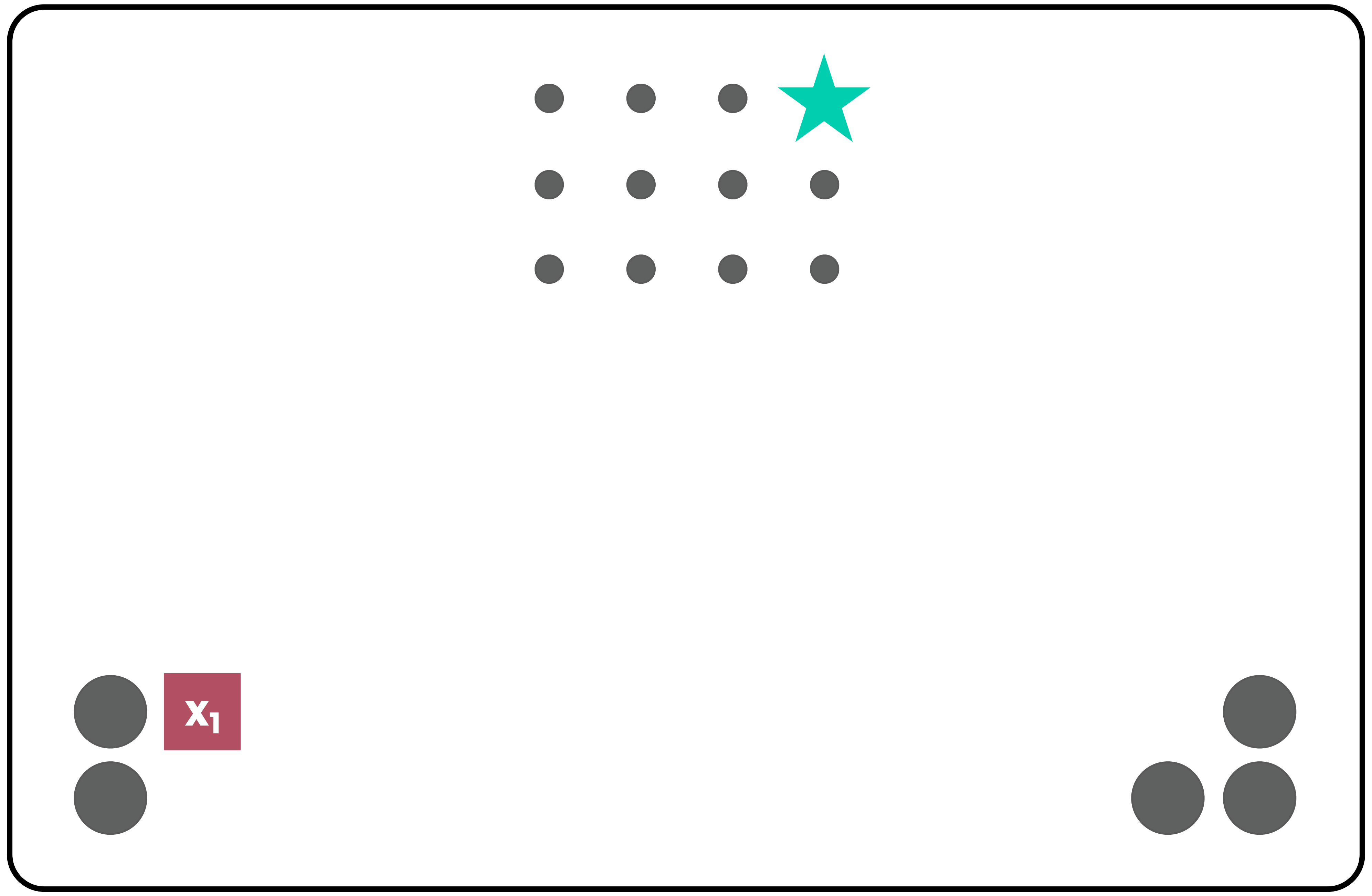
- Set x_i = y_i with probability

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

otherwise keep x_i = x_{i-1}

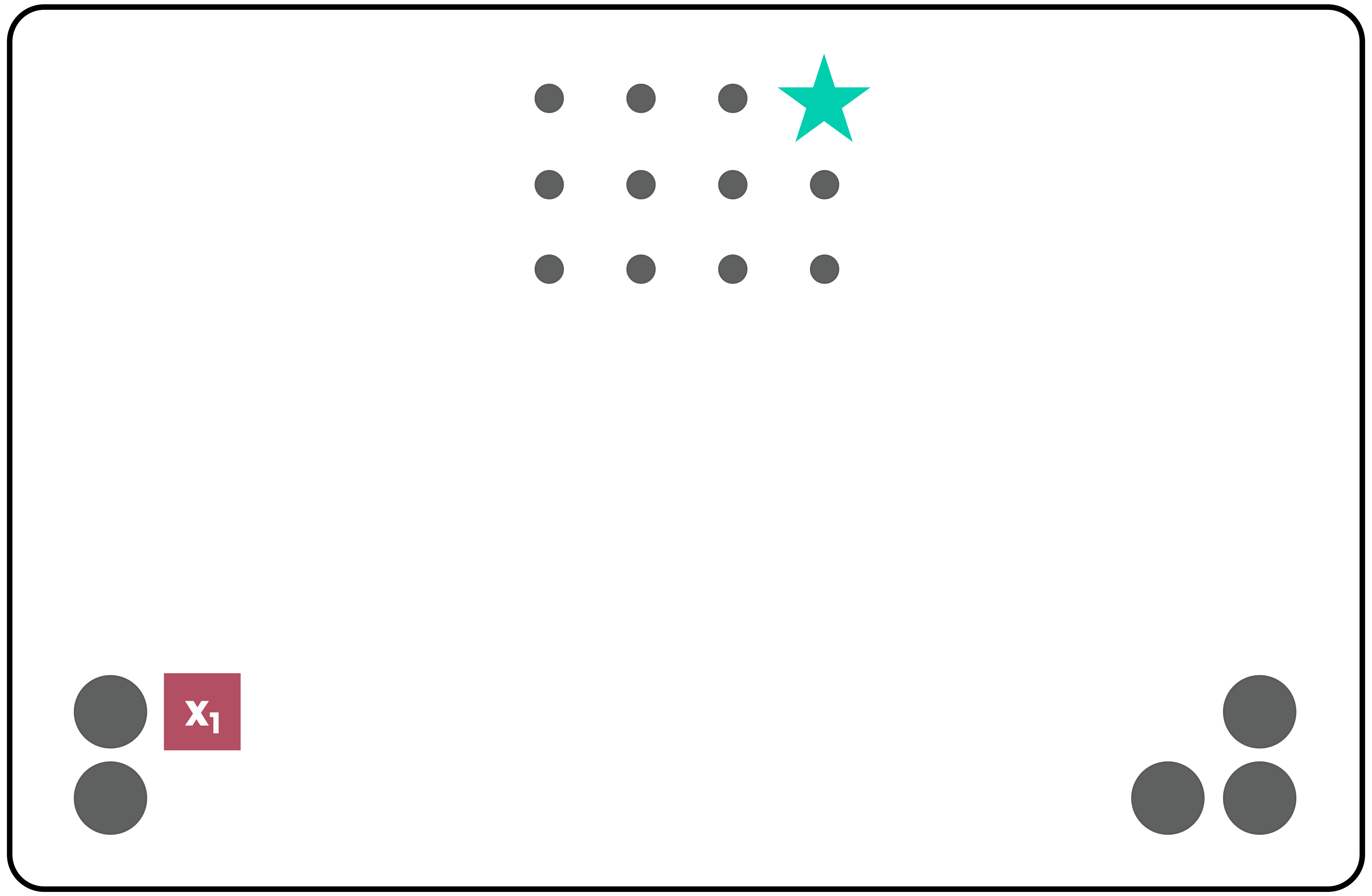


Markov chain construction



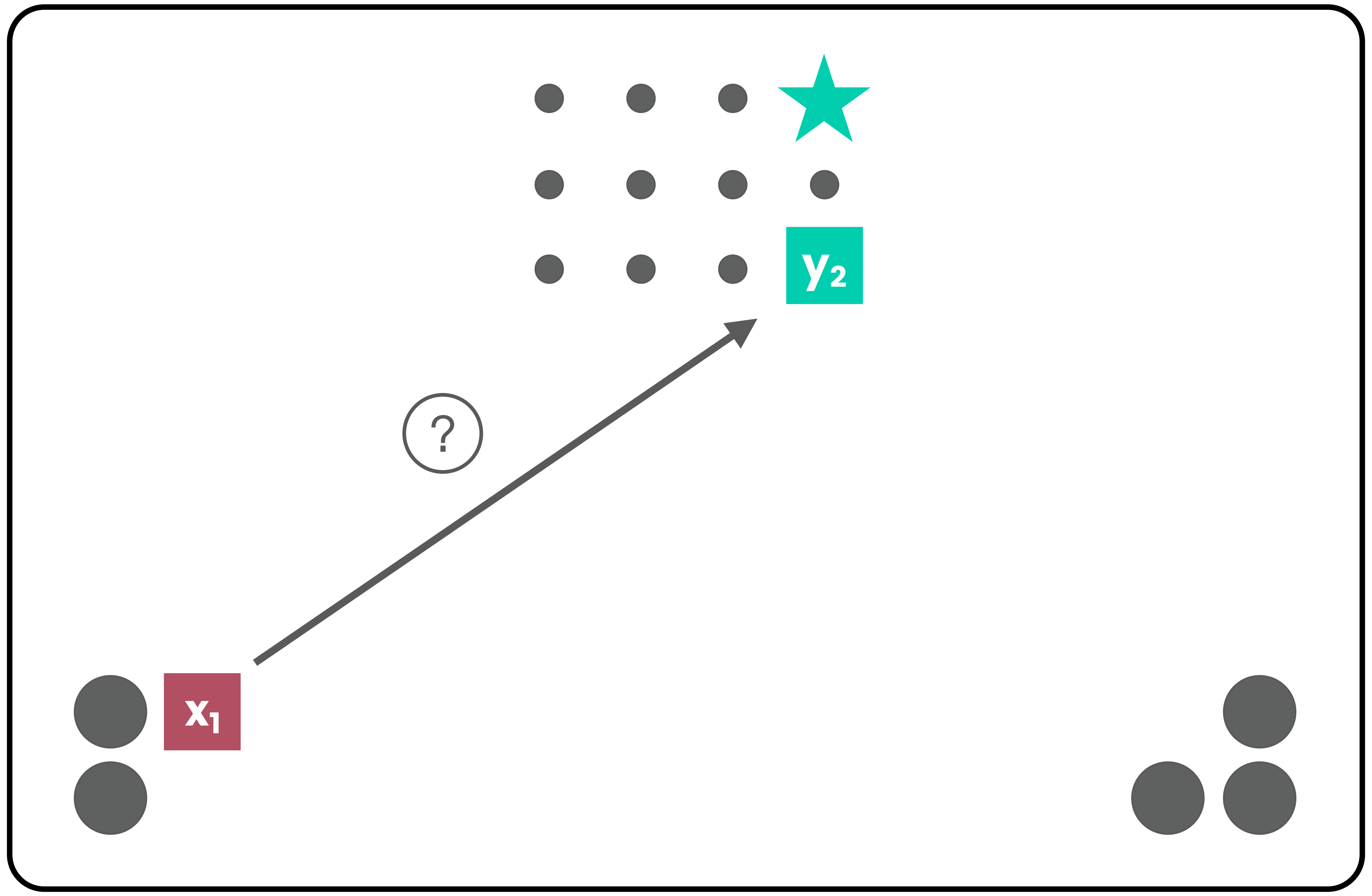
Markov chain construction

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



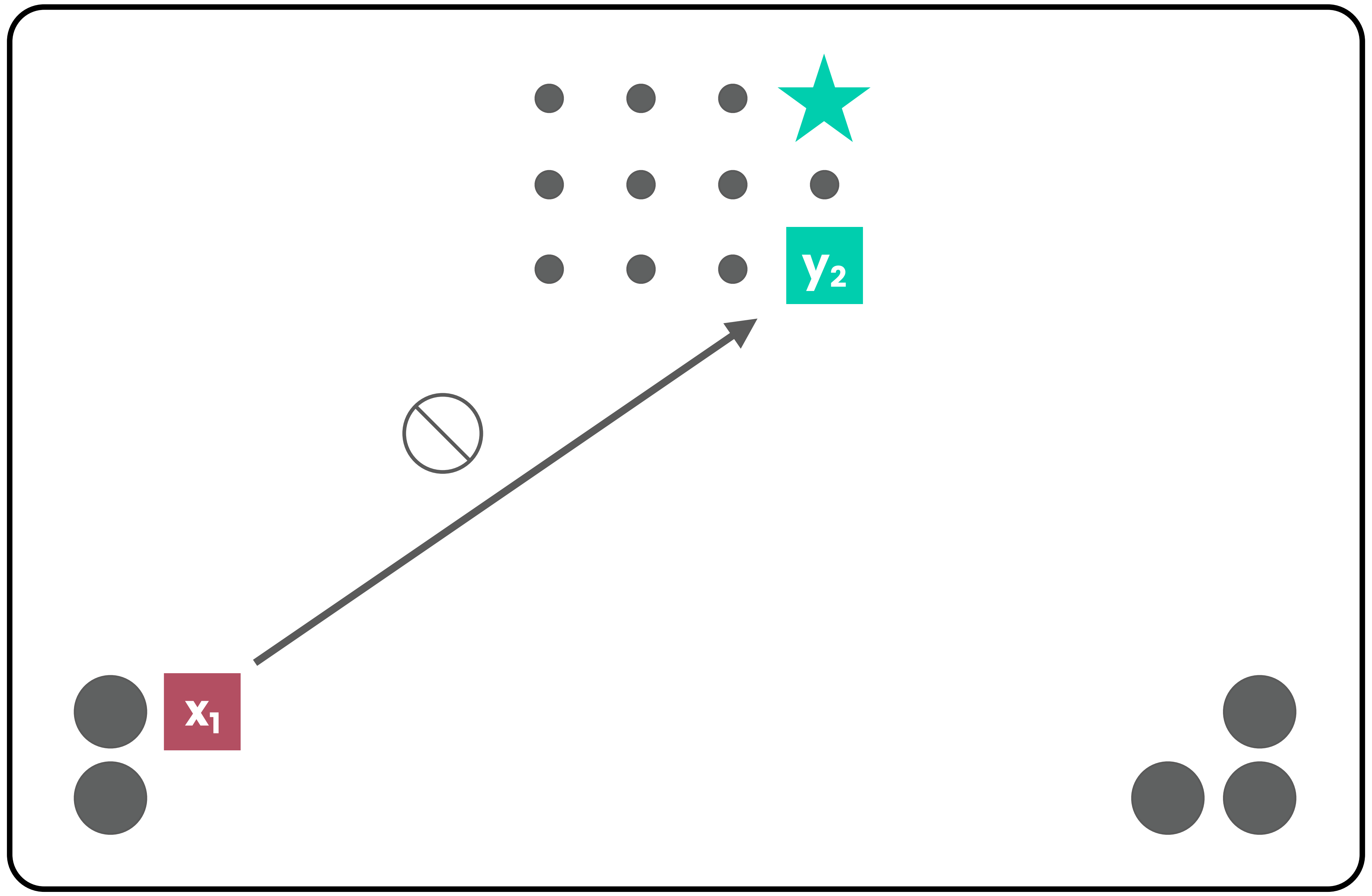
Markov chain construction

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



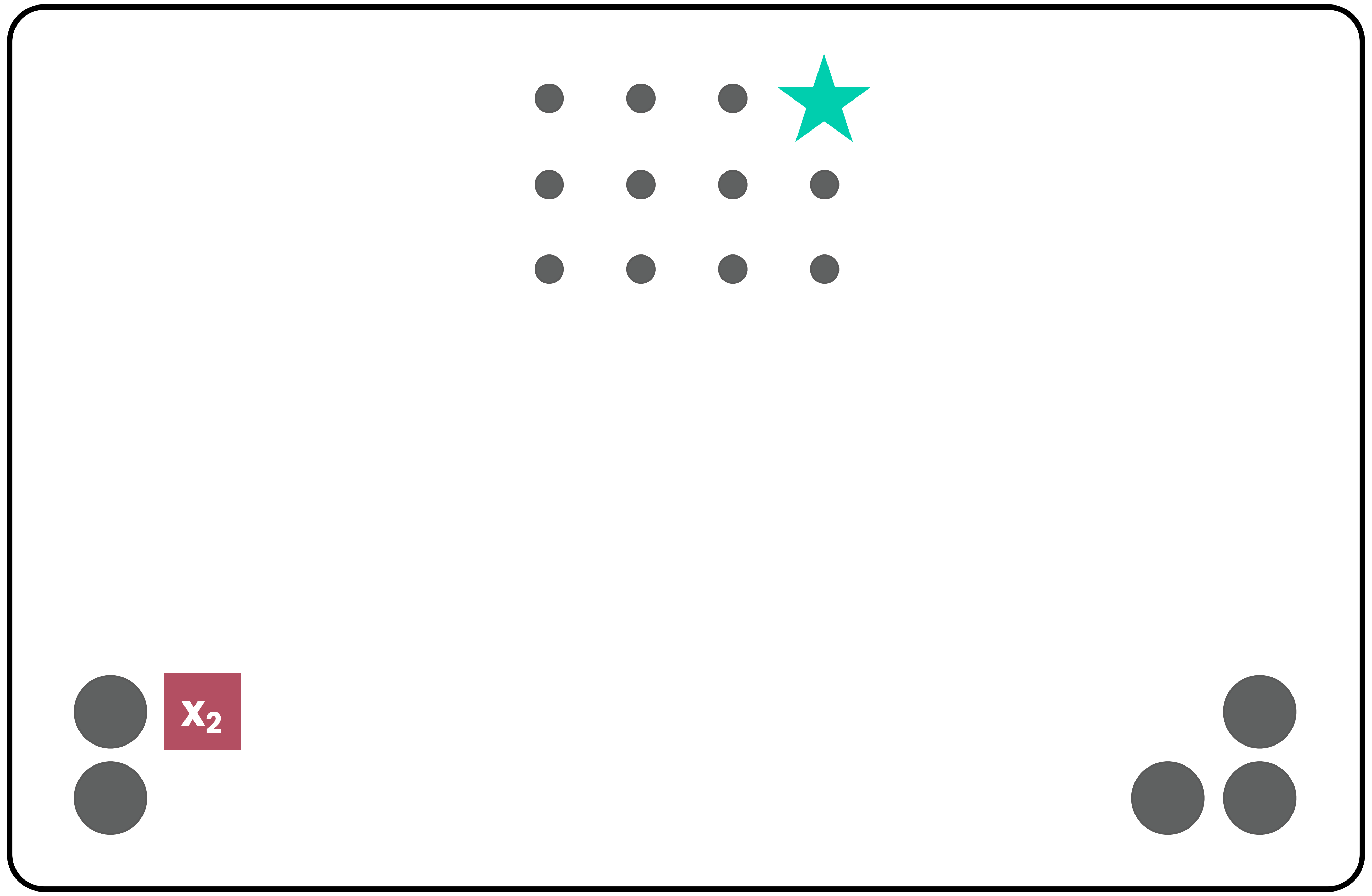
Markov chain construction

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



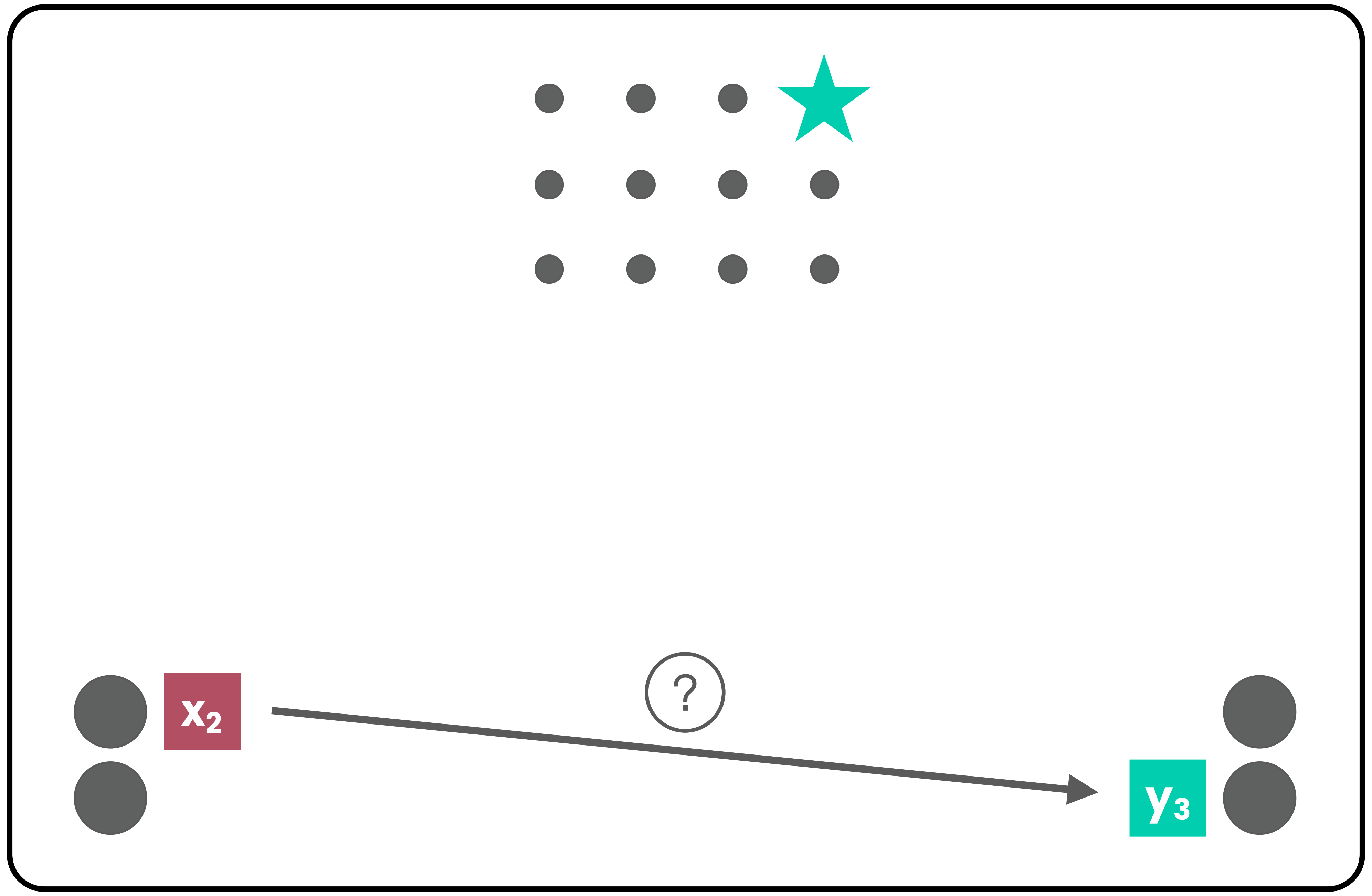
Markov chain construction

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



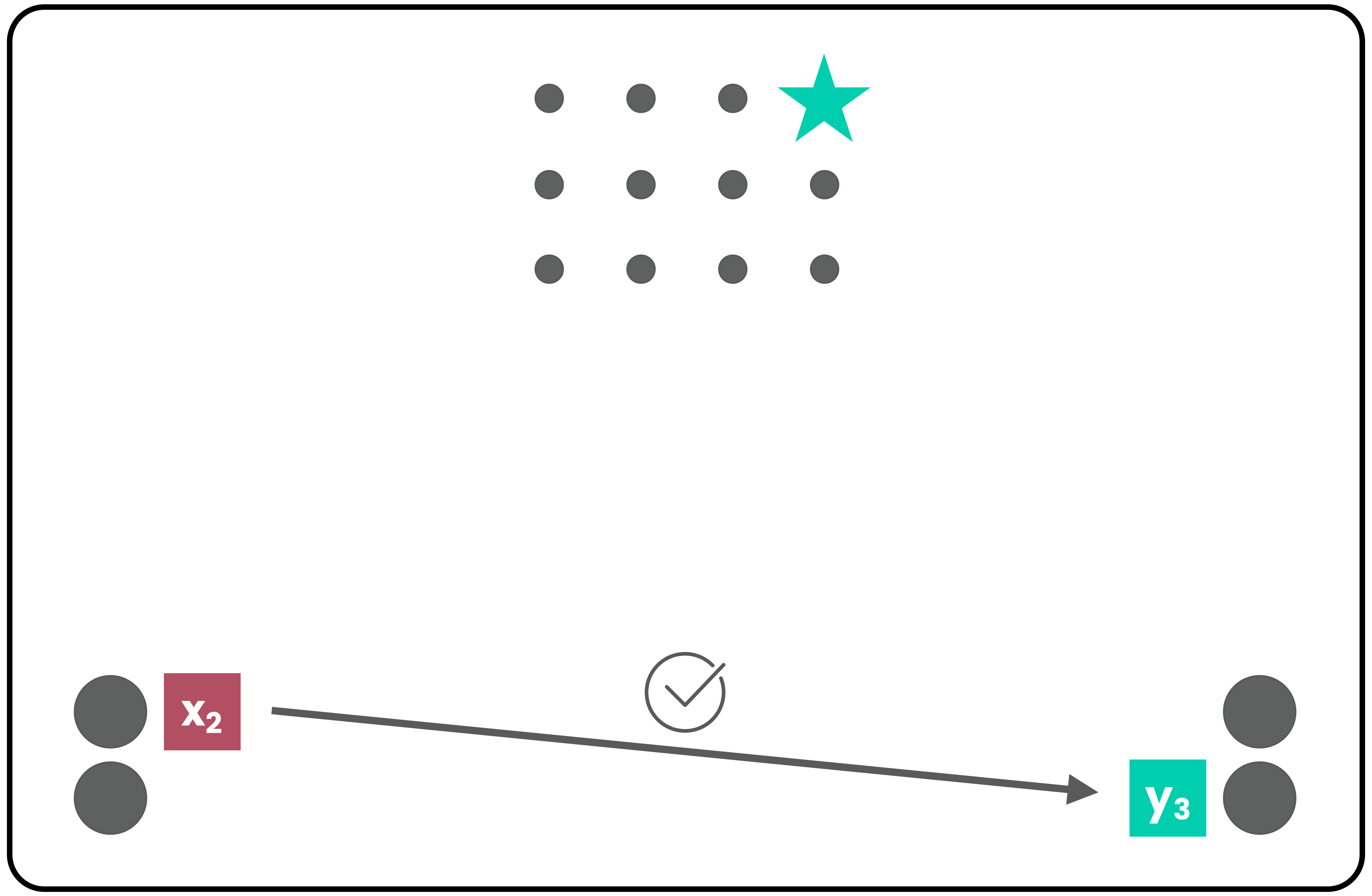
Markov chain construction

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



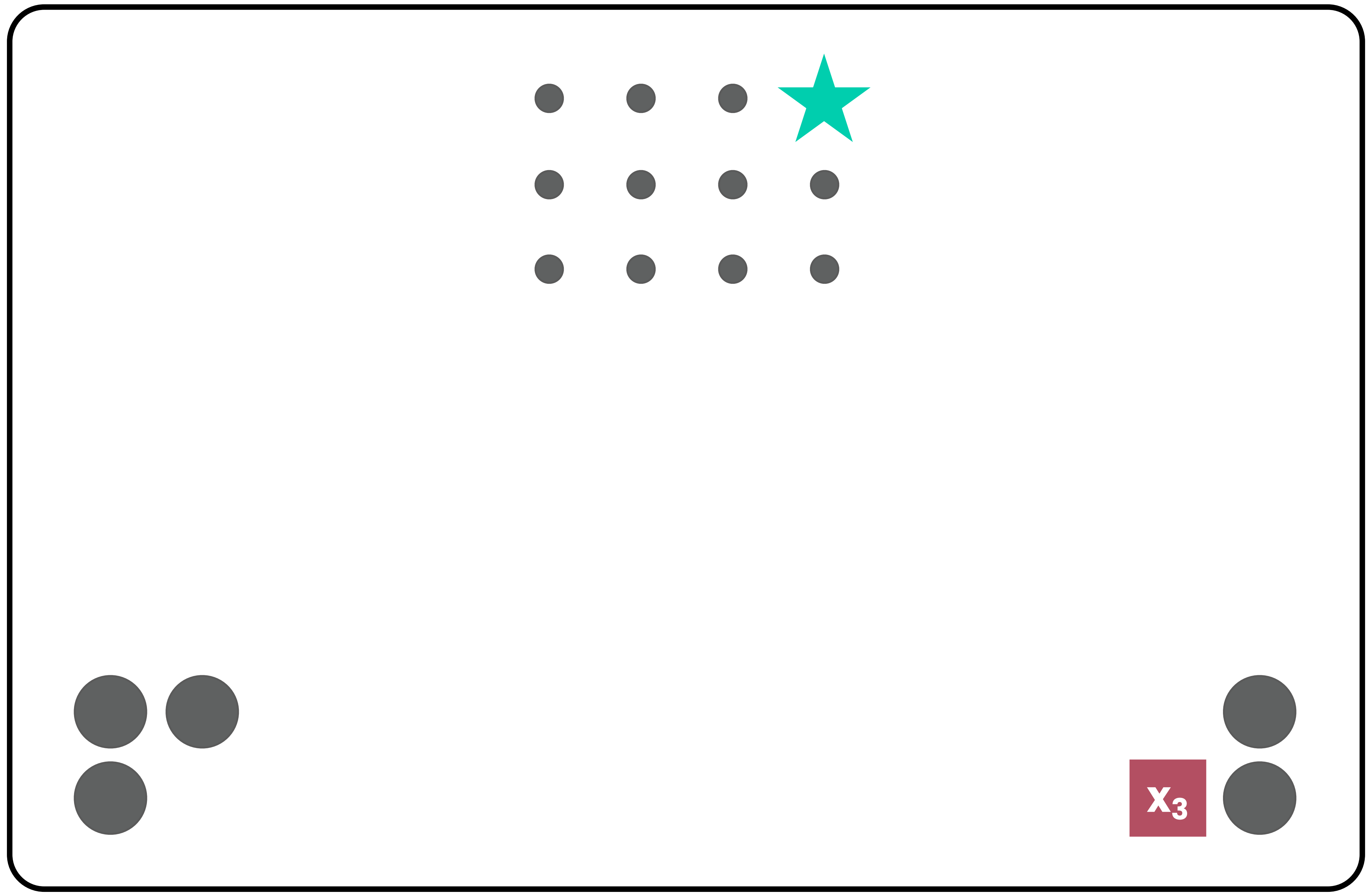
Markov chain construction

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



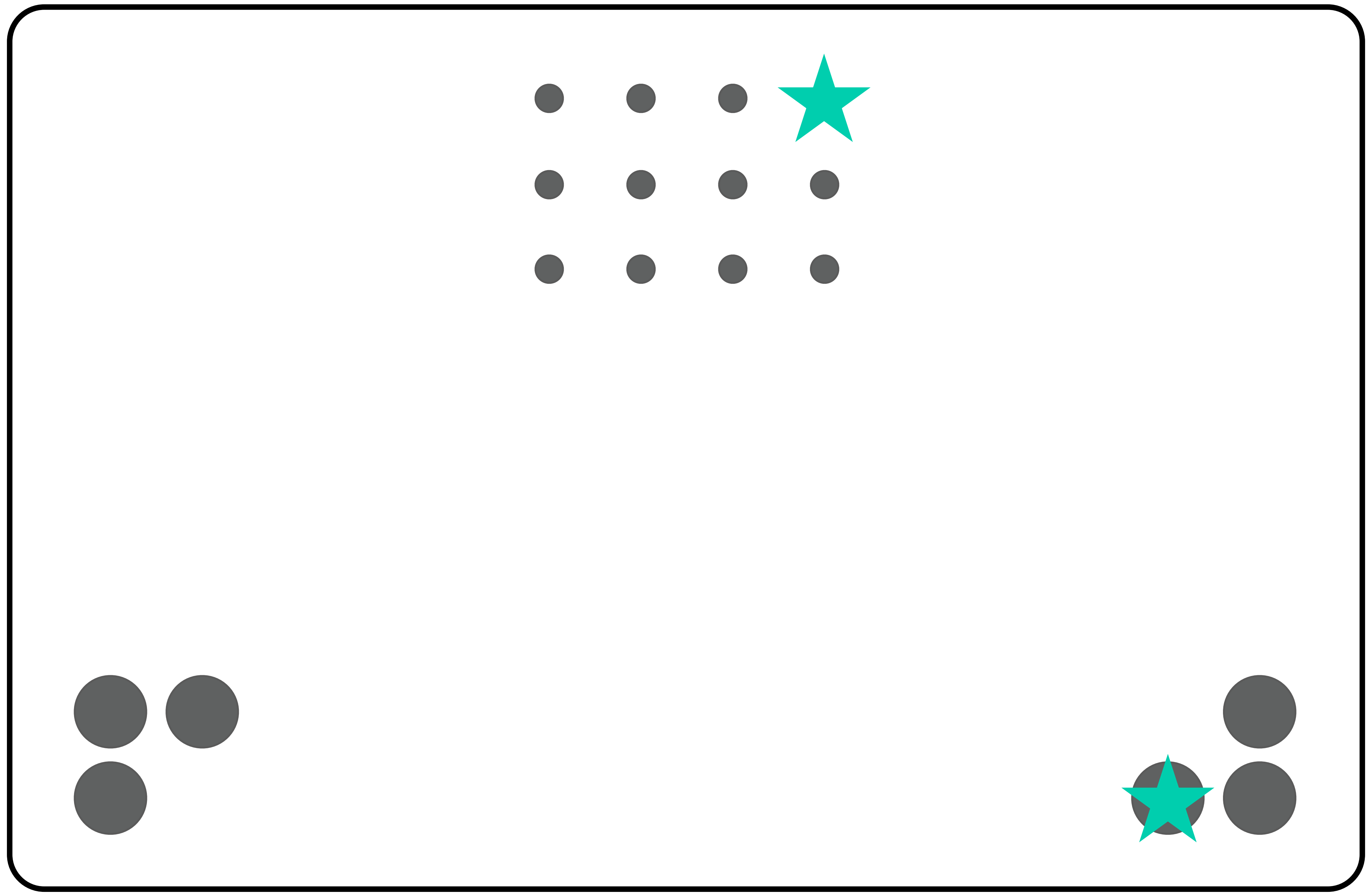
Markov chain construction

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



Markov chain construction

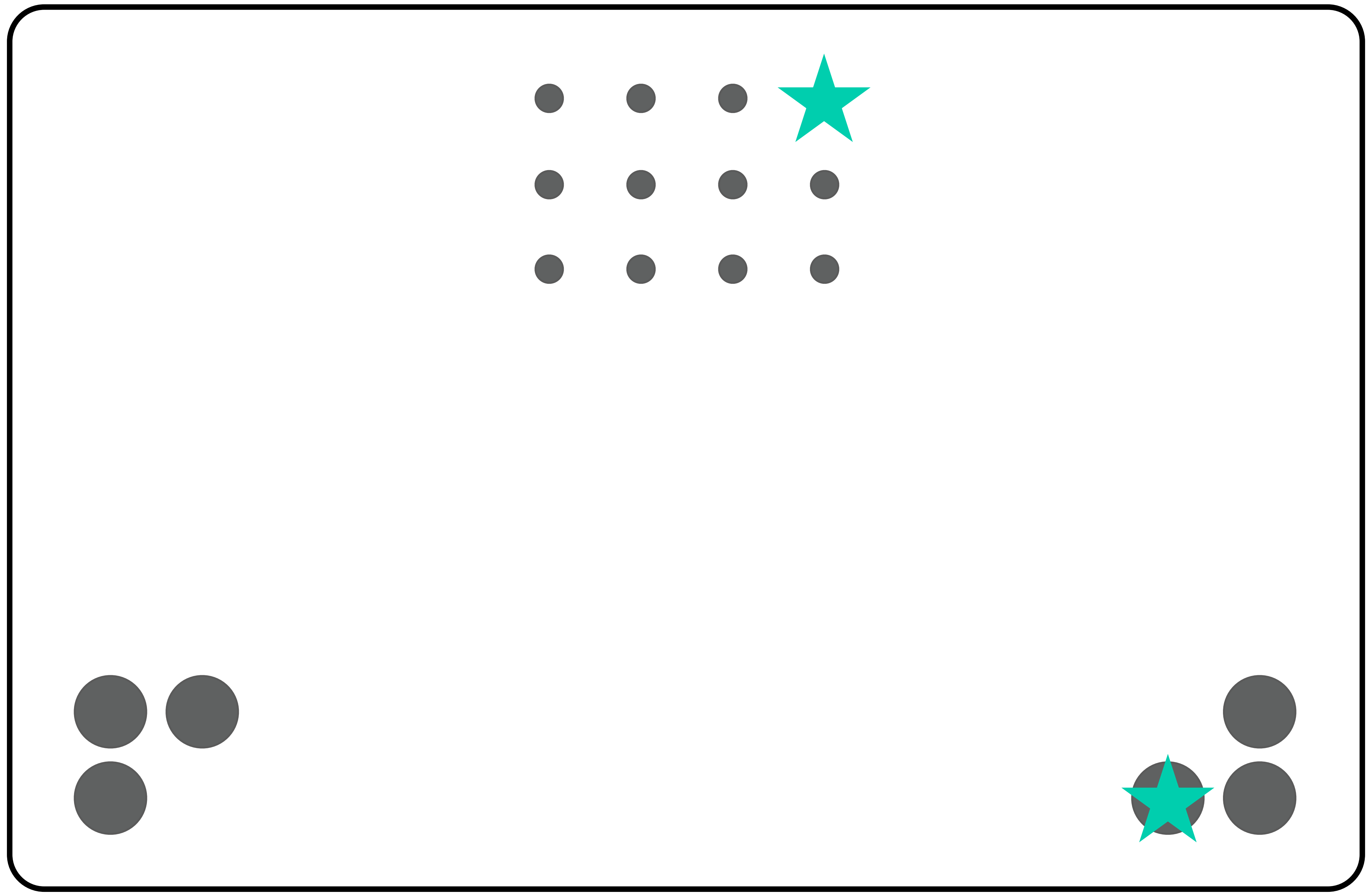
- Repeat **m** times to create Markov chain of length **m**
- Return x_m as cluster center



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



Algorithm

Algorithm

- Sample first center uniformly

Algorithm

- Sample first center uniformly
- Compute proposal distribution $q(x)$

Algorithm

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

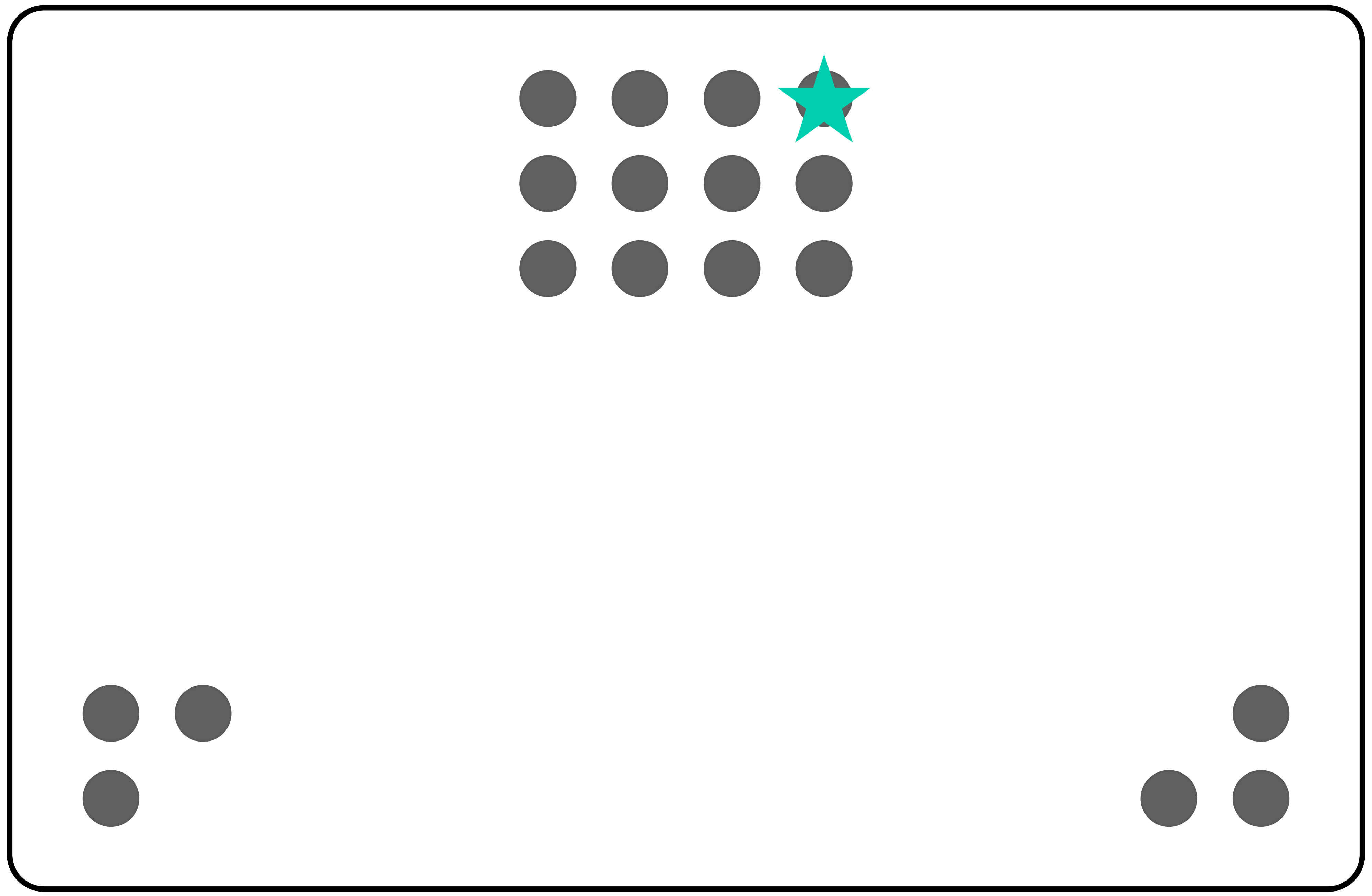
Algorithm

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

- Approximation of k-Means++ seeding

EFFICIENT IF **M** IS SMALL ENOUGH

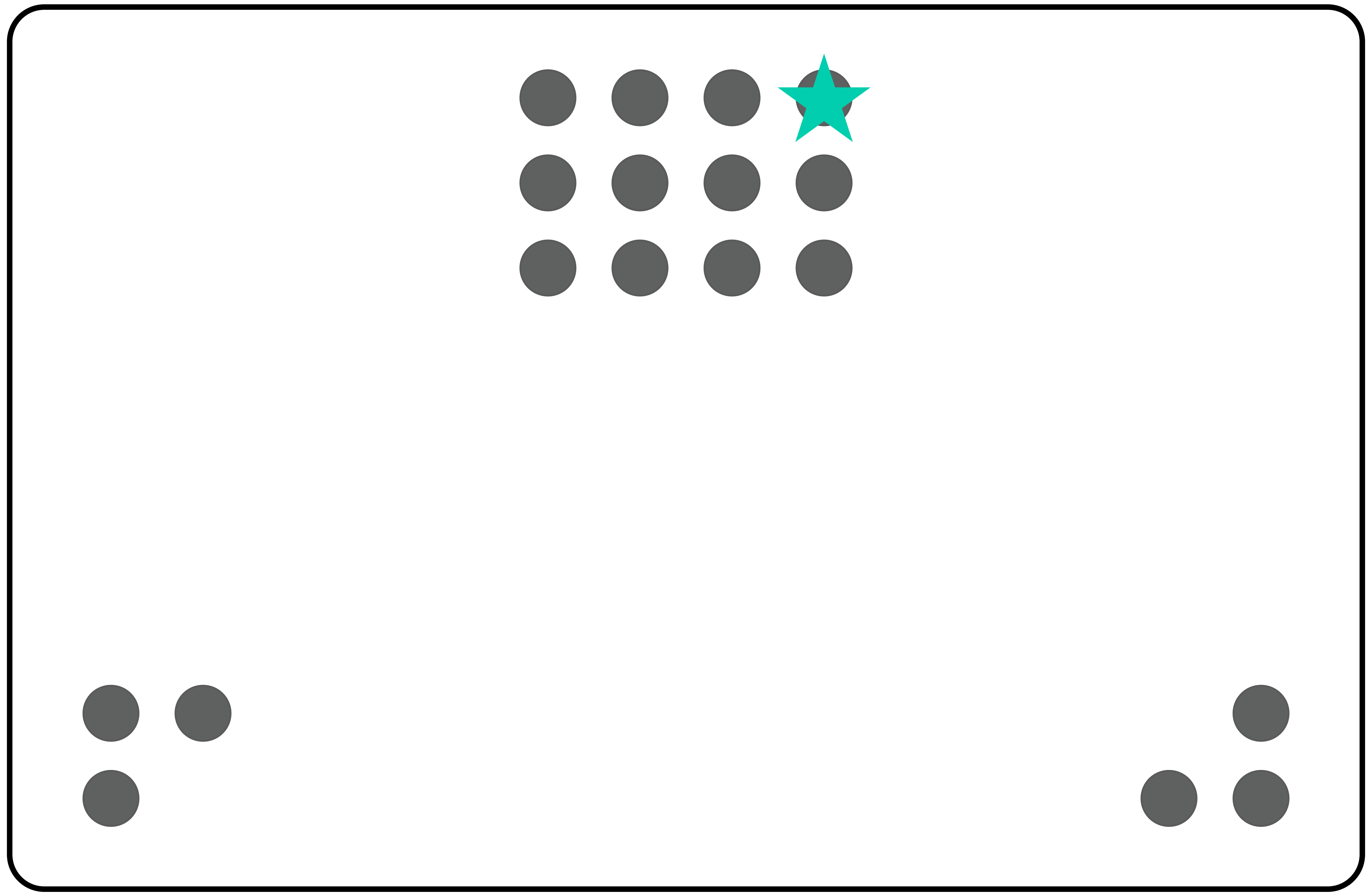
K-MC² [Bachem et al., 2016]



K-MC^2 [Bachem et al., 2016]

Uniform proposal:

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

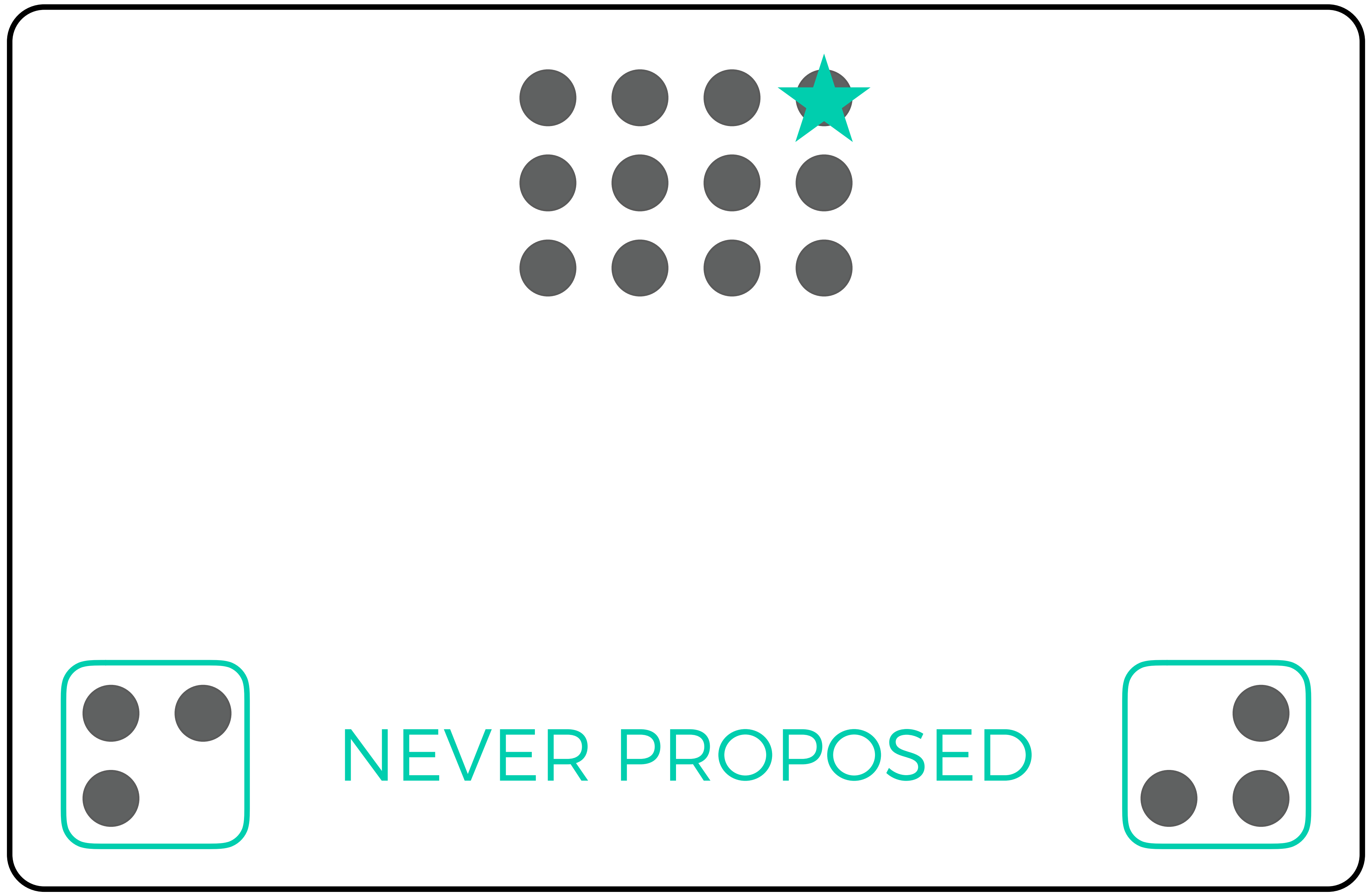


K-MC^2 [Bachem et al., 2016]

Uniform proposal:

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

⚠ Misses small, far away clusters



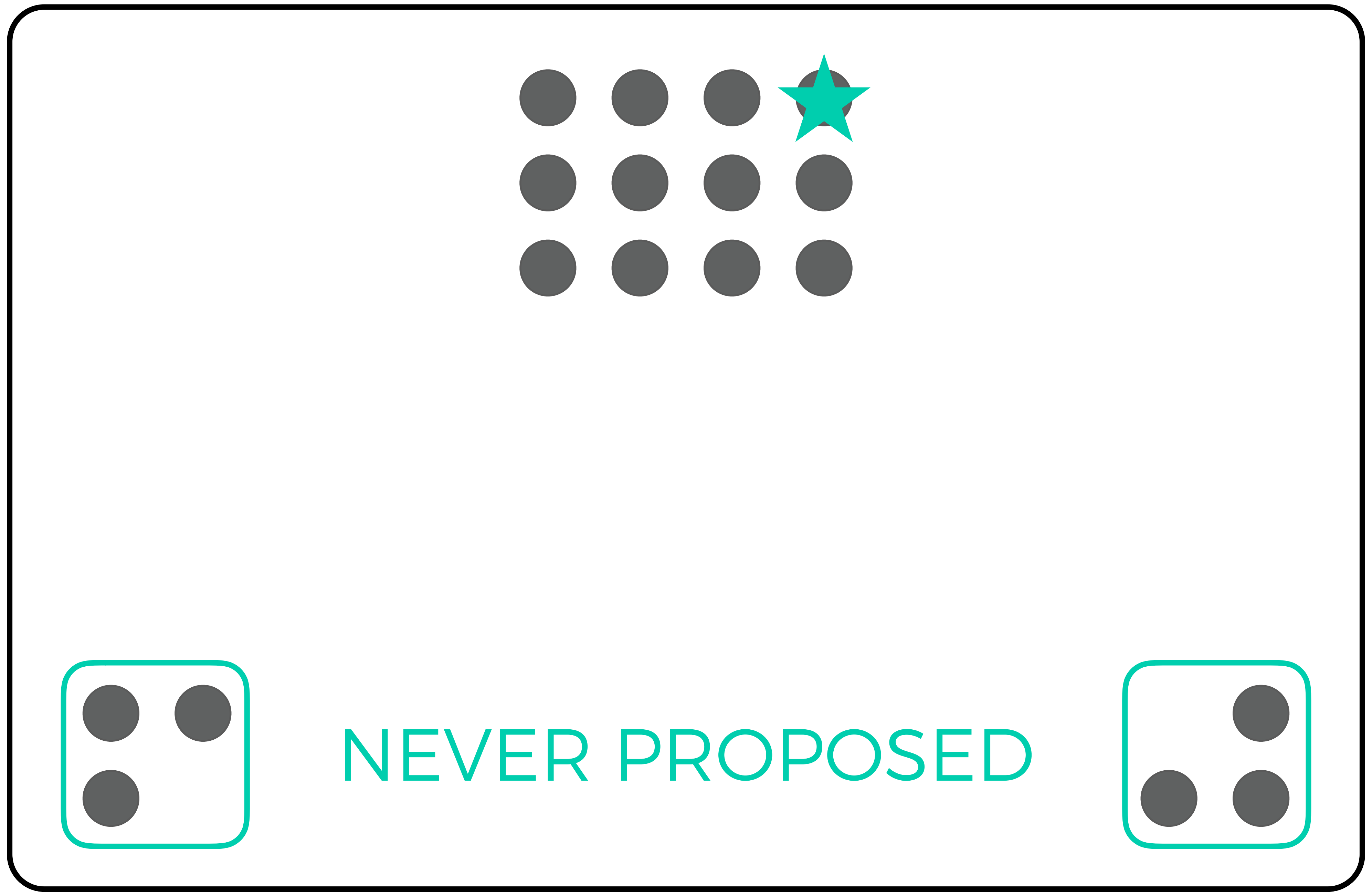
K-MC^2 [Bachem et al., 2016]

Uniform proposal:

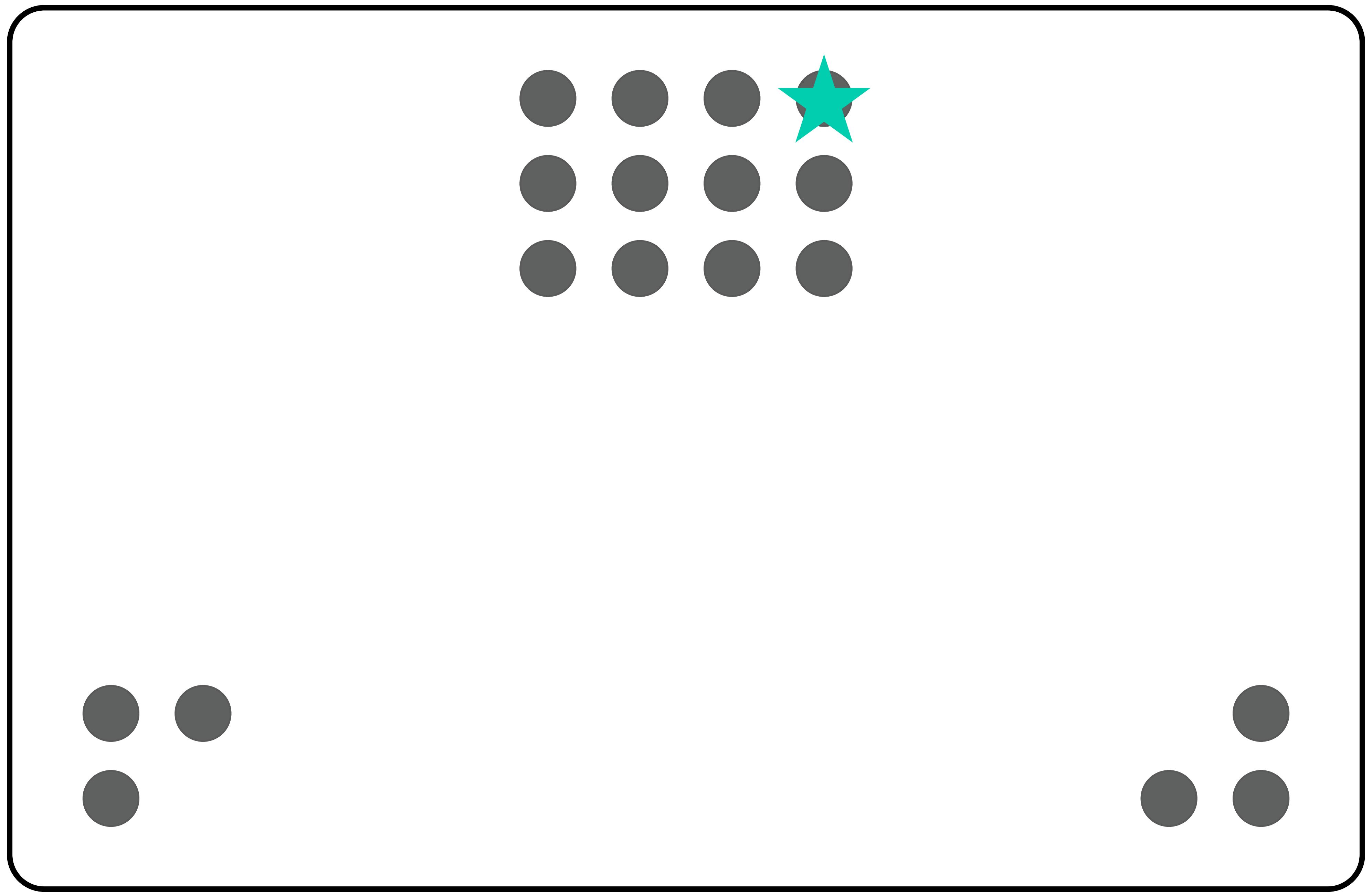
$$q(x) = \frac{1}{n}$$

Misses small, far away clusters

Requires assumptions on data or approach fails



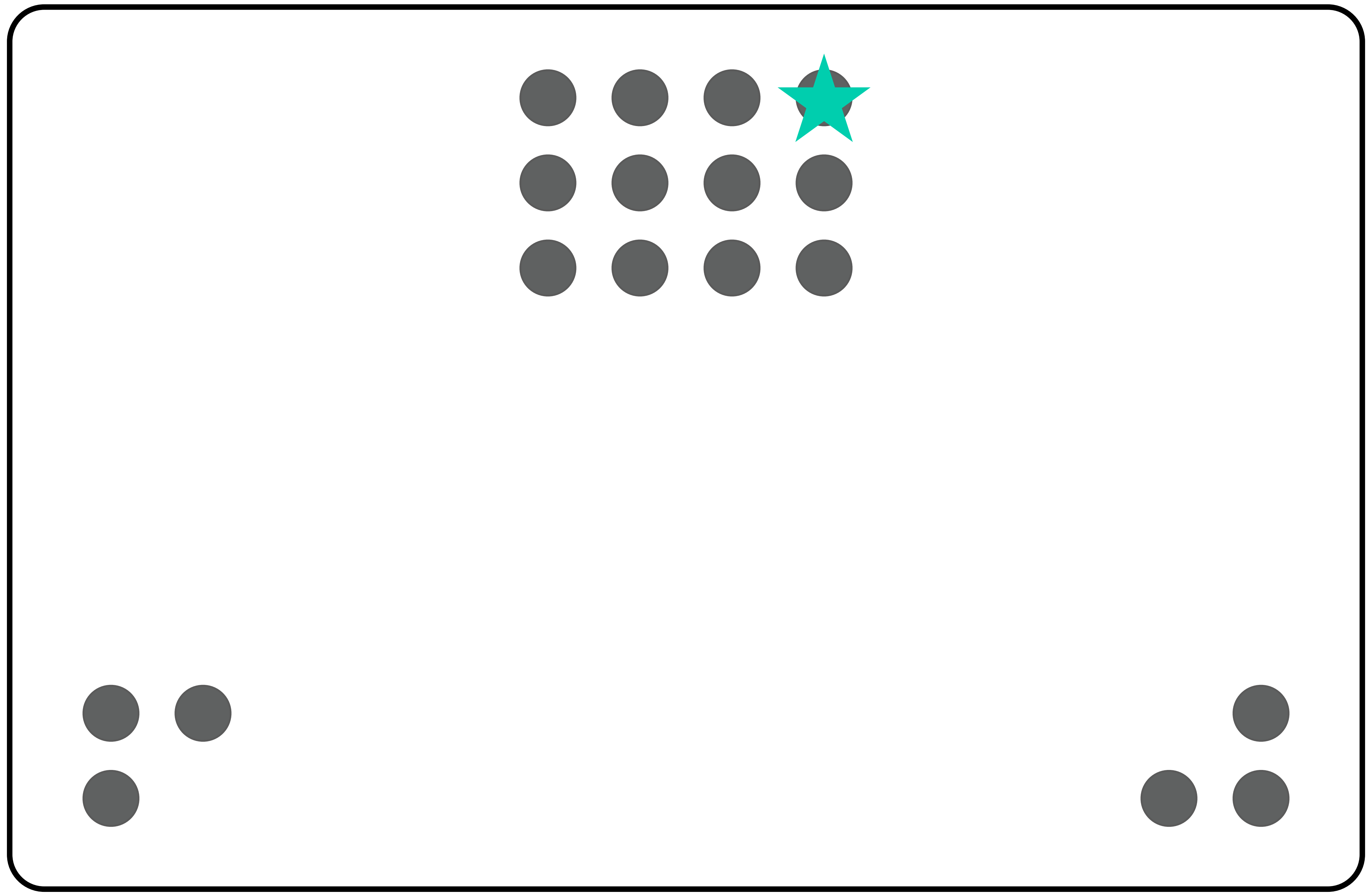
Assumption Free K-MC^2 [This paper]



Assumption Free K-MC^2 [This paper]

Nonuniform proposal:

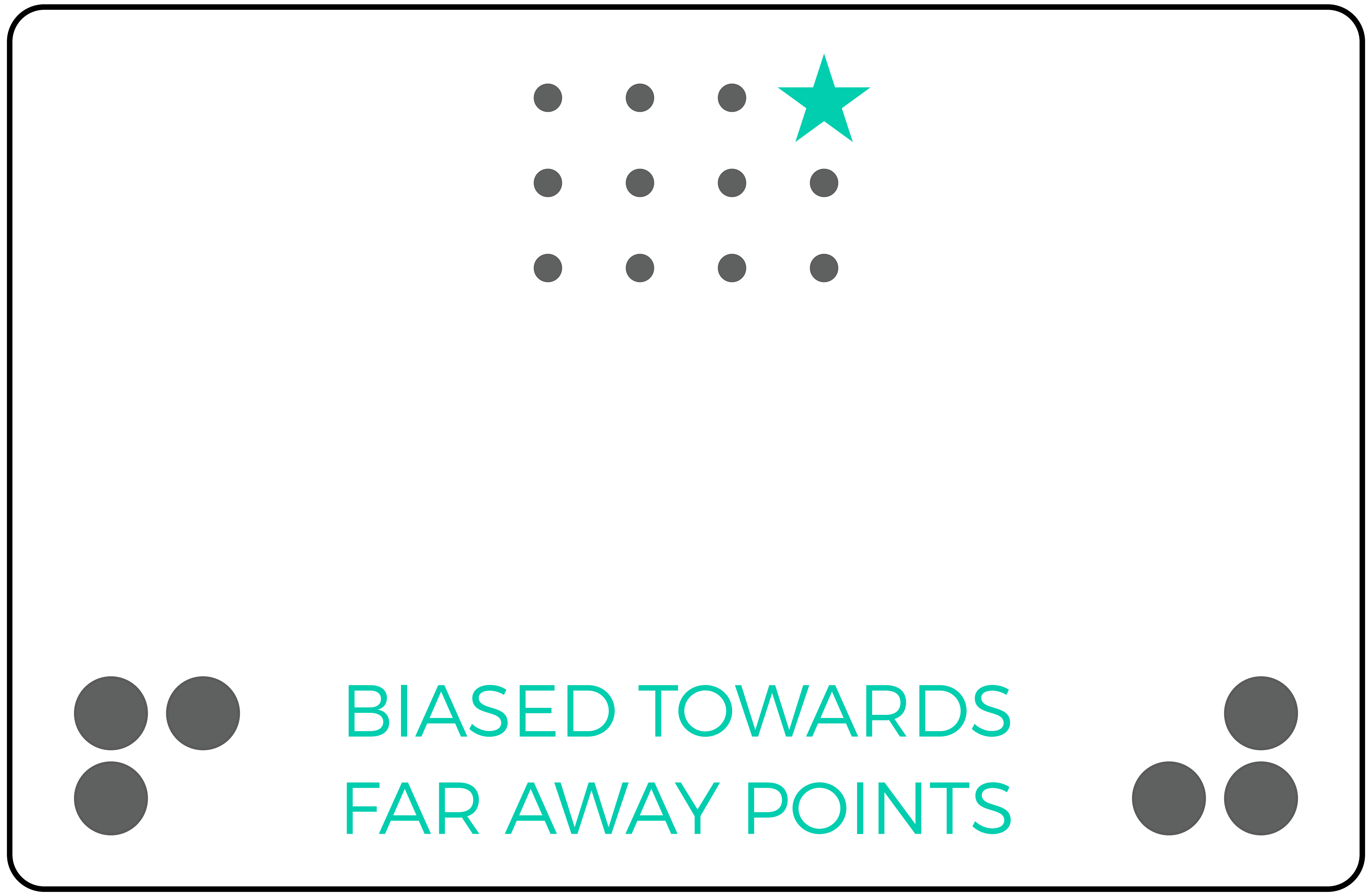
$$q(x) = \frac{1}{2n} + \frac{1}{2}\frac{\mathrm{d}(x, c_1)^2}{\sum_{x' \in \mathcal{X}} \mathrm{d}(x', c_1)^2}$$



Assumption Free K-MC² [This paper]

Nonuniform proposal:

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

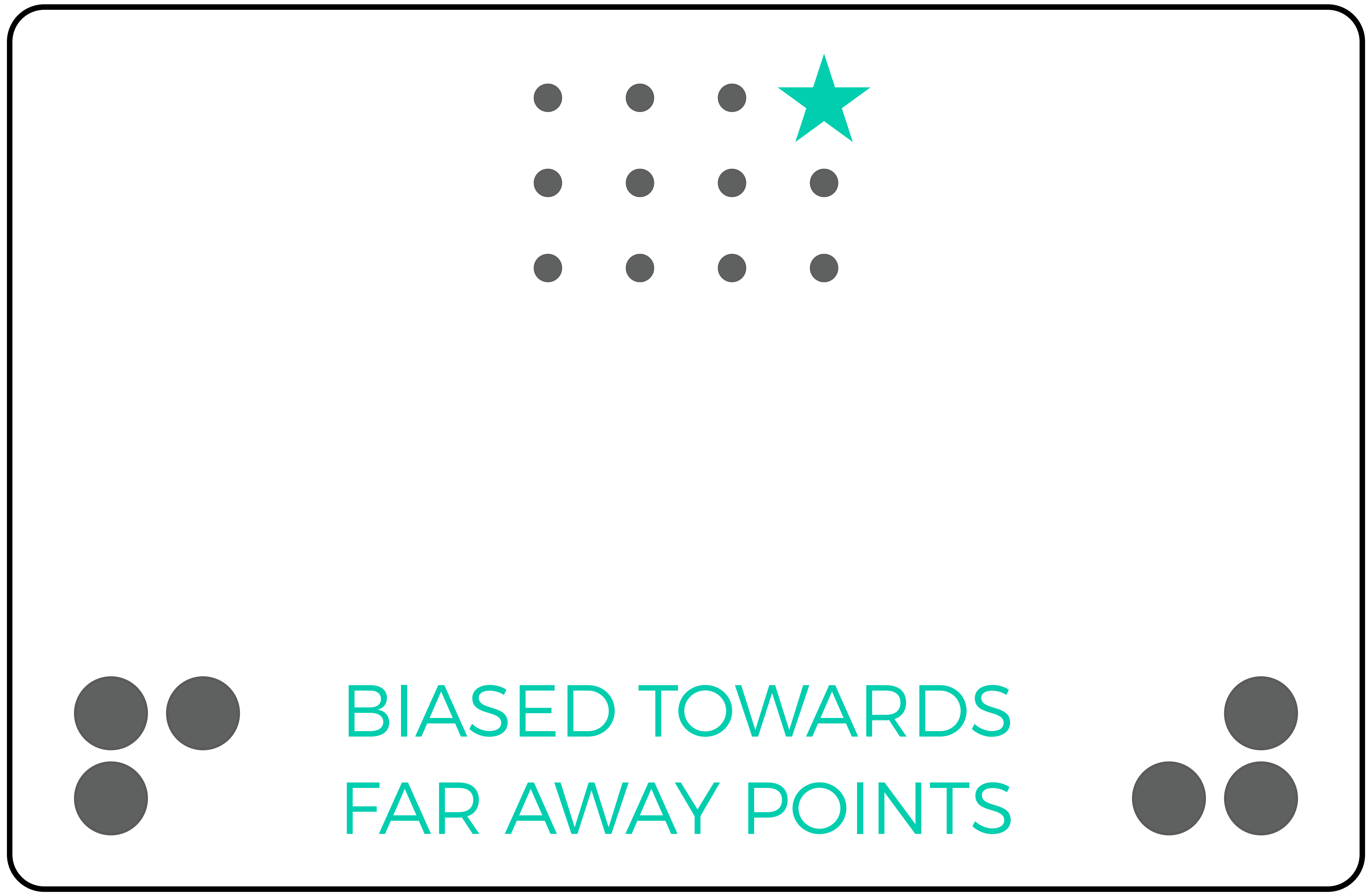


Assumption Free K-MC^2 [This paper]

Nonuniform proposal:

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

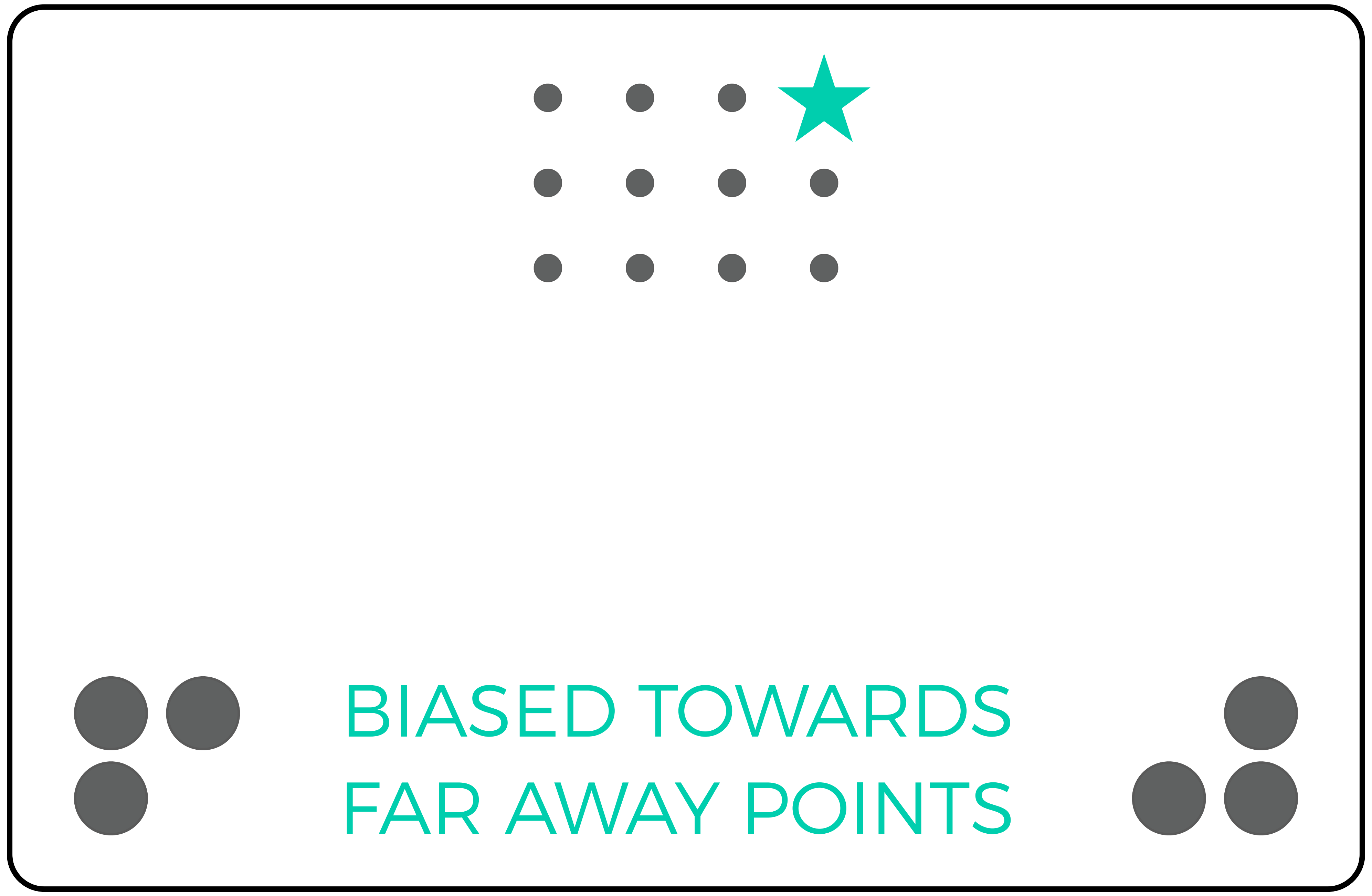
Provably good w/o assumptions



Assumption Free K-MC^2 [This paper]

Nonuniform proposal:

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



Provably good w/o assumptions

Works really well empirically

Assumption Free K-MC^2 [This paper]

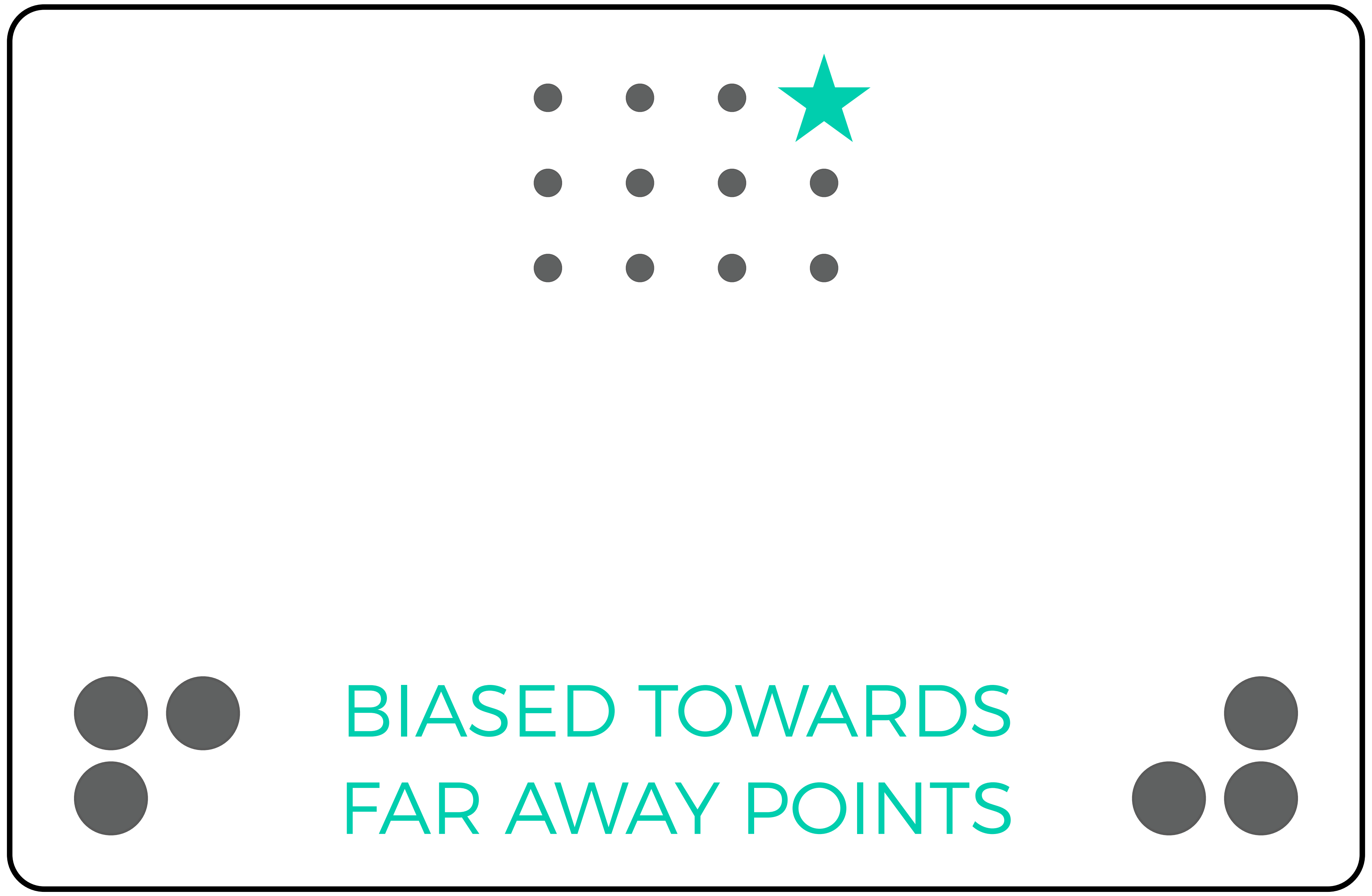
Nonuniform proposal:

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

COMPUTED ONCE IN SINGLE LINEAR PASS

Provably good w/o assumptions

Works really well empirically



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 $\epsilon > 0$
- Run algorithm with $m = 1 + \frac{8}{\epsilon} \log \frac{4k}{\epsilon}$

INDEPENDENT OF DATA SET SIZE

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:

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

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:

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

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:

APPROXIMATION

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

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:

APPROXIMATION

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

SAME AS K-MEANS++

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

Main theoretical result

Choose an error tolerance $\epsilon > 0$

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

Expected solution quality:

APPROXIMATION

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

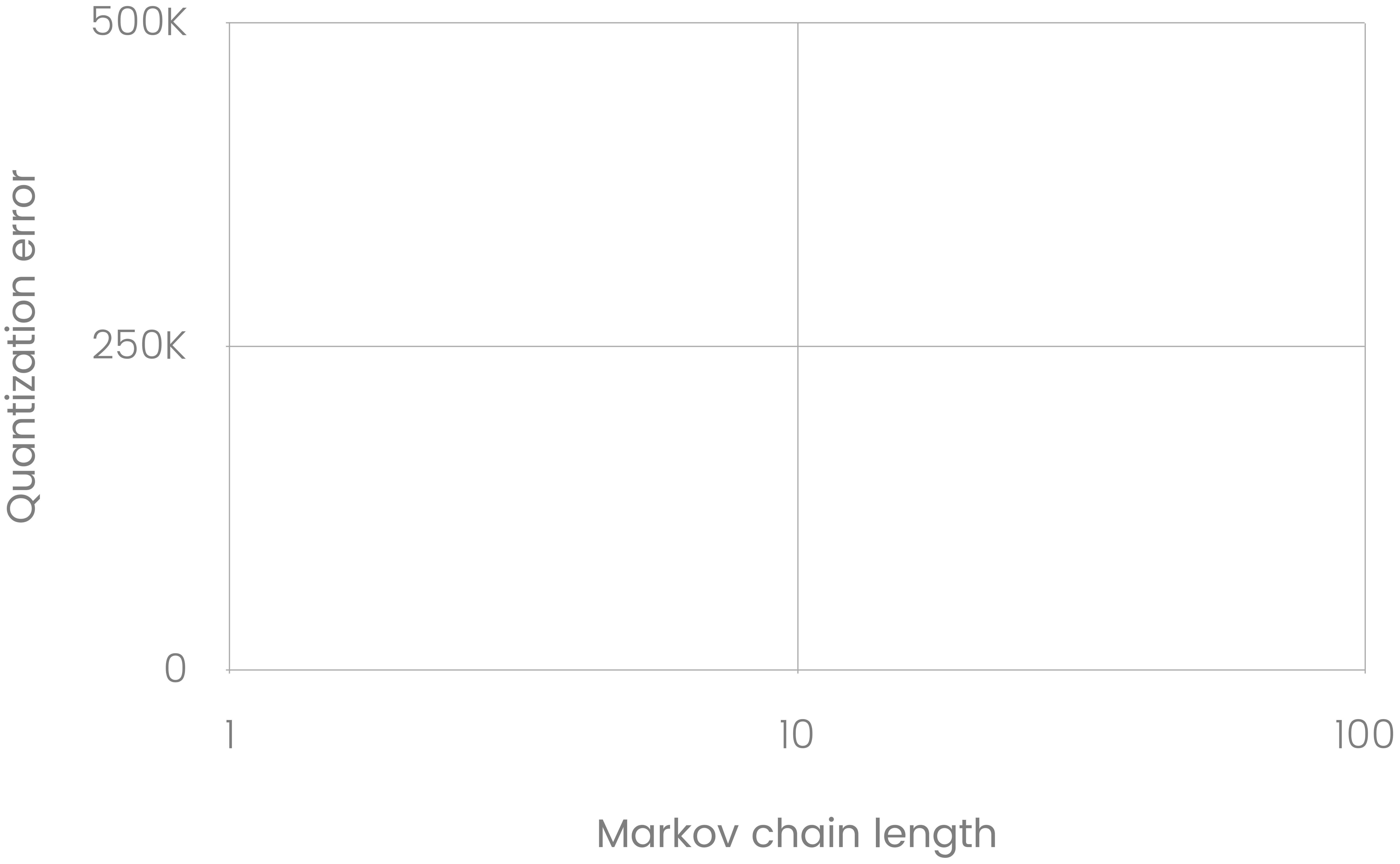
SAME AS K-MEANS++

Total runtime: $\mathcal{O}\left(nd + \frac{1}{\epsilon}k^2 d \log \frac{k}{\epsilon}\right)$ FASTER THAN $\mathcal{O}(nkd)$

Experimental results

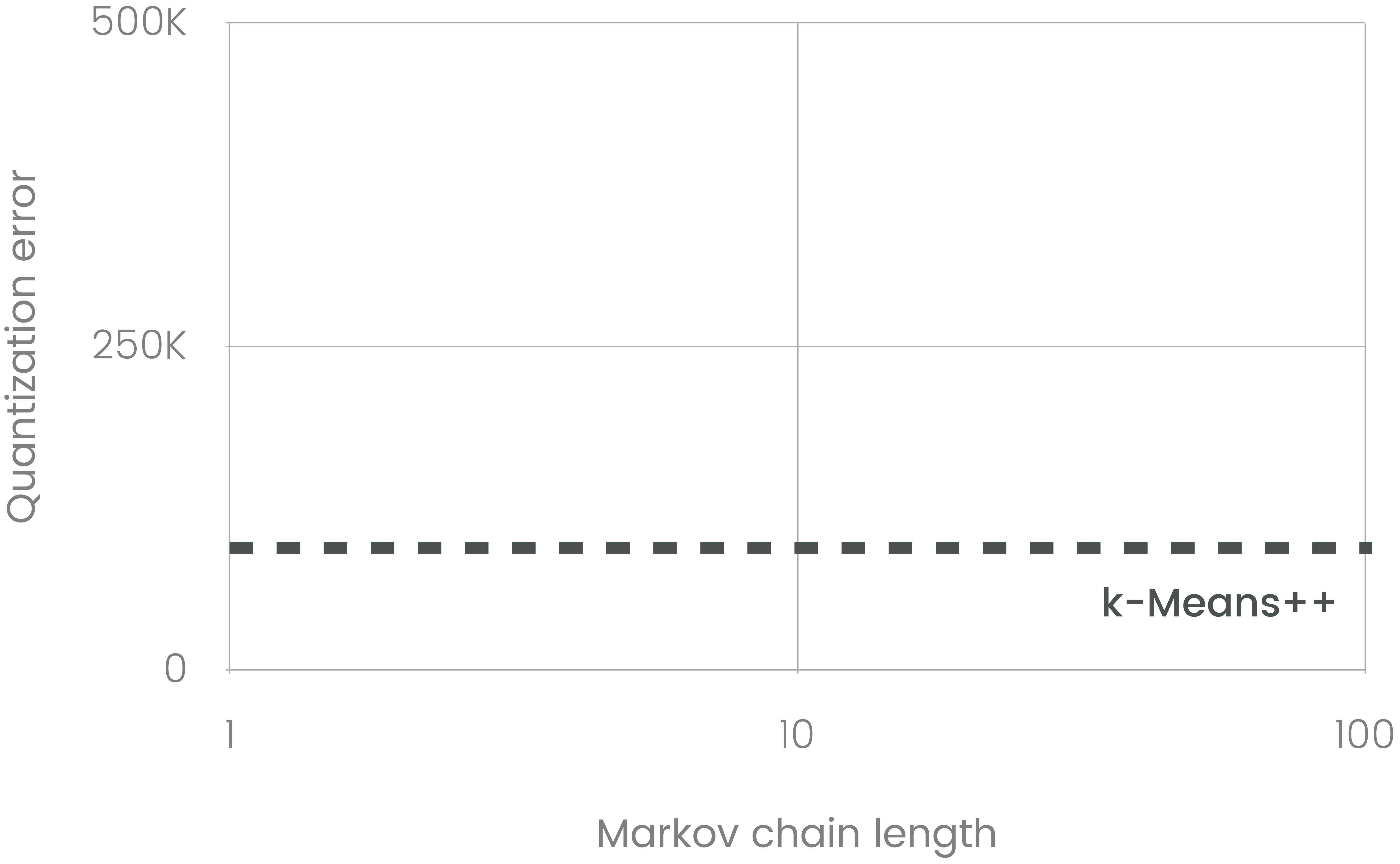
Experimental results

CSN



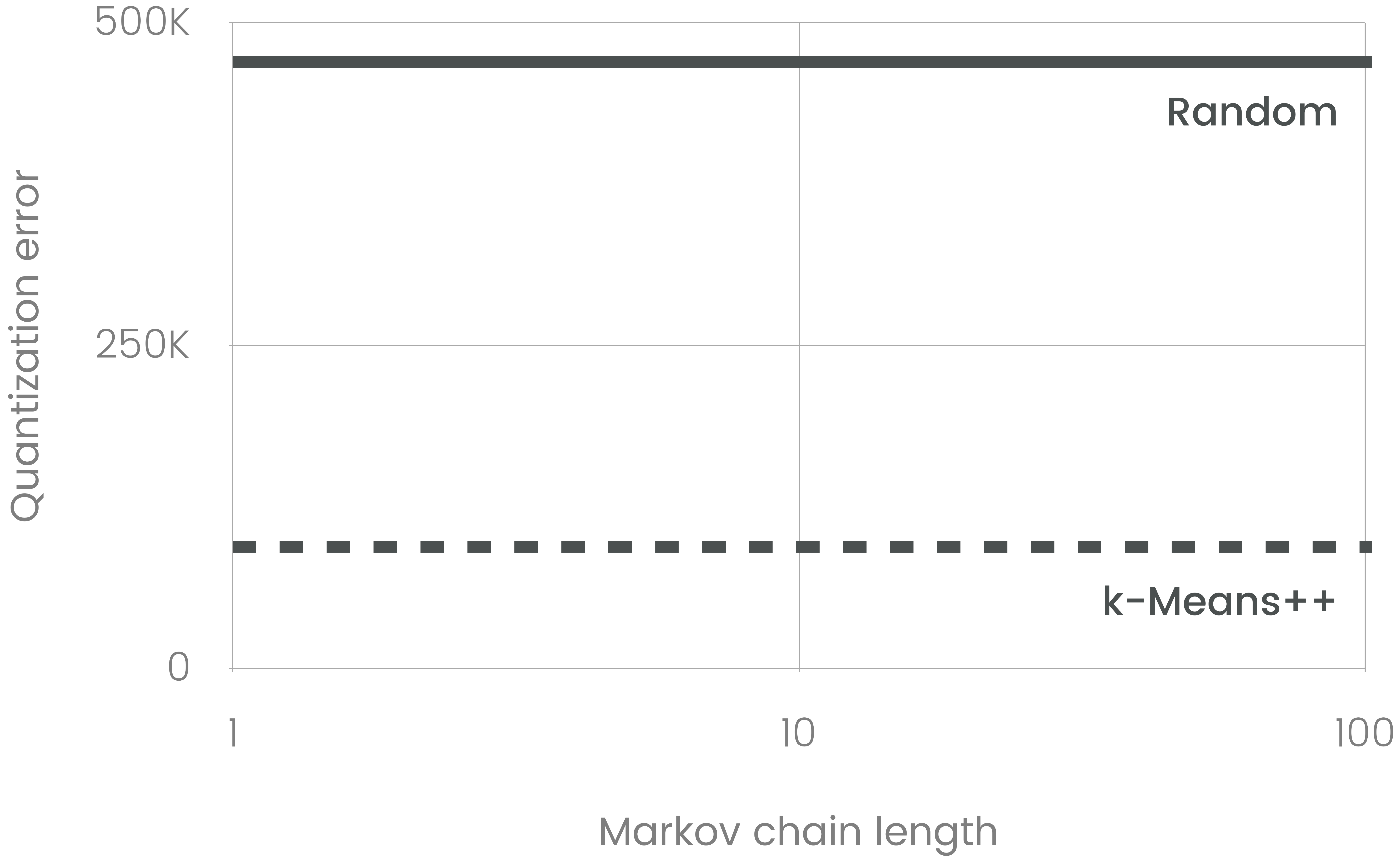
Experimental results

CSN



Experimental results

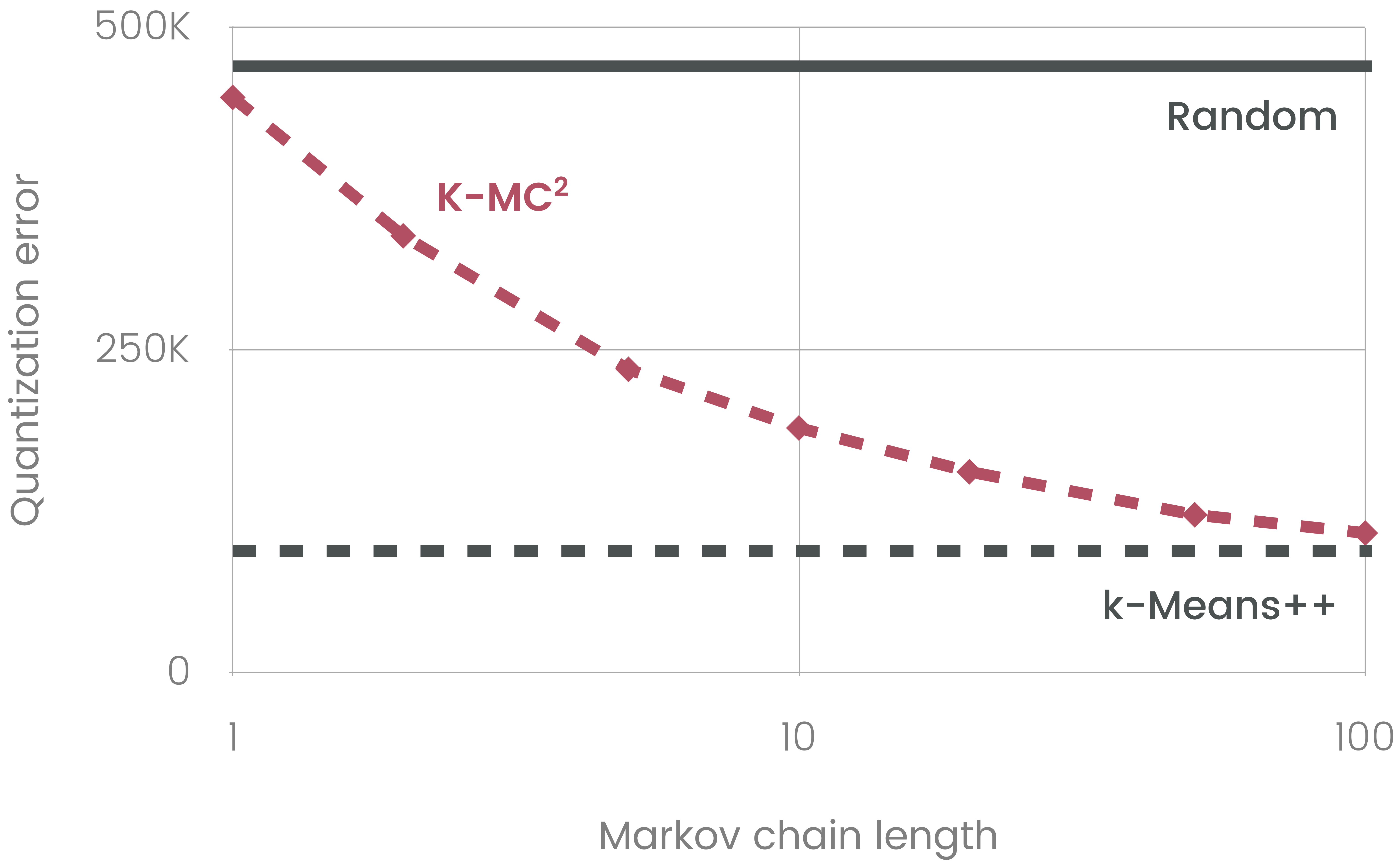
CSN



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

Experimental results

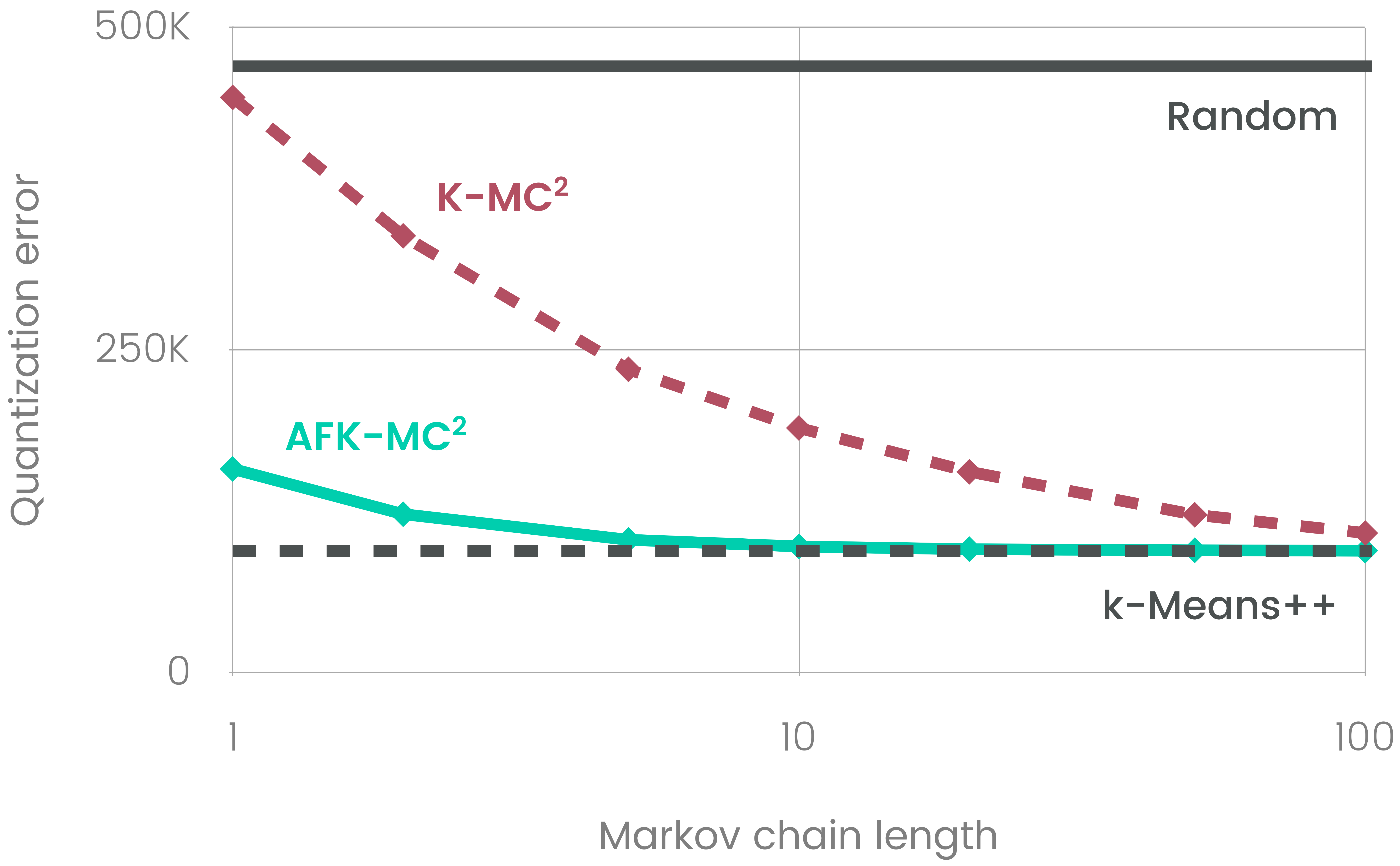
CSN



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

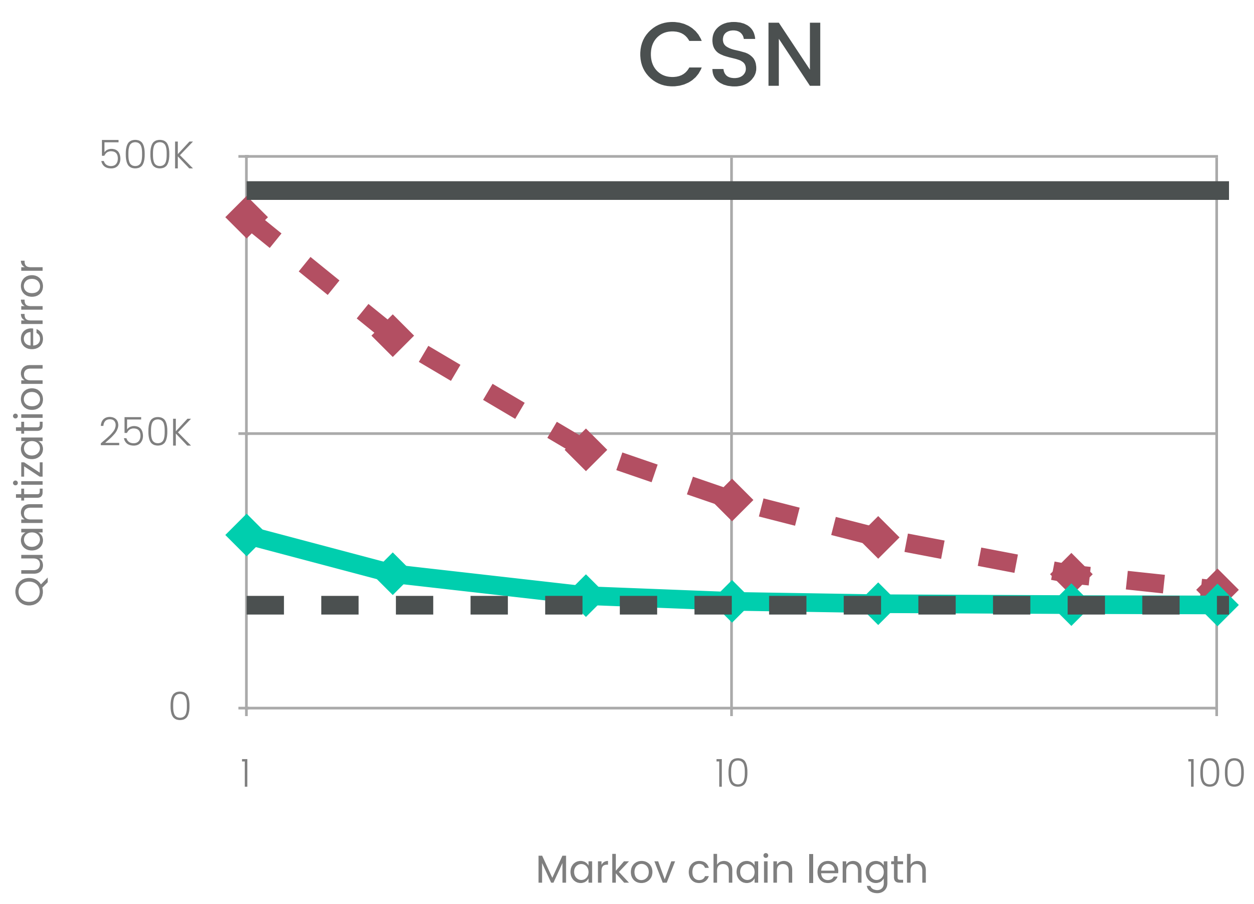
Experimental results

CSN

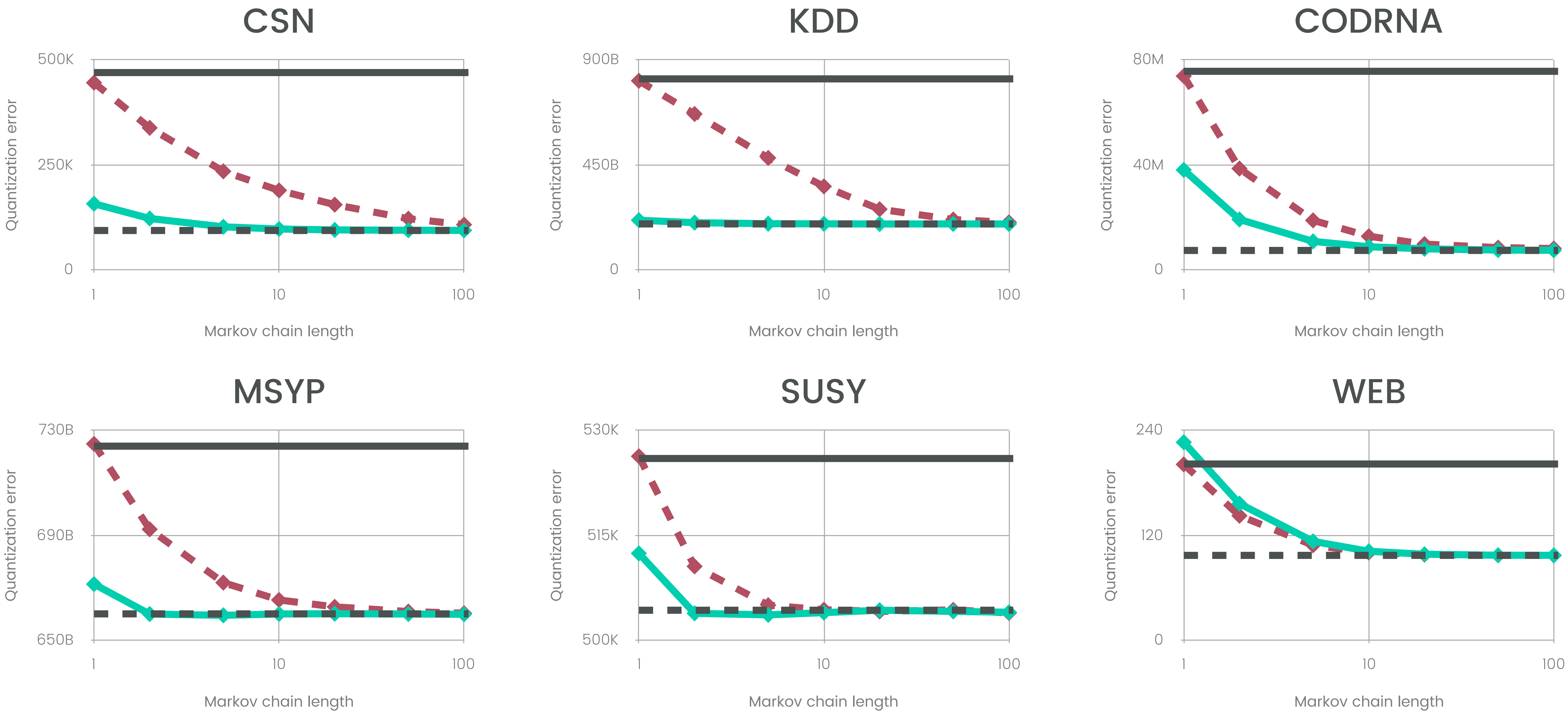


Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause

Experimental results

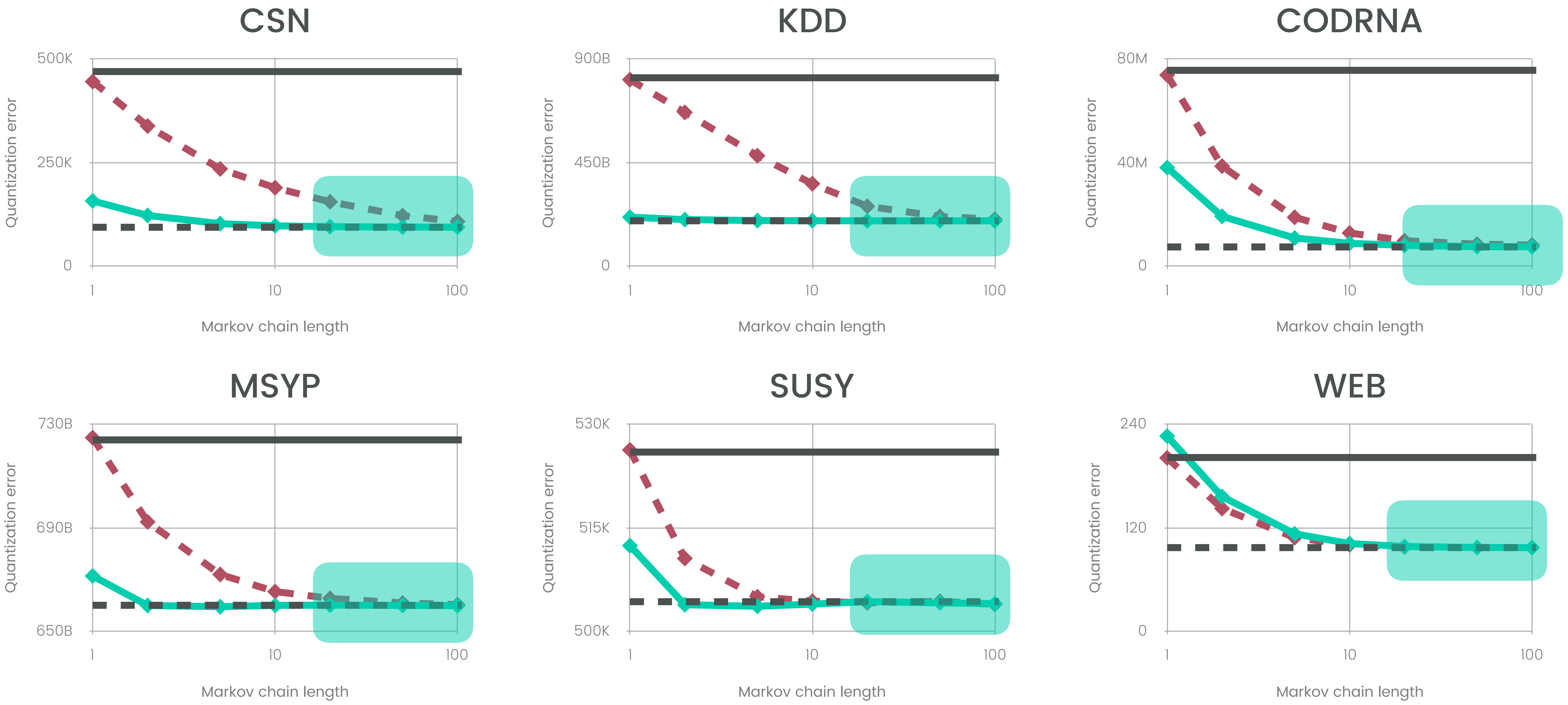


Experimental results



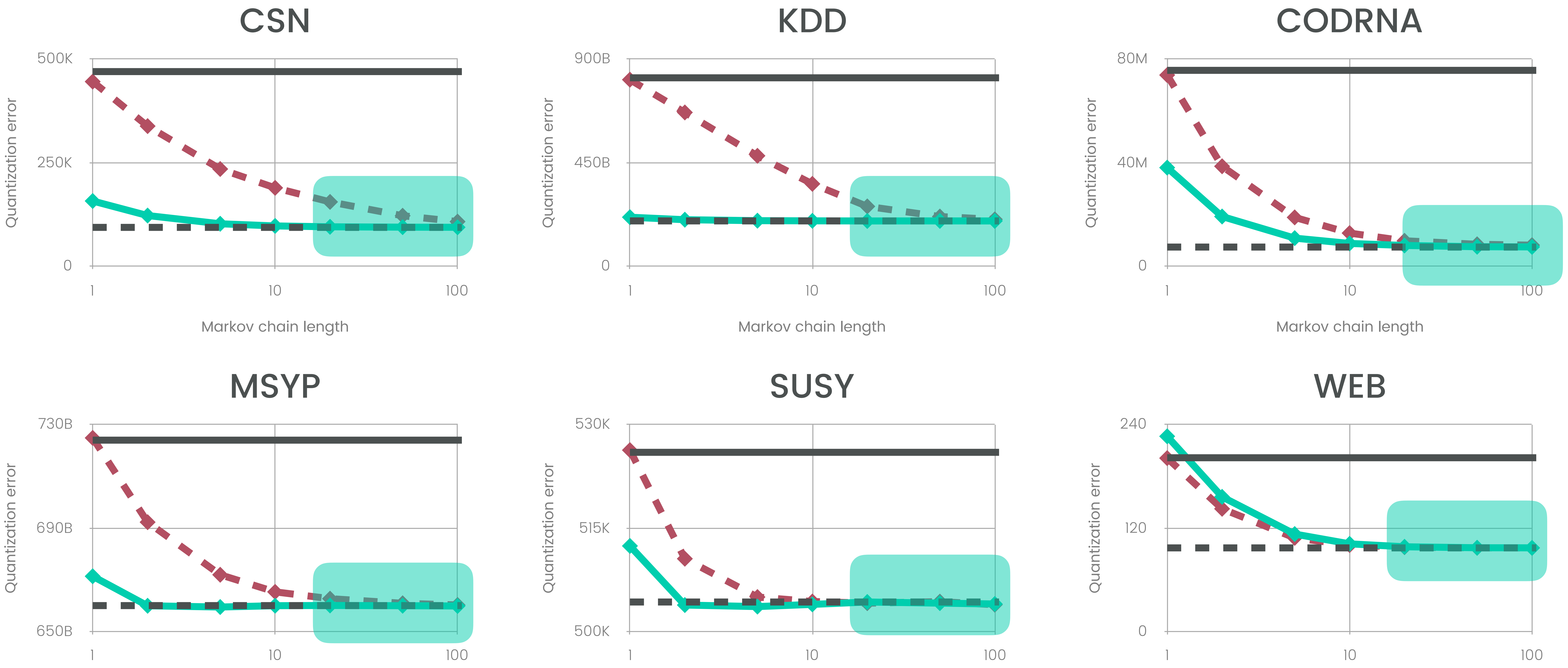
Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause

Experimental results



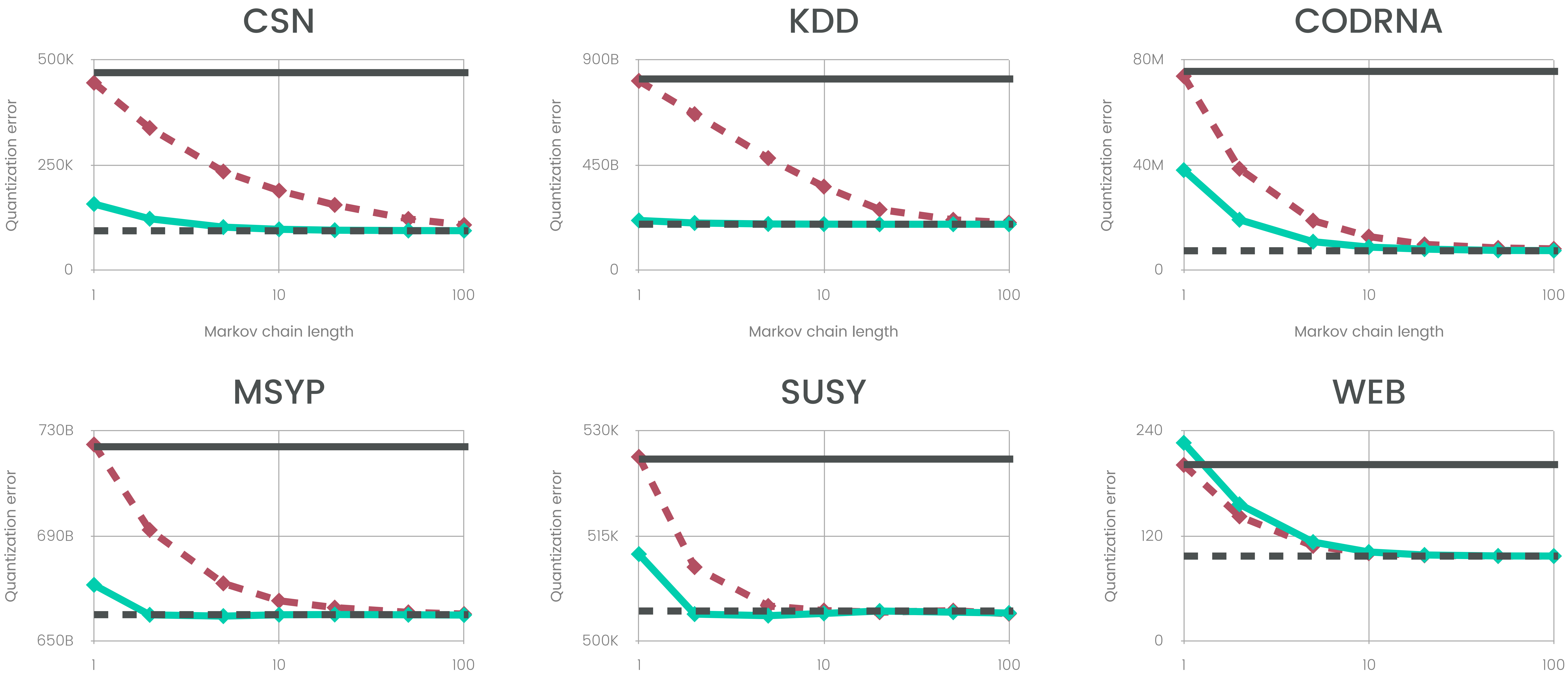
Olivier Bachem, Mario Lucic, S. Hamed Hassani, Andreas Krause

Experimental results



M=100 IS SUFFICIENT IN PRACTICE

Experimental results



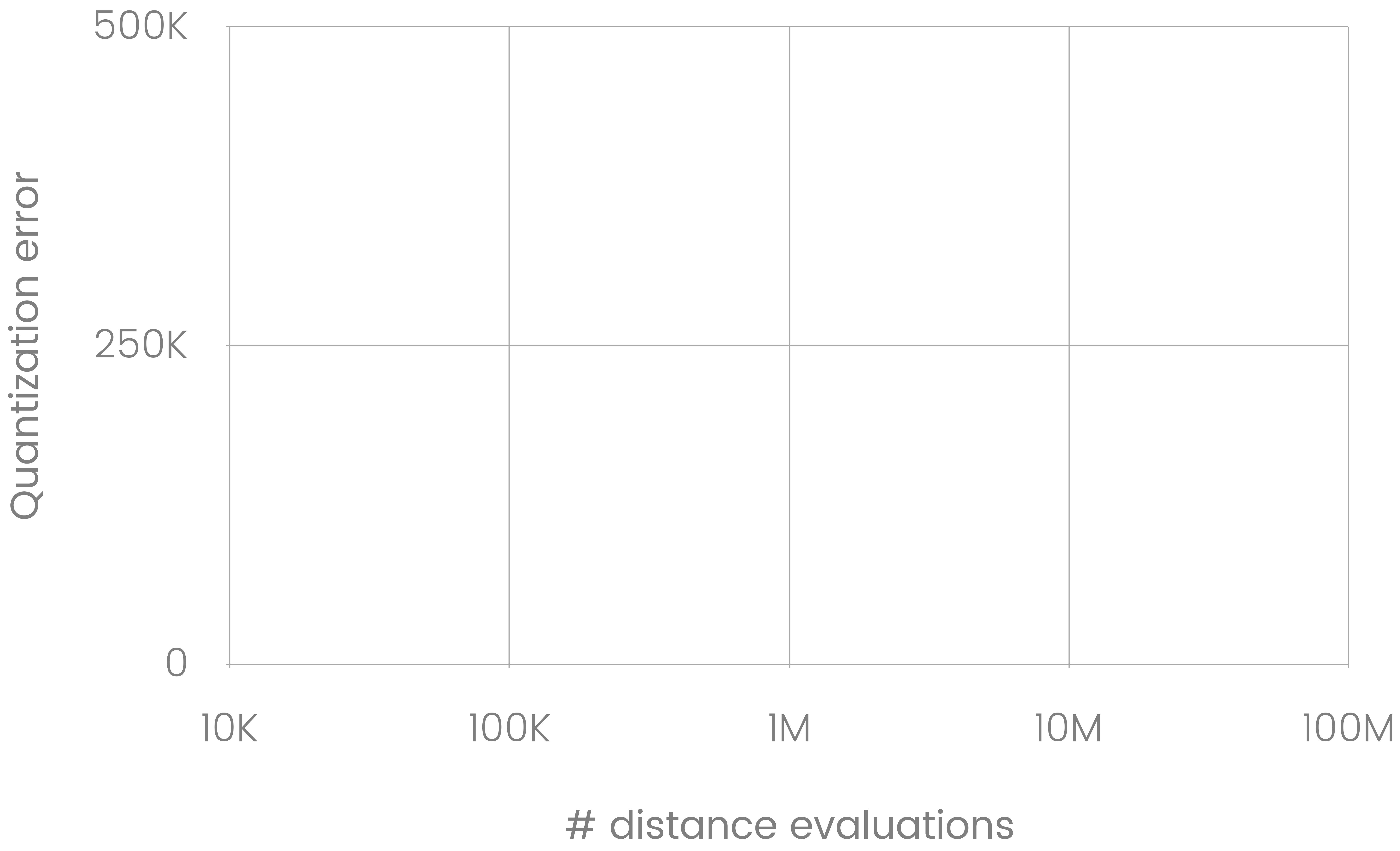
M=100 IS SUFFICIENT IN PRACTICE

Error vs time tradeoff

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

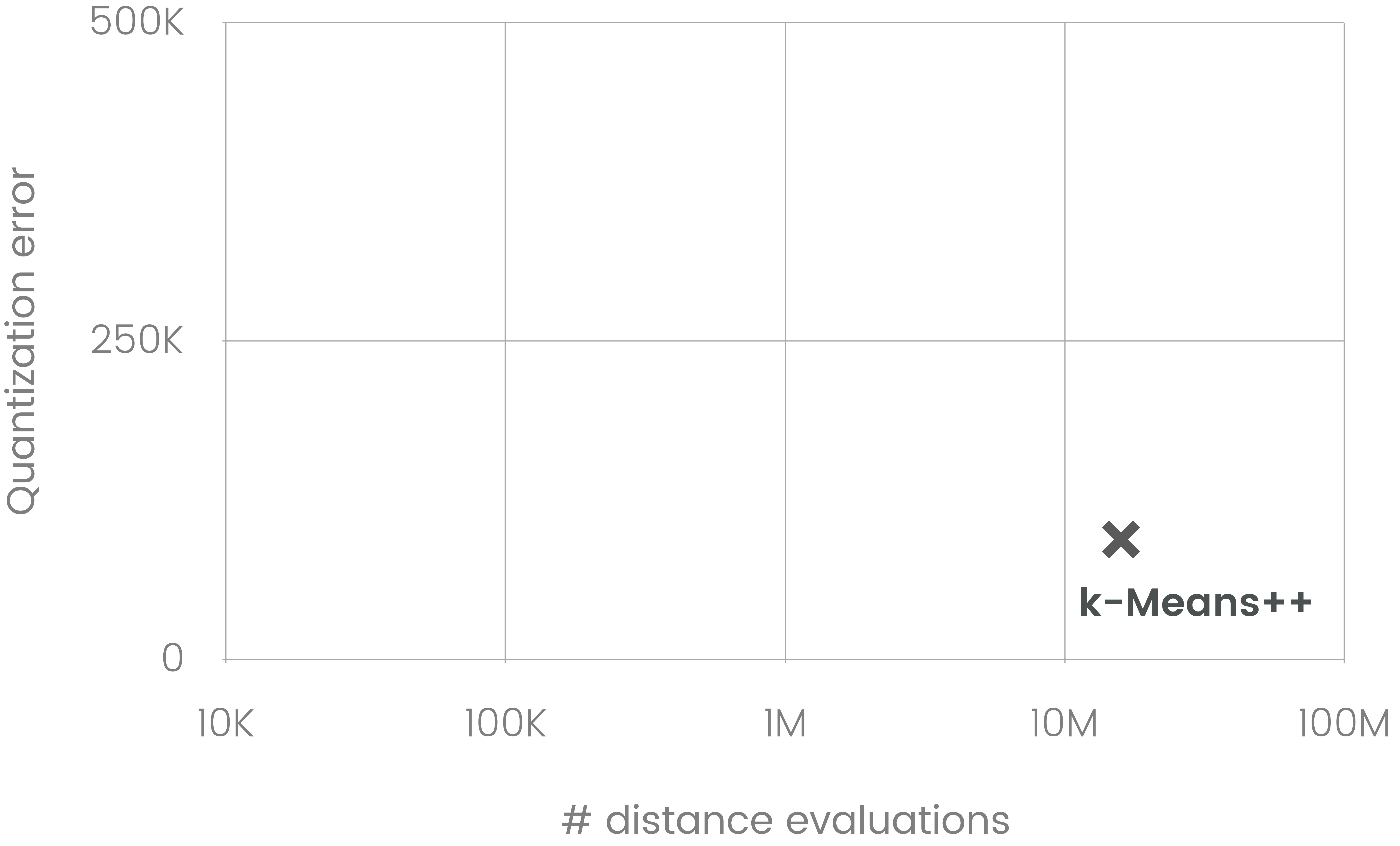
Error vs time tradeoff

CSN



Error vs time tradeoff

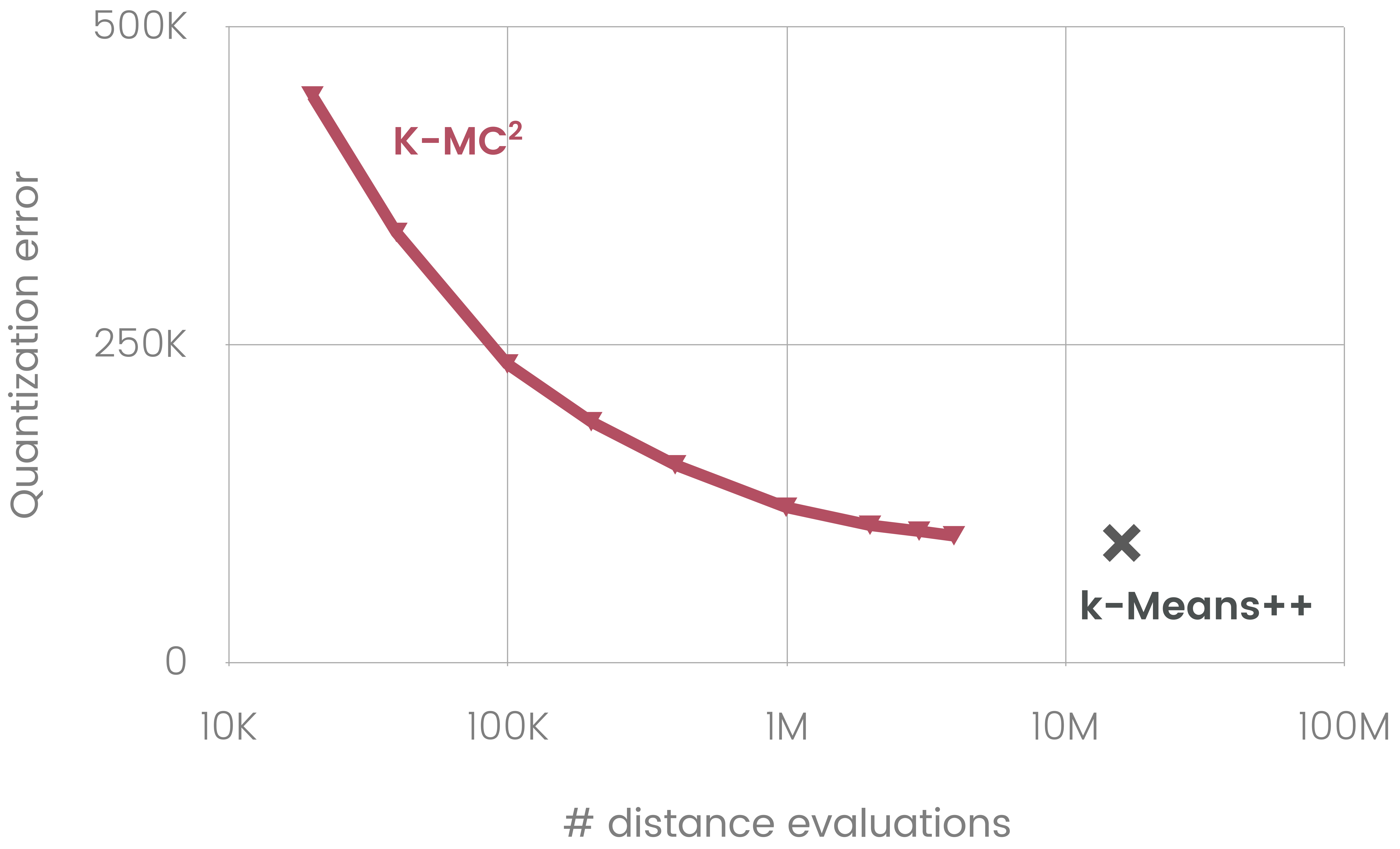
CSN



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

Error vs time tradeoff

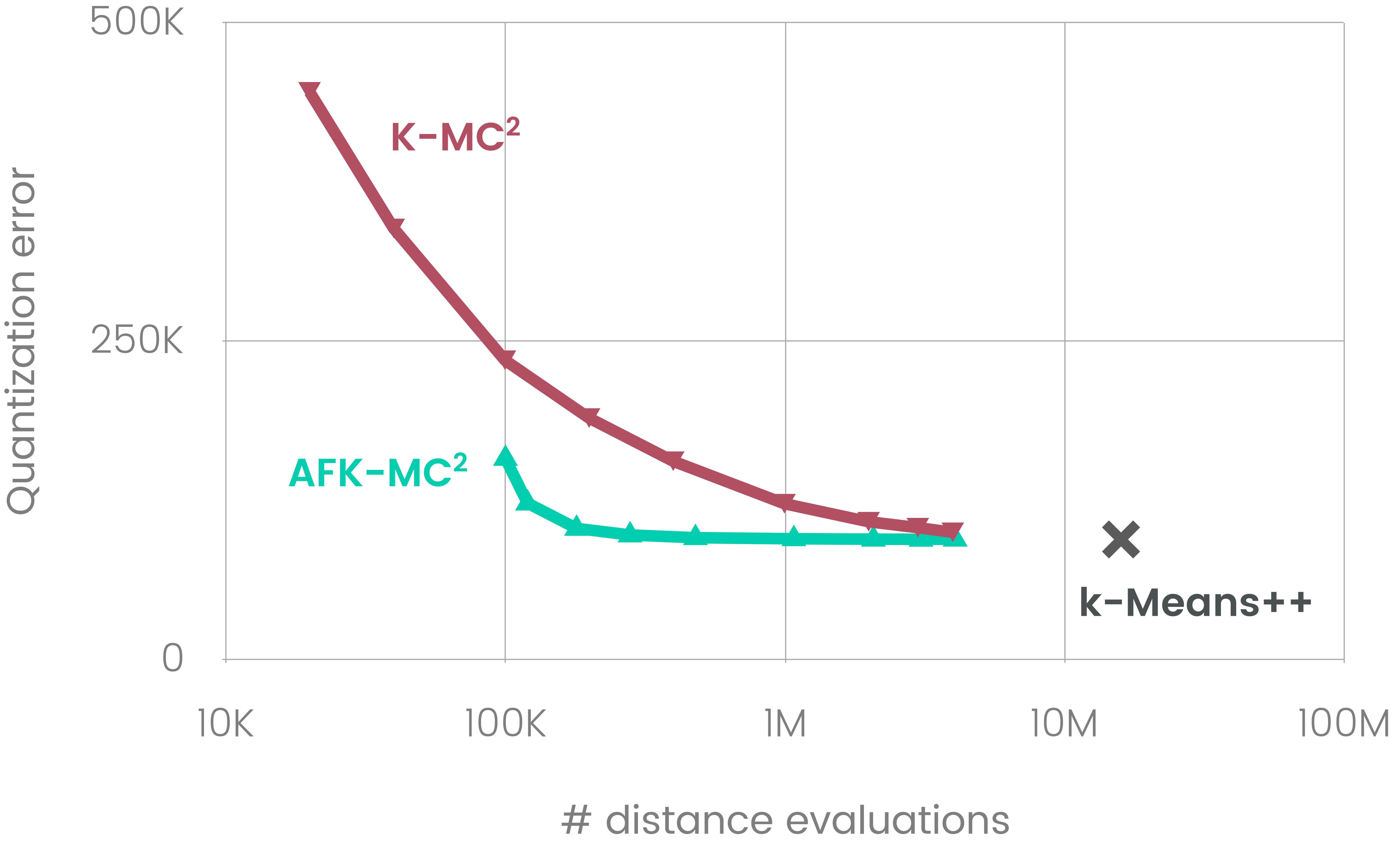
CSN



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

Error vs time tradeoff

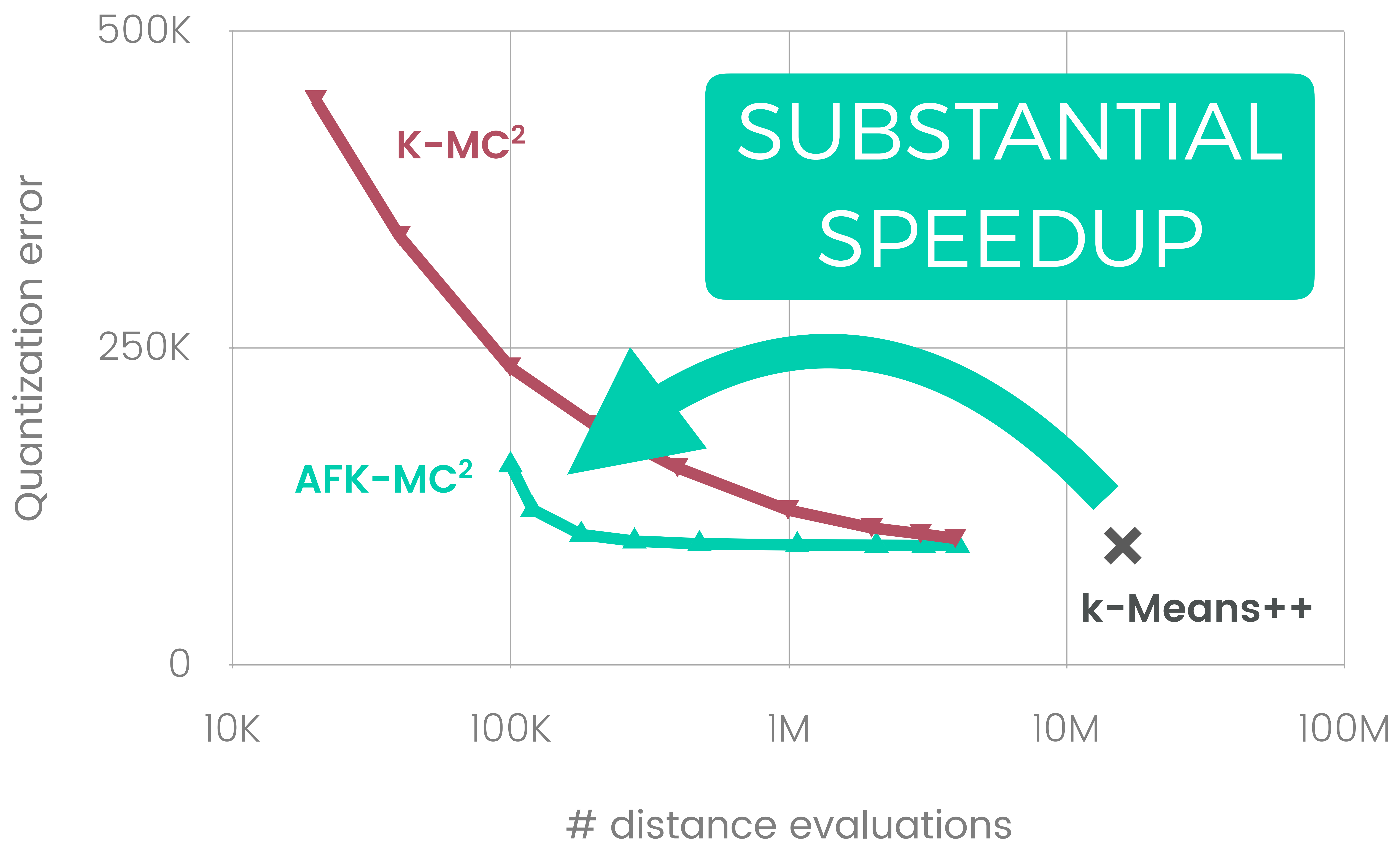
CSN



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

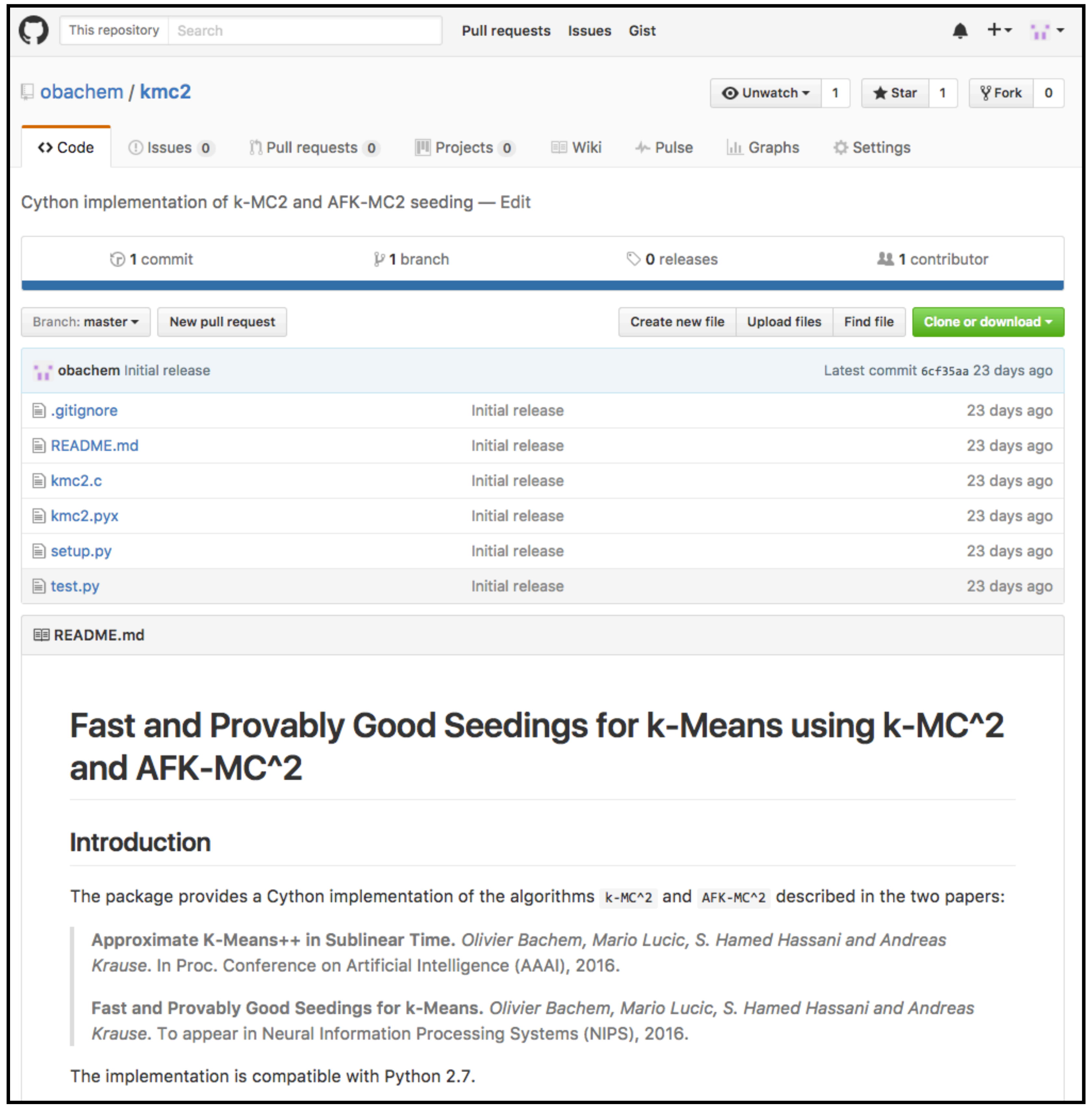
Error vs time tradeoff

CSN



Code

Code



PYTHON IMPLEMENTATION

Available at olivierbachem.ch or with

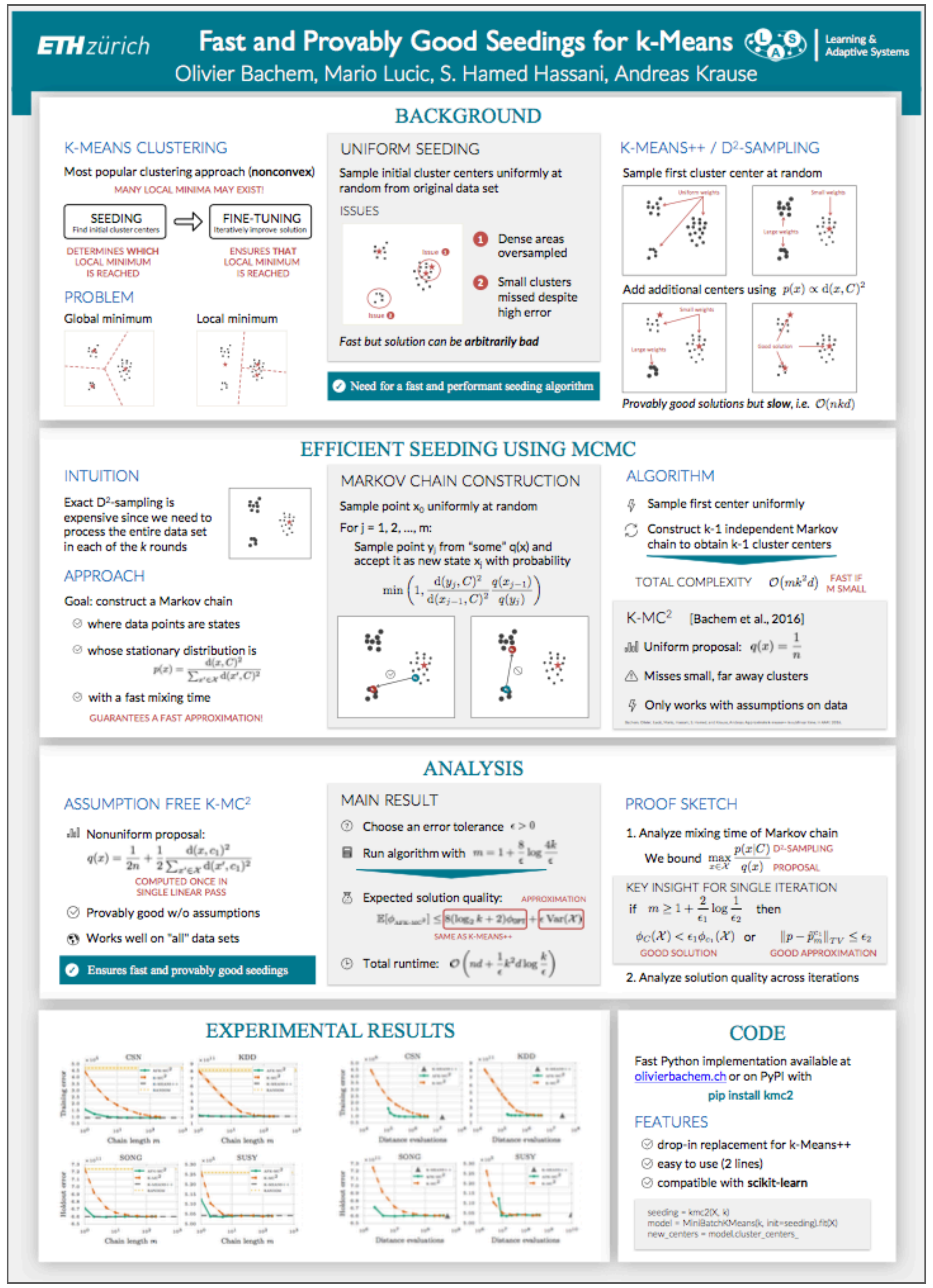
pip install kmc2

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]