# $k$-means

CS 446

# Reminder: unsupervised learning

We only receive $(\boldsymbol{x}_i)_{i=1}^n$;
the goal is not well-defined, but here are a few possibilities:

# Reminder: unsupervised learning

We only receive $(\boldsymbol{x}_i)_{i=1}^n$;
the goal is not well-defined, but here are a few possibilities:

- Encoding data in some compact representation
  (and decoding this).
- Data analysis;
  recovering "hidden structure" in data
  (e.g., recovering cliques or clusters).
- Features for supervised learning.

# Exemplar-based clustering

Today we will study clustering.

- Clustering finds a partition of the data (into clusters),
  putting similar data in one cluster,
  dissimilar data in different clusters.
  (Practical success depends on a good measure of "similarity"!)
- These methods provide a mapping/encoding of data into $\{1, \ldots, k\}$;
  it is akin to multiclass classification, but without labels!
  (Sometimes called "hard clustering";
  "soft clustering" assigns each point a distribution over clusters.)

# Exemplar-based clustering

Today we will study clustering.

- Clustering finds a partition of the data (into clusters),
  putting similar data in one cluster,
  dissimilar data in different clusters.
  (Practical success depends on a good measure of "similarity"!)
- These methods provide a mapping/encoding of data into $\{1, \ldots, k\}$;
  it is akin to multiclass classification, but without labels!
  (Sometimes called "hard clustering";
  "soft clustering" assigns each point a distribution over clusters.)

Today's method will be exemplar-based.

- The method will associate each cluster with a centroid $\boldsymbol{\mu}_j$;
  the centroids $(\boldsymbol{\mu}_j)_{j=1}^k$ suffice define the clustering.

# 1. $k$-means objective function

# $k$-means goal

**Input:** examples $S := (\boldsymbol{x}_i)_{i=1}^n$, integer $k$.
**Output:** centers $(\boldsymbol{\mu})_{i=1}^k$ minimizing

$$\phi_S((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \phi((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \sum_{i=1}^n \min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2,$$

along with encoding $\boldsymbol{x}_i \mapsto \arg\min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$,
decoding $j \mapsto \boldsymbol{\mu}_j$, and clusters $(S_j)_{j=1}^k$ with

$$S_j := \left\{ \boldsymbol{x}_i \in S : \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| \right\}.$$

# $k$-means goal

**Input:** examples $S := (\boldsymbol{x}_i)_{i=1}^n$, integer $k$.
**Output:** centers $(\boldsymbol{\mu})_{i=1}^k$ minimizing

$$\phi_S((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \phi((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \sum_{i=1}^n \min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2,$$

along with encoding $\boldsymbol{x}_i \mapsto \arg\min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$,
decoding $j \mapsto \boldsymbol{\mu}_j$, and clusters $(S_j)_{j=1}^k$ with

$$S_j := \left\{ \boldsymbol{x}_i \in S : \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| \right\}.$$

**Is this an algorithm?**
Nope, it's just an objective function, and it's NP-hard even when $d = 2$.

# $k$-means goal

**Input:** examples $S := (\boldsymbol{x}_i)_{i=1}^n$, integer $k$.
**Output:** centers $(\boldsymbol{\mu})_{i=1}^k$ minimizing

$$\phi_S((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \phi((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \sum_{i=1}^n \min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2,$$

along with encoding $\boldsymbol{x}_i \mapsto \arg\min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$,
decoding $j \mapsto \boldsymbol{\mu}_j$, and clusters $(S_j)_{j=1}^k$ with

$$S_j := \left\{ \boldsymbol{x}_i \in S : \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| \right\}.$$

**Does it fit our "unsupervised learning goals"?**

- Encoding/decoding: needs $\mathcal{O}(\ln(k))$ bits for each, plus $\mathcal{O}(kd)$ for centers; called "vector quantization".
- "Hidden stucture": this is the main goal, extracting $(S_j)_{j=1}^k$.
- Feature learning: we'll have a homework problem on this!

# $k$-means goal

**Input:** examples $S := (\boldsymbol{x}_i)_{i=1}^n$, integer $k$.
**Output:** centers $(\boldsymbol{\mu})_{i=1}^k$ minimizing

$$\phi_S((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \phi((\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)) = \sum_{i=1}^n \min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2,$$

along with encoding $\boldsymbol{x}_i \mapsto \arg\min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$,
decoding $j \mapsto \boldsymbol{\mu}_j$, and clusters $(S_j)_{j=1}^k$ with

$$S_j := \left\{ \boldsymbol{x}_i \in S : \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| \right\}.$$

**Does it make sense? Are the clusters meaningful?**
Maybe: depends on data, and how well it is suited to $k$ and $\|\cdot\|$ (or some similarity measure replacing it).

# Alternate form with explicit assignments

Given $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and centers $\boldsymbol{C} := (\mu_1, \ldots, \mu_k)$, define $k$-means cost

$$\phi_S(\boldsymbol{C}) := \phi(\boldsymbol{C}) := \sum_{\boldsymbol{x} \in S} \min_{\mu \in \boldsymbol{C}} \|\boldsymbol{x} - \mu\|^2.$$

# Alternate form with explicit assignments

Given $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and centers $\boldsymbol{C} := (\mu_1, \ldots, \mu_k)$, define $k$-means cost

$$\phi_S(\boldsymbol{C}) := \phi(\boldsymbol{C}) := \sum_{\boldsymbol{x} \in S} \min_{\mu \in \boldsymbol{C}} \|\boldsymbol{x} - \mu\|^2.$$

**Refined form with explicit assignments:**

- Let $\mathcal{C}_{d,k} := \mathcal{C} := \mathbb{R}^{d \times k}$ denote valid centers;
- Let $\mathcal{A}_{n,k} := \mathcal{A} := \{\boldsymbol{A} \in \{0,1\}^{n \times k} : \boldsymbol{A}\mathbf{1}_k = \mathbf{1}_n\}$ denote valid assignments of examples to centers.

Given centers $\boldsymbol{C} = (\boldsymbol{\mu}_1^\mathsf{T}, \ldots, \boldsymbol{\mu}_k^\mathsf{T}) \in \mathcal{C}_{d,k}$ and assignment $\boldsymbol{A} \in \mathcal{A}_{n,k}$,

$$\phi_S(\boldsymbol{C}; \boldsymbol{A}) := \phi(\boldsymbol{C}; \boldsymbol{A}) := \sum_{i=1}^{n} \sum_{j=1}^{k} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2.$$

(Lots of bad notation this lecture.)

# Alternate form with explicit assignments

Given $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and centers $\boldsymbol{C} := (\mu_1, \ldots, \mu_k)$, define $k$-means cost

$$\phi_S(\boldsymbol{C}) := \phi(\boldsymbol{C}) := \sum_{\boldsymbol{x} \in S} \min_{\mu \in \boldsymbol{C}} \|\boldsymbol{x} - \mu\|^2.$$

**Refined form with explicit assignments:**

- Let $\mathcal{C}_{d,k} := \mathcal{C} := \mathbb{R}^{d \times k}$ denote valid centers;
- Let $\mathcal{A}_{n,k} := \mathcal{A} := \{\boldsymbol{A} \in \{0,1\}^{n \times k} : \boldsymbol{A}\mathbf{1}_k = \mathbf{1}_n\}$ denote valid assignments of examples to centers.

Given centers $\boldsymbol{C} = (\boldsymbol{\mu}_1^\mathsf{T}, \ldots, \boldsymbol{\mu}_k^\mathsf{T}) \in \mathcal{C}_{d,k}$ and assignment $\boldsymbol{A} \in \mathcal{A}_{n,k}$,

$$\phi_S(\boldsymbol{C}; \boldsymbol{A}) := \phi(\boldsymbol{C}; \boldsymbol{A}) := \sum_{i=1}^{n} \sum_{j=1}^{k} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2.$$

(Lots of bad notation this lecture.)

**Who cares?**

# Alternate form with explicit assignments

Given $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and centers $\boldsymbol{C} := (\mu_1, \ldots, \mu_k)$, define $k$-means cost

$$\phi_S(\boldsymbol{C}) := \phi(\boldsymbol{C}) := \sum_{\boldsymbol{x} \in S} \min_{\mu \in \boldsymbol{C}} \|\boldsymbol{x} - \mu\|^2.$$

**Refined form with explicit assignments:**

- Let $\mathcal{C}_{d,k} := \mathcal{C} := \mathbb{R}^{d \times k}$ denote valid centers;
- Let $\mathcal{A}_{n,k} := \mathcal{A} := \{\boldsymbol{A} \in \{0,1\}^{n \times k} : \boldsymbol{A}\mathbf{1}_k = \mathbf{1}_n\}$ denote valid assignments of examples to centers.

Given centers $\boldsymbol{C} = (\boldsymbol{\mu}_1^\mathsf{T}, \ldots, \boldsymbol{\mu}_k^\mathsf{T}) \in \mathcal{C}_{d,k}$ and assignment $\boldsymbol{A} \in \mathcal{A}_{n,k}$,

$$\phi_S(\boldsymbol{C}; \boldsymbol{A}) := \phi(\boldsymbol{C}; \boldsymbol{A}) := \sum_{i=1}^{n} \sum_{j=1}^{k} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2.$$

(Lots of bad notation this lecture.)

**Who cares?**

$\phi$ is NP-hard to minimize, but:

if we fix the assignment, the centers are easy to find;

if we fix the centers, the assignment is easy to find.

# "Easy to find?"

- Given assignment $\boldsymbol{A} \in \mathcal{A}_{n,k} = \mathcal{A}$, define optimal centers $\mathcal{C}_{d,k}(\boldsymbol{A}) = \mathcal{C}(\boldsymbol{A})$ to be the mean of points with common assignment:

$$\mathcal{C}(\boldsymbol{A})\mathbf{e}_j = \mu_j = \text{mean}(\{\boldsymbol{x}_i : A_{ij} = j\}) \qquad \forall j.$$

  Computing $\mathcal{C}(\boldsymbol{A})$ takes time $\mathcal{O}(nkd)$.

# "Easy to find?"

- Given assignment $\boldsymbol{A} \in \mathcal{A}_{n,k} = \mathcal{A}$, define optimal centers $\mathcal{C}_{d,k}(\boldsymbol{A}) = \mathcal{C}(\boldsymbol{A})$ to be the mean of points with common assignment:

$$\mathcal{C}(\boldsymbol{A})\mathbf{e}_j = \mu_j = \text{mean}(\{\boldsymbol{x}_i : A_{ij} = j\}) \qquad \forall j.$$

  Computing $\mathcal{C}(\boldsymbol{A})$ takes time $\mathcal{O}(nkd)$.

- Given centers $\boldsymbol{C} \in \mathcal{C}_{d,k} = \mathcal{C}$, define optimal assignment $\mathcal{A}_{n,k}(\boldsymbol{C}) = \mathcal{A}(\boldsymbol{C}) \in \mathcal{A}_{n,k}$ to associate data points with closest centers:

$$A_{ij} = 1 \qquad \Longrightarrow \qquad \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_l\| \qquad \forall i, j.$$

  (Ties broken in an arbitrary but consistent manner).
  Computing $\mathcal{A}(\boldsymbol{C})$ takes time $\mathcal{O}(nkd)$.

# "Easy to find?"

- Given assignment $\boldsymbol{A} \in \mathcal{A}_{n,k} = \mathcal{A}$, define optimal centers $\mathcal{C}_{d,k}(\boldsymbol{A}) = \mathcal{C}(\boldsymbol{A})$ to be the mean of points with common assignment:

$$\mathcal{C}(\boldsymbol{A})\mathbf{e}_j = \mu_j = \text{mean}(\{\boldsymbol{x}_i : A_{ij} = j\}) \qquad \forall j.$$

  Computing $\mathcal{C}(\boldsymbol{A})$ takes time $\mathcal{O}(nkd)$.

- Given centers $\boldsymbol{C} \in \mathcal{C}_{d,k} = \mathcal{C}$, define optimal assignment $\mathcal{A}_{n,k}(\boldsymbol{C}) = \mathcal{A}(\boldsymbol{C}) \in \mathcal{A}_{n,k}$ to associate data points with closest centers:

$$A_{ij} = 1 \qquad \Longrightarrow \qquad \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\| = \min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_l\| \qquad \forall i, j.$$

  (Ties broken in an arbitrary but consistent manner).
  Computing $\mathcal{A}(\boldsymbol{C})$ takes time $\mathcal{O}(nkd)$.

Can we really say "optimal"?

# Optimal assignements and centerings

**Theorem.**
Let data $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ be given.

- Given centers $\boldsymbol{C} \in \mathcal{C}_{d,k}$, then

$$\min_{\boldsymbol{A} \in \mathcal{A}_{n,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) = \phi(\boldsymbol{C}; \mathcal{A}(\boldsymbol{C})) = \phi(\boldsymbol{C}).$$

- Given assignment $\boldsymbol{A} \in \mathcal{A}_{n,k}$, then

$$\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) = \phi(\mathcal{C}(\boldsymbol{A}); \boldsymbol{A}).$$

# Optimal assignements and centerings

**Theorem.**
Let data $S := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ be given.

- Given centers $\boldsymbol{C} \in \mathcal{C}_{d,k}$, then

$$\min_{\boldsymbol{A} \in \mathcal{A}_{n,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) = \phi(\boldsymbol{C}; \mathcal{A}(\boldsymbol{C})) = \phi(\boldsymbol{C}).$$

- Given assignment $\boldsymbol{A} \in \mathcal{A}_{n,k}$, then

$$\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) = \phi(\mathcal{C}(\boldsymbol{A}); \boldsymbol{A}).$$

**Remarks.**
This *does* suggest an algorithm; we'll get to that.

**Proof.**

- Directly,

$$\min_{\boldsymbol{A}\in\mathcal{A}_{n,k}} \phi(\boldsymbol{C};\boldsymbol{A}) = \sum_{\boldsymbol{x}_i\in S} \min_{a\in\mathcal{A}_{1,k}} \sum_{j=1}^{k} a_j \|\boldsymbol{x}_i - \mu_j\|^2 = \sum_{\boldsymbol{x}_i\in S} \min_j \|\boldsymbol{x}_i - \mu_j\|^2$$
$$= \sum_{\boldsymbol{x}_i\in S} \mathcal{A}(\boldsymbol{C})_{i,j} \|\boldsymbol{x}_i - \mu_j\|^2.$$

□

**Proof.**

- Directly,

$$\min_{\boldsymbol{A}\in\mathcal{A}_{n,k}} \phi(\boldsymbol{C};\boldsymbol{A}) = \sum_{\boldsymbol{x}_i\in S} \min_{a\in\mathcal{A}_{1,k}} \sum_{j=1}^{k} a_j \|\boldsymbol{x}_i - \mu_j\|^2 = \sum_{\boldsymbol{x}_i\in S} \min_j \|\boldsymbol{x}_i - \mu_j\|^2$$
$$= \sum_{\boldsymbol{x}_i\in S} \mathcal{A}(\boldsymbol{C})_{i,j} \|\boldsymbol{x}_i - \mu_j\|^2.$$

- First,

$$\min_{\boldsymbol{C}\in\mathcal{C}_{d,k}} \phi(\boldsymbol{C};\boldsymbol{A}) = \sum_{j=1}^{k} \min_{\boldsymbol{\mu}_j\in\mathbb{R}^d} \sum_{i=1}^{n} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|.$$

Taking the gradient with respect to $\boldsymbol{\mu}_j$ and setting to 0,

$$\sum_{i=1}^{n} 2A_{i,j}(\boldsymbol{x}_i - \boldsymbol{\mu}_j) = 0$$
$$\implies \qquad \boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \mathbb{1}[A_{ij}=1]}{\sum_{i=1}^{n} \mathbb{1}[A_{ij}=1]} = \text{mean}(\{\boldsymbol{x}_i : A_{ij}=1\}).$$

□

## 2. Lloyd's method ($k$-means algorithm)

# $k$-means algorithms.

Algorithmic facts so far:

- Minimizing $\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C})$ is NP-hard.
- Joint minimization $\min\limits_{\substack{\boldsymbol{C} \in \mathcal{C}_{d,k} \\ \boldsymbol{A} \in \mathcal{A}_{n,k}}} \phi(\boldsymbol{C}; \boldsymbol{A})$ is NP-hard

# $k$-means algorithms.

Algorithmic facts so far:

- Minimizing $\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C})$ is NP-hard.
- Joint minimization $\min\limits_{\substack{\boldsymbol{C} \in \mathcal{C}_{d,k} \\ \boldsymbol{A} \in \mathcal{A}_{n,k}}} \phi(\boldsymbol{C}; \boldsymbol{A})$ is NP-hard
- Individual minimizations

$$\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) \qquad \text{and} \min_{\boldsymbol{A} \in \mathcal{A}_{n,k}} \phi(\boldsymbol{C}; \boldsymbol{A})$$

  are easy (exact solution in time $\mathcal{O}(nkd)$).

# $k$-means algorithms.

Algorithmic facts so far:

- Minimizing $\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C})$ is NP-hard.
- Joint minimization $\min\limits_{\substack{\boldsymbol{C} \in \mathcal{C}_{d,k} \\ \boldsymbol{A} \in \mathcal{A}_{n,k}}} \phi(\boldsymbol{C}; \boldsymbol{A})$ is NP-hard
- Individual minimizations

$$\min_{\boldsymbol{C} \in \mathcal{C}_{d,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) \qquad \text{and} \min_{\boldsymbol{A} \in \mathcal{A}_{n,k}} \phi(\boldsymbol{C}; \boldsymbol{A})$$

  are easy (exact solution in time $\mathcal{O}(nkd)$).

**Last part gives the standard algorithm!**

# Lloyd's method ("$k$-means algorithm")

**Lloyd's method:**

1. Choose initial assignment $A_0 \in \mathcal{A}_{n,k}$.
2. For $t = 1, 2, \ldots$:

   2.1 **(Recenter.)** Set $C_t := \mathcal{C}_{d,k}(A_{t-1})$.
   2.2 **(Reassign).** Set $A_t := \mathcal{A}_{n,k}(C_t)$.
   2.3 If $\phi$ did not change, terminate.

# Lloyd's method ("$k$-means algorithm")

**Lloyd's method:**

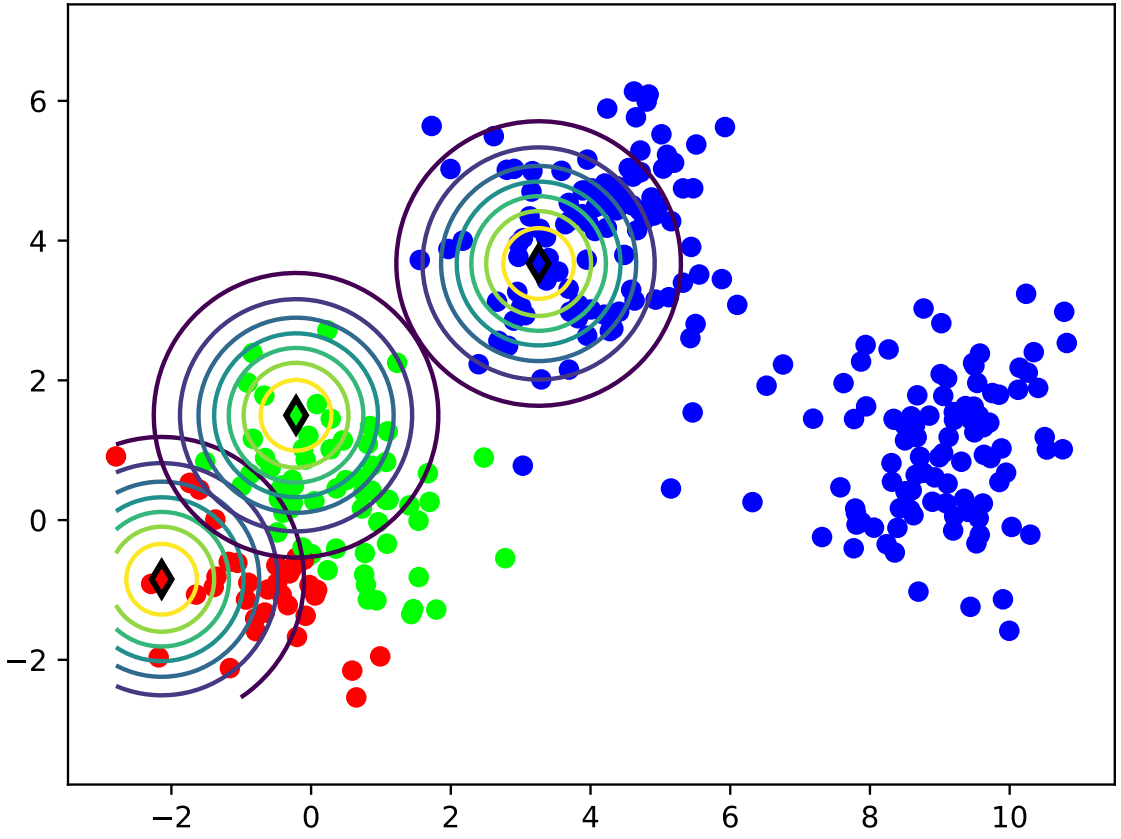
1. Choose initial assignment $A_0 \in \mathcal{A}_{n,k}$.
2. For $t = 1, 2, \ldots$:

   2.1 **(Recenter.)** Set $C_t := \mathcal{C}_{d,k}(A_{t-1})$.
   2.2 **(Reassign).** Set $A_t := \mathcal{A}_{n,k}(C_t)$.
   2.3 If $\phi$ did not change, terminate.

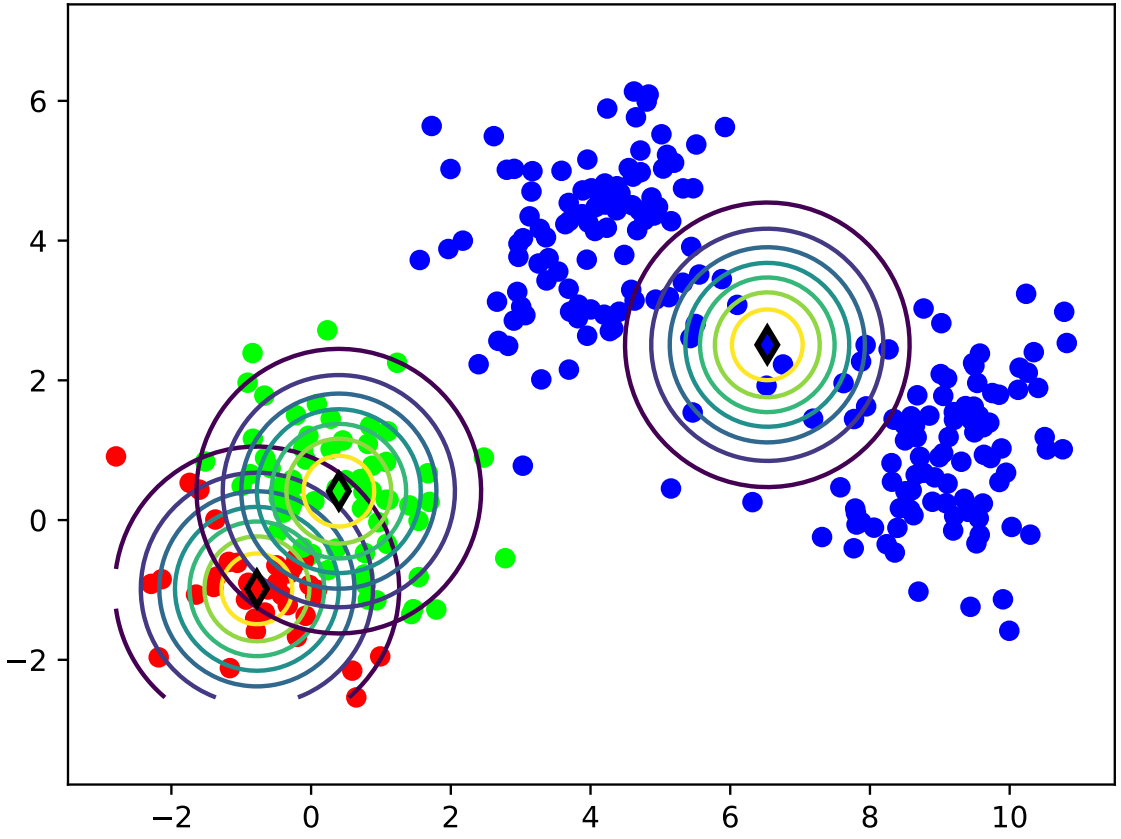
**Remarks.**

- This is **alternating minimization** on cluster assignments and cluster centers; this is a general optimization technique when the joint problem is hard but the individual problems are easy. It is only a heuristic.
- $\mathcal{O}(nkd)$ per iteration (as before).
- Initialization is crucial; standard implementation is now a method a called "`kmeans++`".

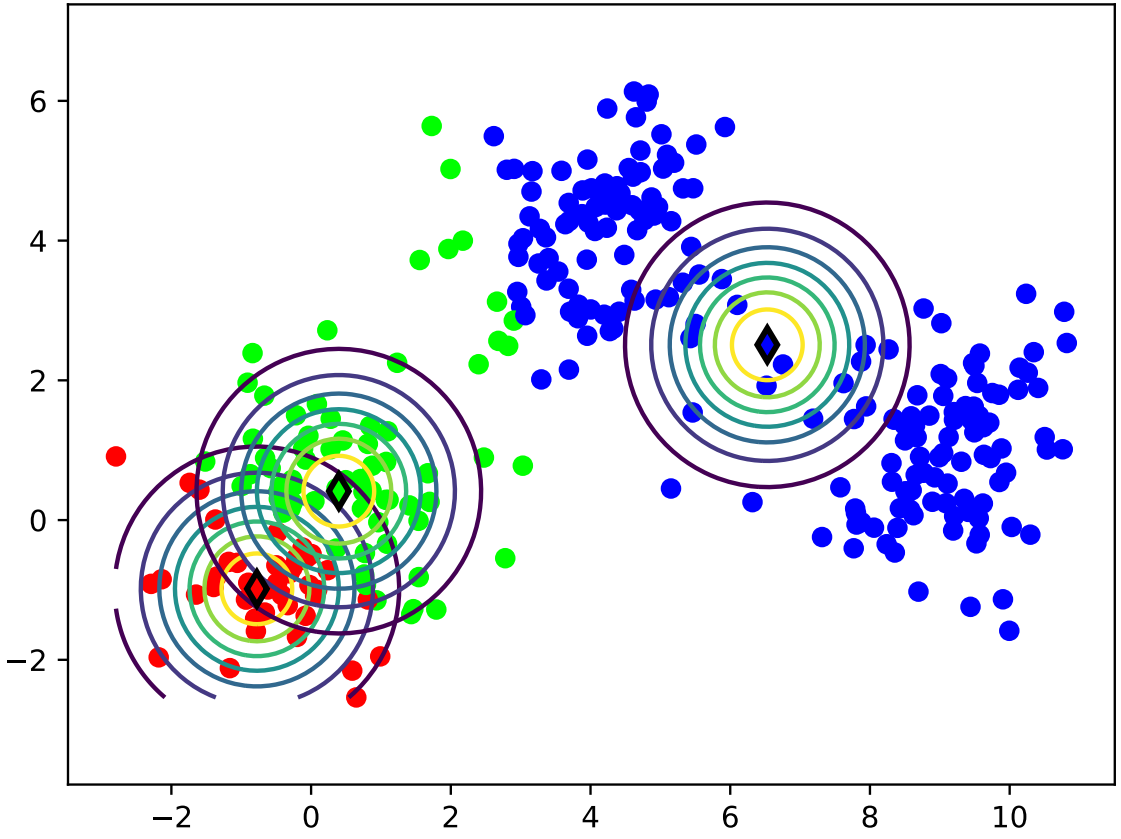
Reassign. . .



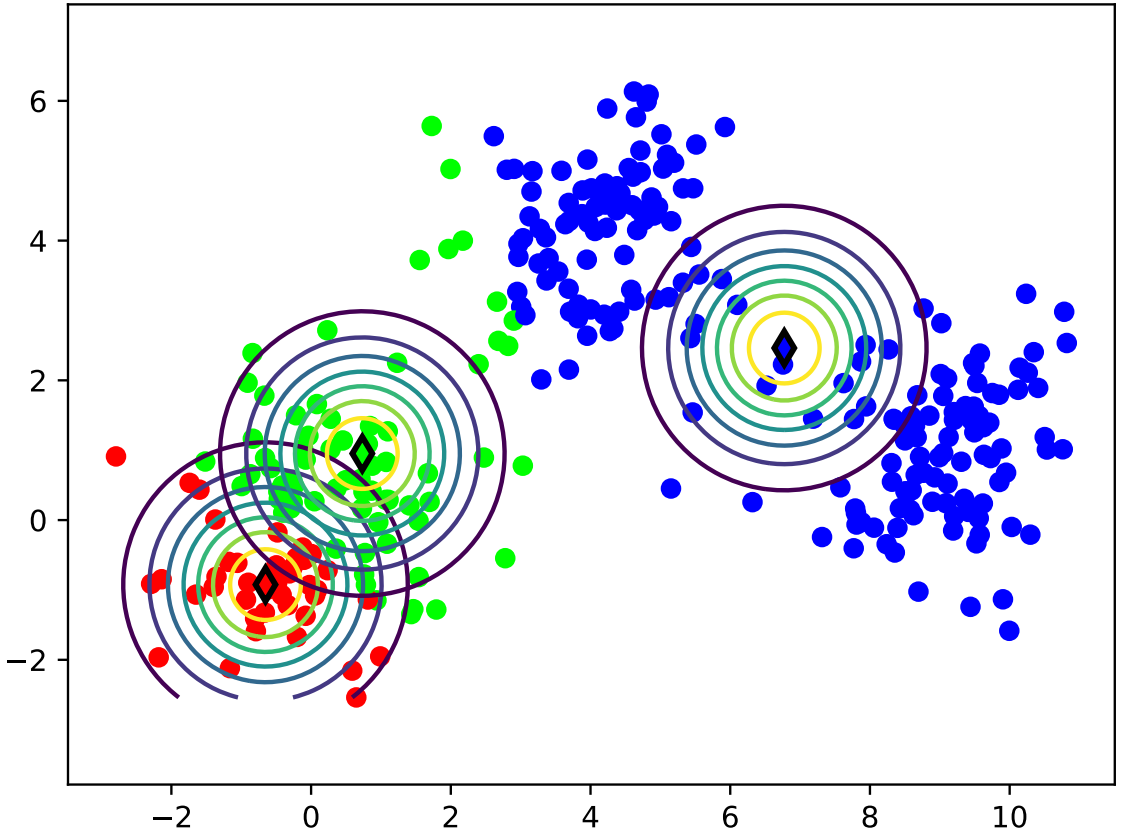
Reassign... Recenter...



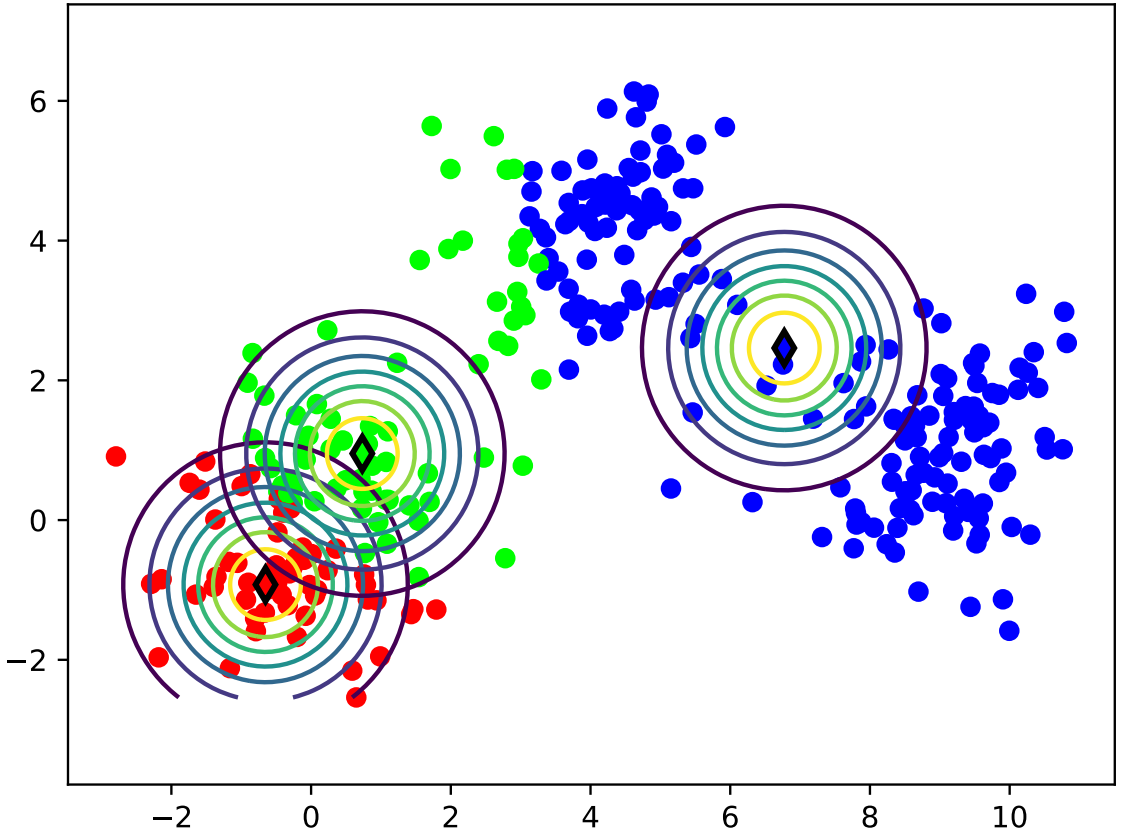
Reassign... Recenter... Reassign...



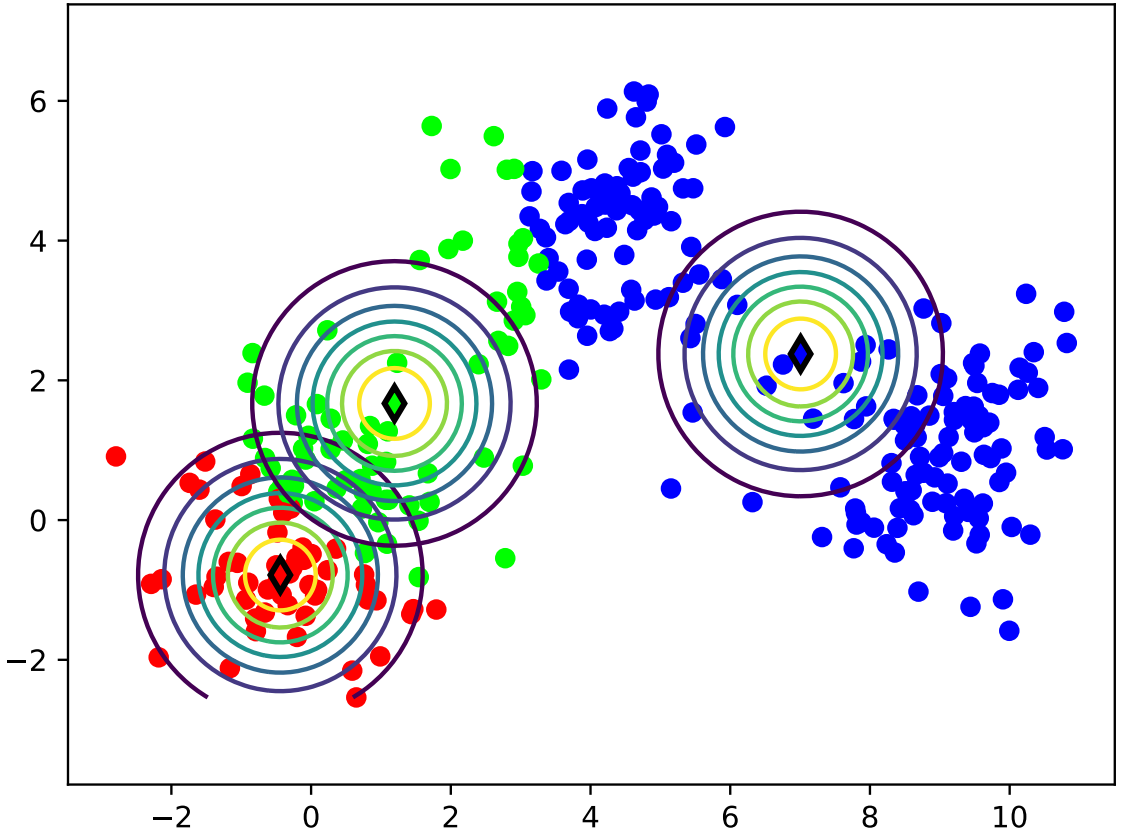
Reassign... Recenter... Reassign... Recenter...



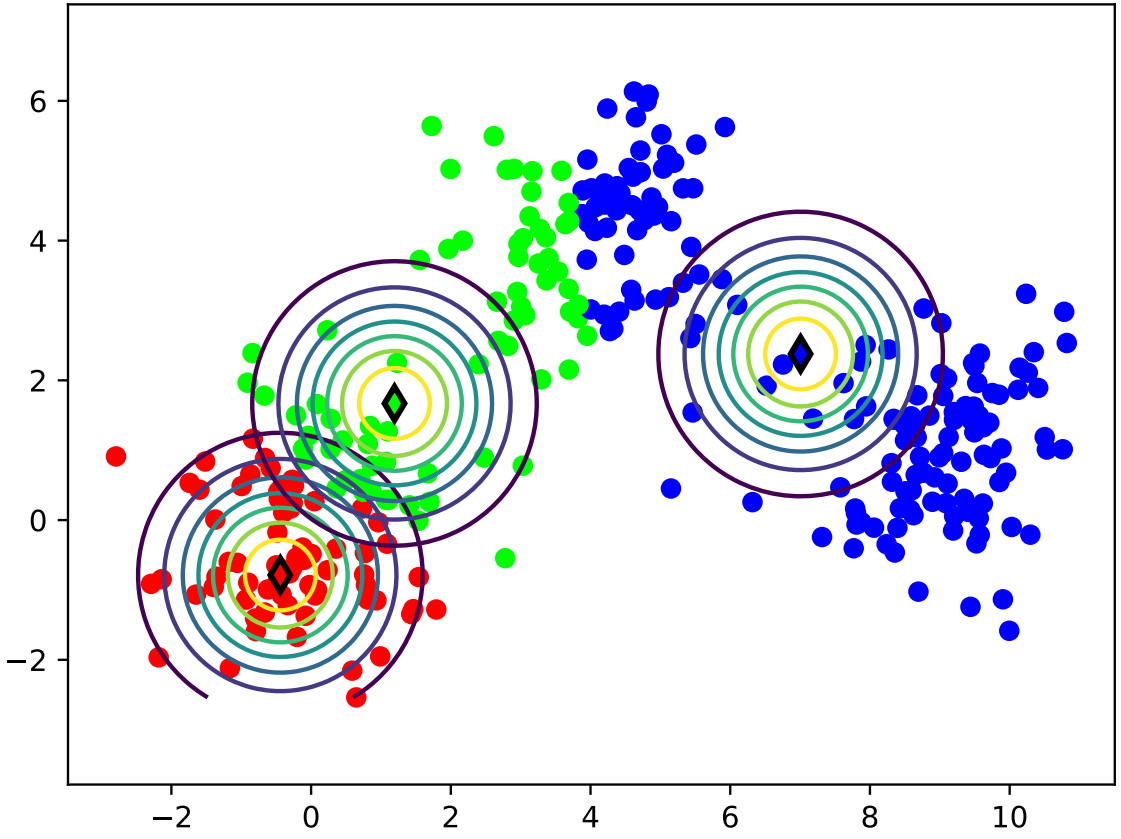
Reassign... Recenter... Reassign... Recenter... Reassign...



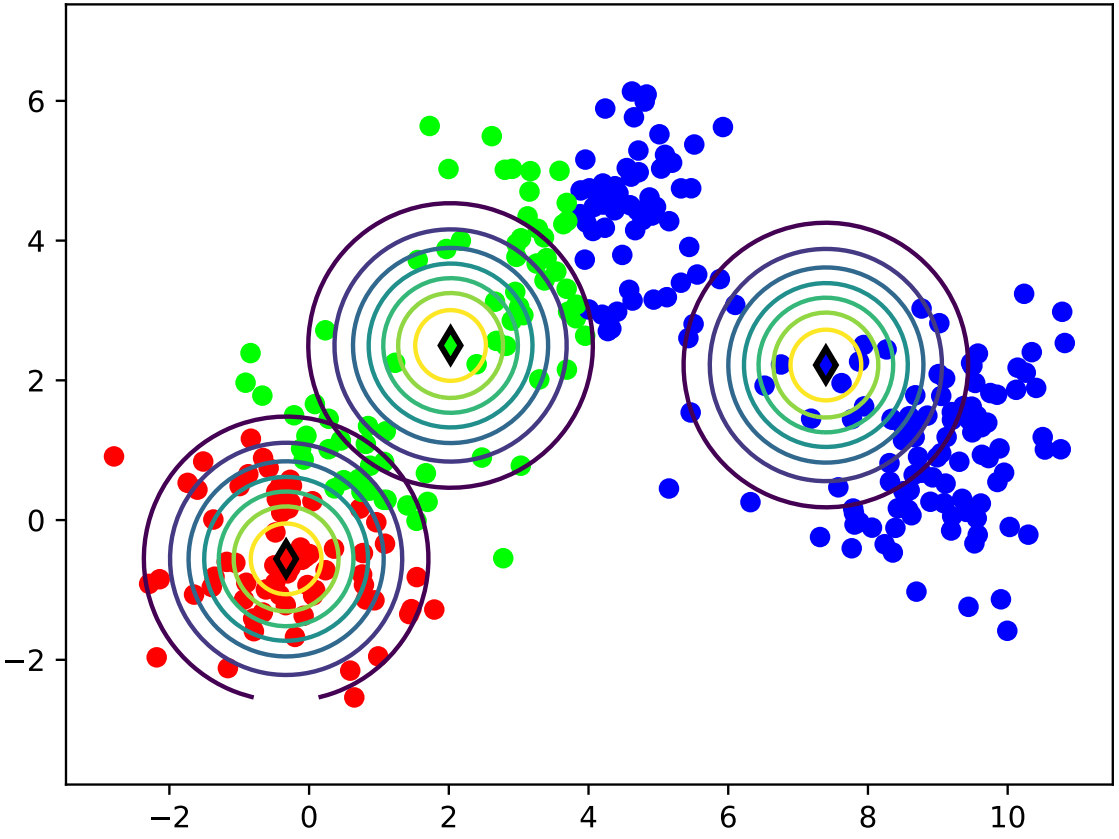
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...



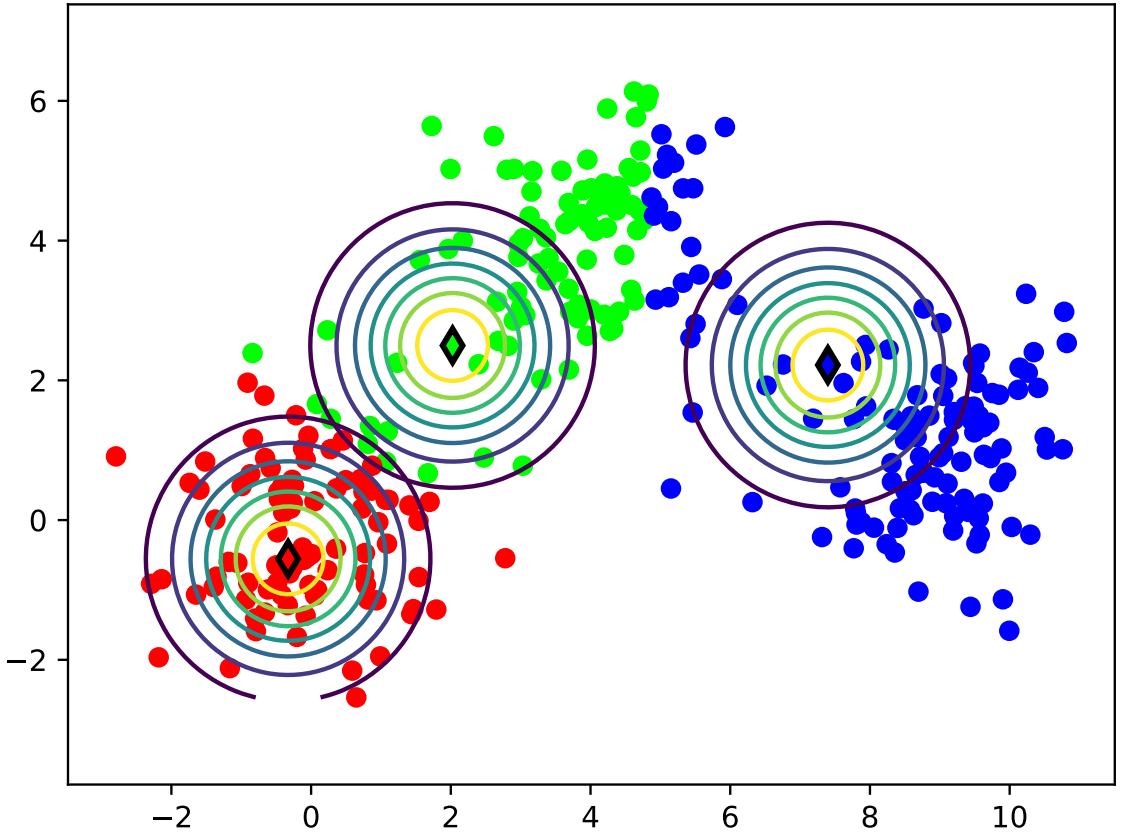
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter... Reassign...



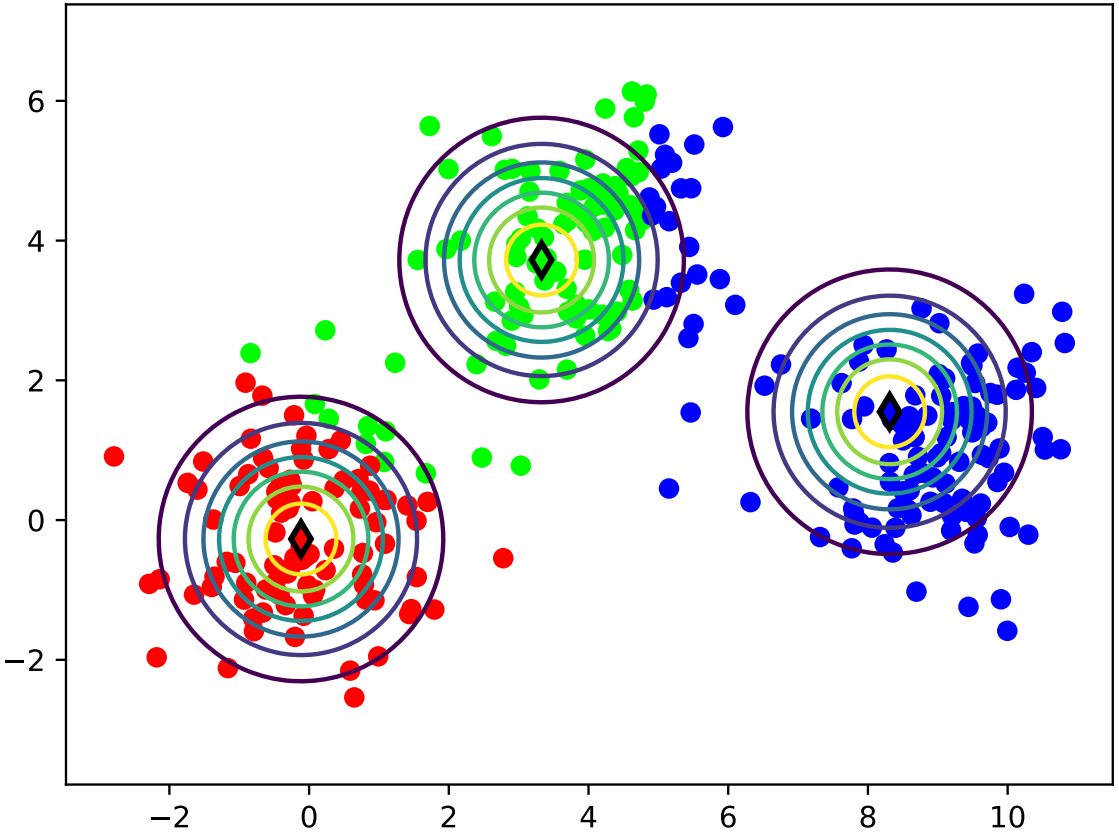
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter...



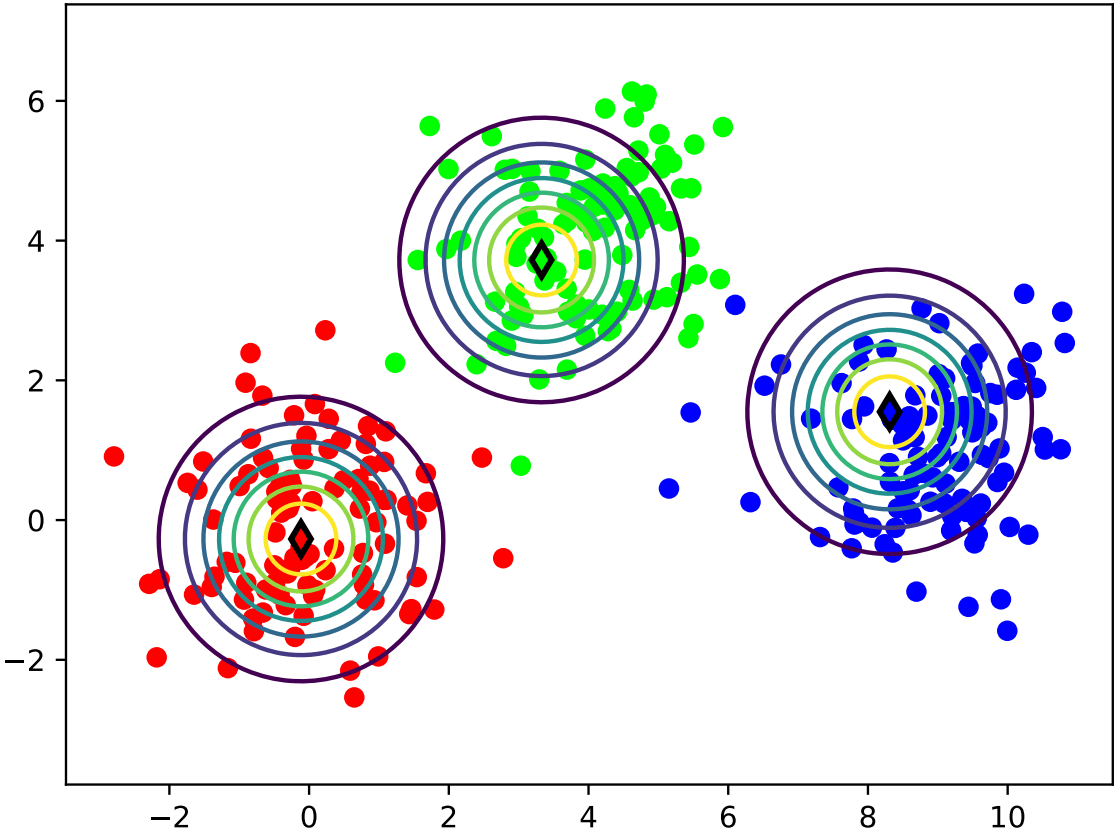
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign...



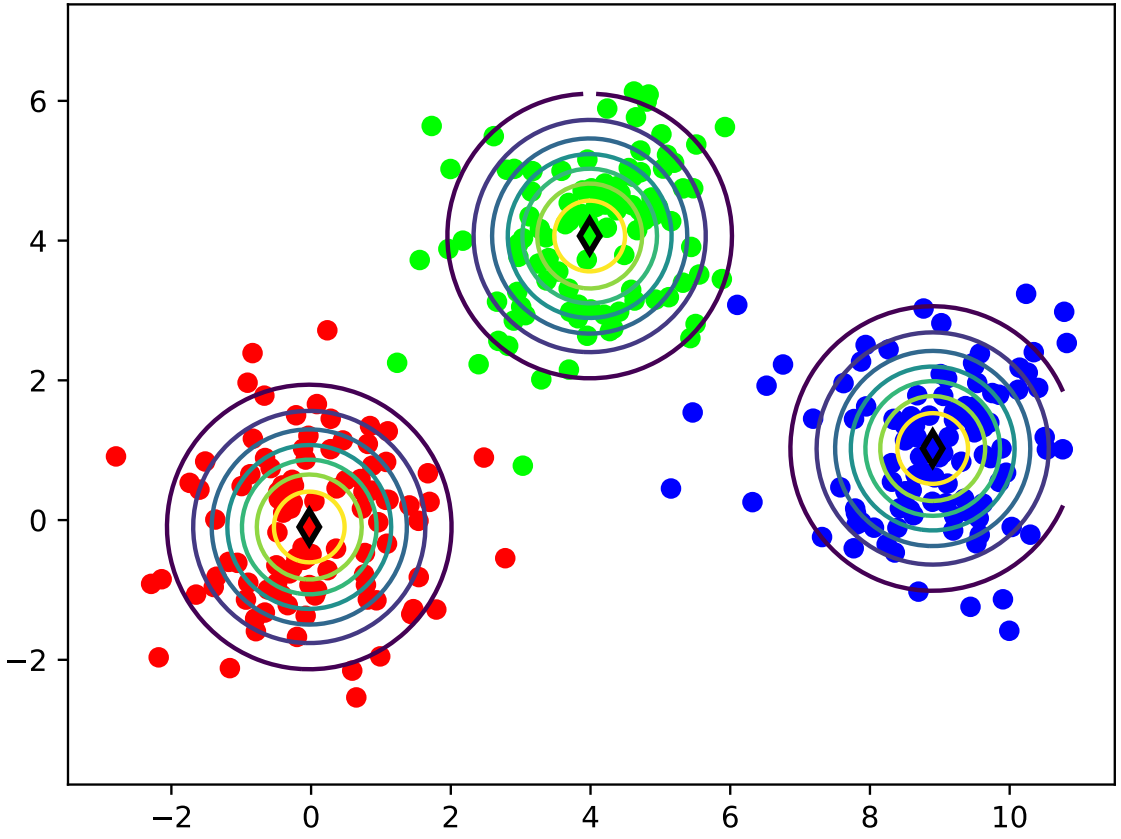
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter...



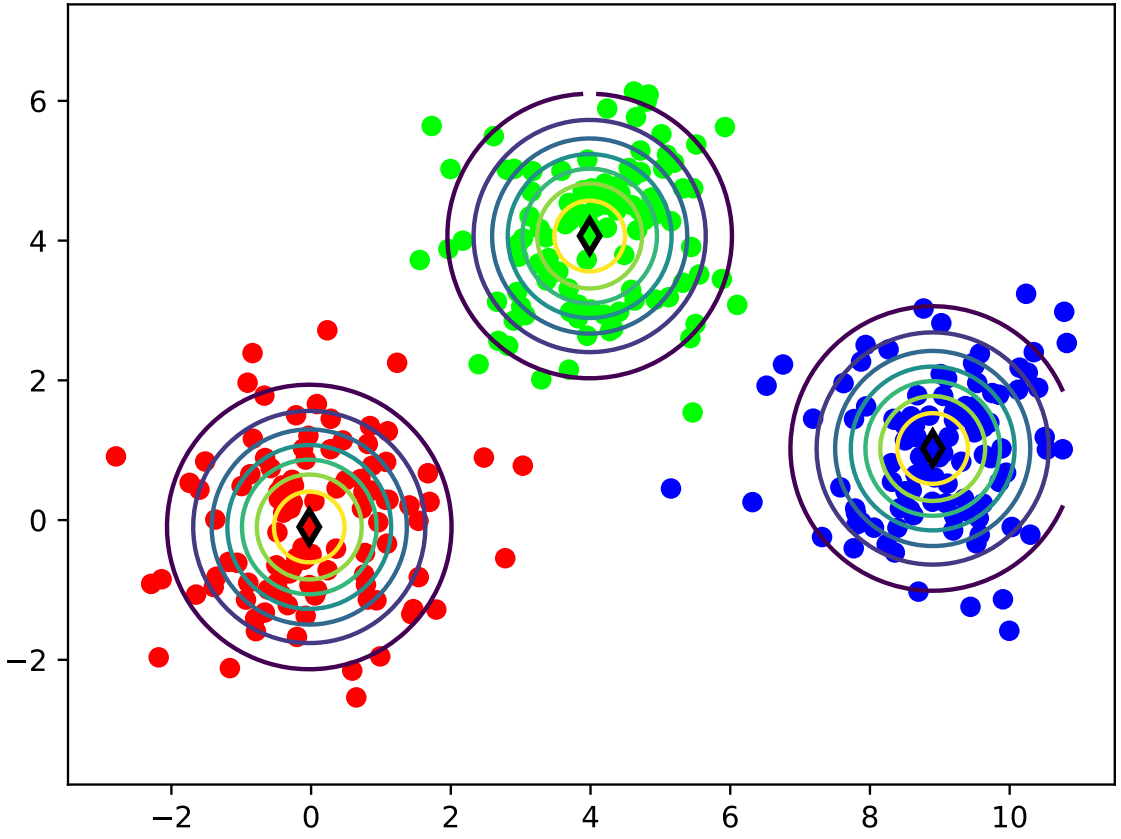
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter... Reassign...



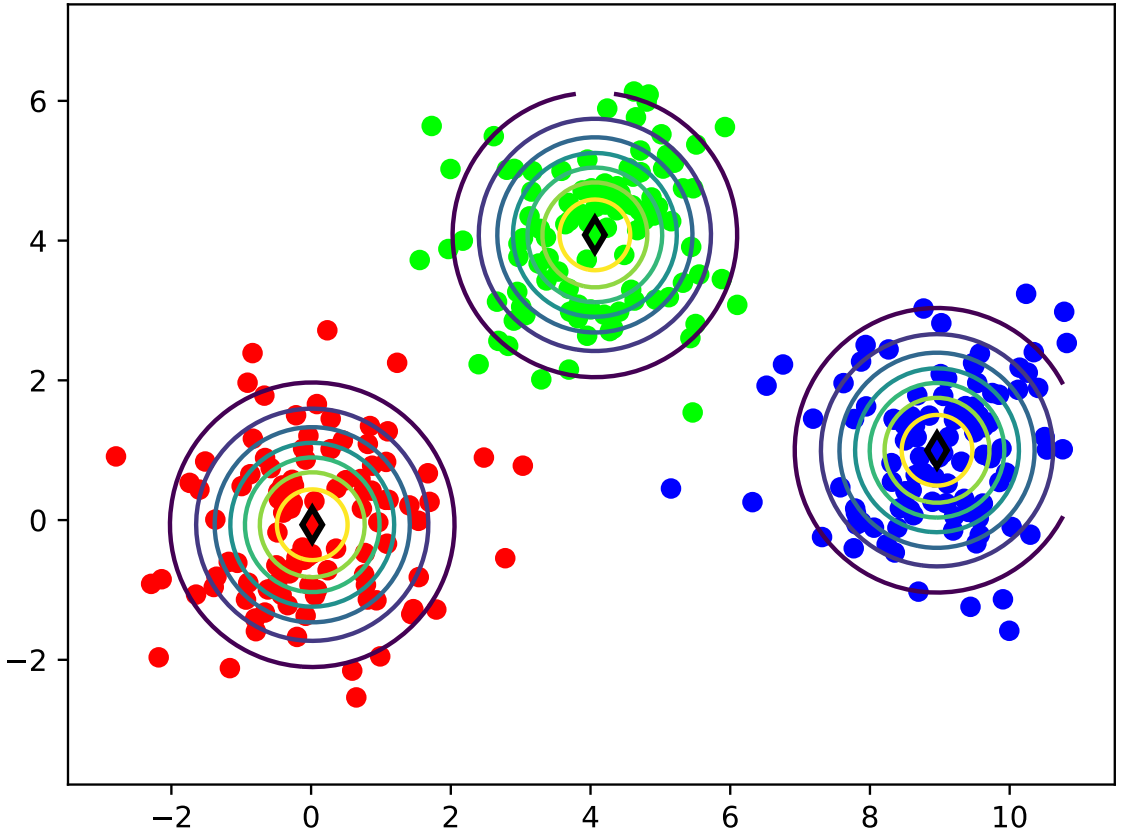
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...



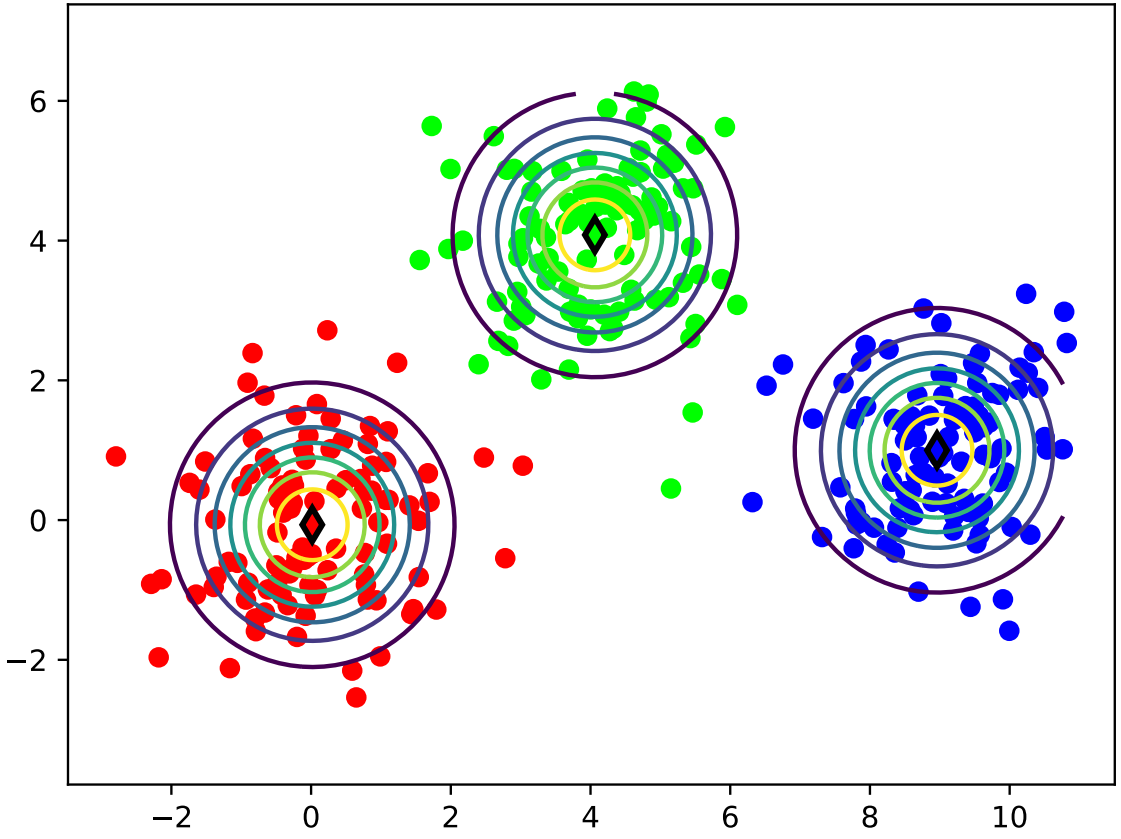
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign...



Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter...



Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign... Recenter... Reassign... Recenter...
Reassign... Recenter... Reassign...



# Lloyd's method revisited.

1. Choose initial clusters $(S_1, \ldots, S_k)$.
2. Repeat until convergence:

   2.1 **(Recenter.)** Set $\boldsymbol{\mu}_j := \text{mean}(S_j)$ for $j \in (1, \ldots, k)$.
   2.2 **(Reassign).** Update $S_j := \{\boldsymbol{x}_i : \boldsymbol{\mu}(\boldsymbol{x}_i) = \boldsymbol{\mu}_j\}$ for $j \in (1, \ldots, k)$.
   ("$\boldsymbol{\mu}(\boldsymbol{x}_i)$" means "center closest to $\boldsymbol{x}_i$"; break ties arbitrarily).

# Lloyd's method revisited.

1. Choose initial clusters $(S_1, \ldots, S_k)$.
2. Repeat until convergence:

   2.1 **(Recenter.)** Set $\boldsymbol{\mu}_j := \text{mean}(S_j)$ for $j \in (1, \ldots, k)$.
   2.2 **(Reassign).** Update $S_j := \{\boldsymbol{x}_i : \boldsymbol{\mu}(\boldsymbol{x}_i) = \boldsymbol{\mu}_j\}$ for $j \in (1, \ldots, k)$.
   ("$\boldsymbol{\mu}(\boldsymbol{x}_i)$" means "center closest to $\boldsymbol{x}_i$"; break ties arbitrarily).

**Geometric perspective:**

- Centers define a **Voronoi diagram/partition**:
  for each $\boldsymbol{\mu}_j$, define cell $V_j := \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{\mu}(\boldsymbol{x}) = \boldsymbol{\mu}_j\}$
  (break ties arbitrarily).
- Reassignment leaves assignment consistent with Voronoi cells.
- Recentering might shift data outside Voronoi cells,
  **except** if we've converged!
- See `http://mjt.cs.illinois.edu/htv/` for an interactive demo.

# Does Lloyd's method solve the original problem?

**Theorem.**

- For all $t$, $\phi(C_t; A_{t-1}) \geq \phi(C_t; A_t) \geq \phi(C_{t+1}; A_t)$.
- The method terminates.

# Does Lloyd's method solve the original problem?

**Theorem.**

- For all $t$, $\phi(C_t; A_{t-1}) \geq \phi(C_t; A_t) \geq \phi(C_{t+1}; A_t)$.
- The method terminates.

**Proof.**

- This first property is from the earlier theorem and the definition of the algorithm:

$$\phi(C_t; A_t) = \phi(C_t; \mathcal{A}(C_{t-1})) = \min_{\boldsymbol{A} \in \mathcal{A}} \phi(C_t; \boldsymbol{A}) \leq \phi(C_t, A_{t-1}),$$
$$\phi(C_{t+1}; A_t) = \phi(\mathcal{C}(A_t); A_t) = \min_{\boldsymbol{C} \in \mathcal{C}} \phi(\boldsymbol{C}; A_t) \leq \phi(C_t, A_t),$$

- Previous property implies: cost is nonincreasing.
  Combined with termination condition:
  all but final partition visited at most once.
  There are finitely many partitions of $(\boldsymbol{x}_i)_{i=1}^n$.

$\square$

# Does Lloyd's method solve the original problem?

**Theorem.**

- For all $t$, $\phi(C_t; A_{t-1}) \geq \phi(C_t; A_t) \geq \phi(C_{t+1}; A_t)$.
- The method terminates.

**Proof.**

- This first property is from the earlier theorem and the definition of the algorithm:

$$\phi(C_t; A_t) = \phi(C_t; \mathcal{A}(C_{t-1})) = \min_{\boldsymbol{A} \in \mathcal{A}} \phi(C_t; \boldsymbol{A}) \leq \phi(C_t, A_{t-1}),$$
$$\phi(C_{t+1}; A_t) = \phi(\mathcal{C}(A_t); A_t) = \min_{\boldsymbol{C} \in \mathcal{C}} \phi(\boldsymbol{C}; A_t) \leq \phi(C_t, A_t),$$

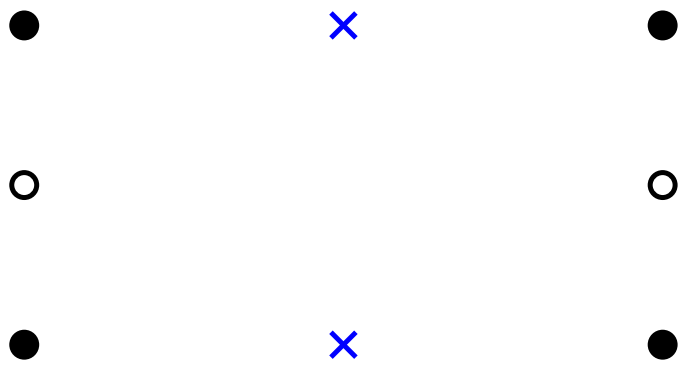
- Previous property implies: cost is nonincreasing.
  Combined with termination condition:
  all but final partition visited at most once.
  There are finitely many partitions of $(\boldsymbol{x}_i)_{i=1}^n$.

□

**(That didn't answer the question. . . )**

# Seriously: does Lloyd's method solve the original problem?

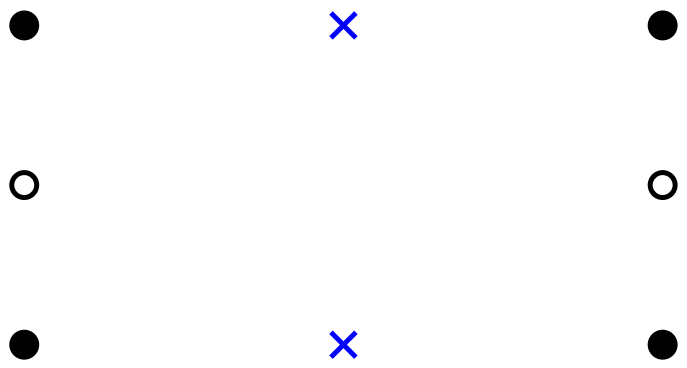
- In practice, Lloyd's method seems to optimize well;
  In theory, output can have **unboundedly poor cost**.



(Suppose width is $c > 1$ and height is 1.)

# Seriously: does Lloyd's method solve the original problem?

- In practice, Lloyd's method seems to optimize well;
  In theory, output can have **unboundedly poor cost**.

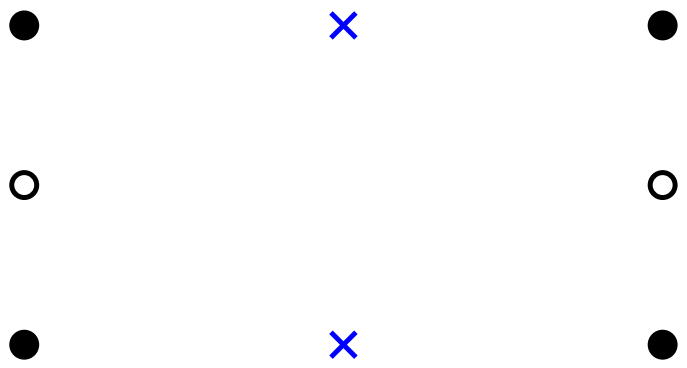


  (Suppose width is $c > 1$ and height is 1.)

- In practice, method takes few iterations;
  in theory: can take $2^{\Omega(\sqrt{n})}$ iterations!
  (Examples of this are painful; but note, problem is NP-hard, and convergence proof used number of partitions. . . )

# Seriously: does Lloyd's method solve the original problem?

- In practice, Lloyd's method seems to optimize well;
  In theory, output can have **unboundedly poor cost**.



  (Suppose width is $c > 1$ and height is 1.)
- In practice, method takes few iterations;
  in theory: can take $2^{\Omega(\sqrt{n})}$ iterations!
  (Examples of this are painful; but note, problem is NP-hard, and convergence proof used number of partitions. . . )

**So: in practice, yes; in theory, don't know. . .**

# 3. Standard application: vector quantization

# Application: vector quantization.

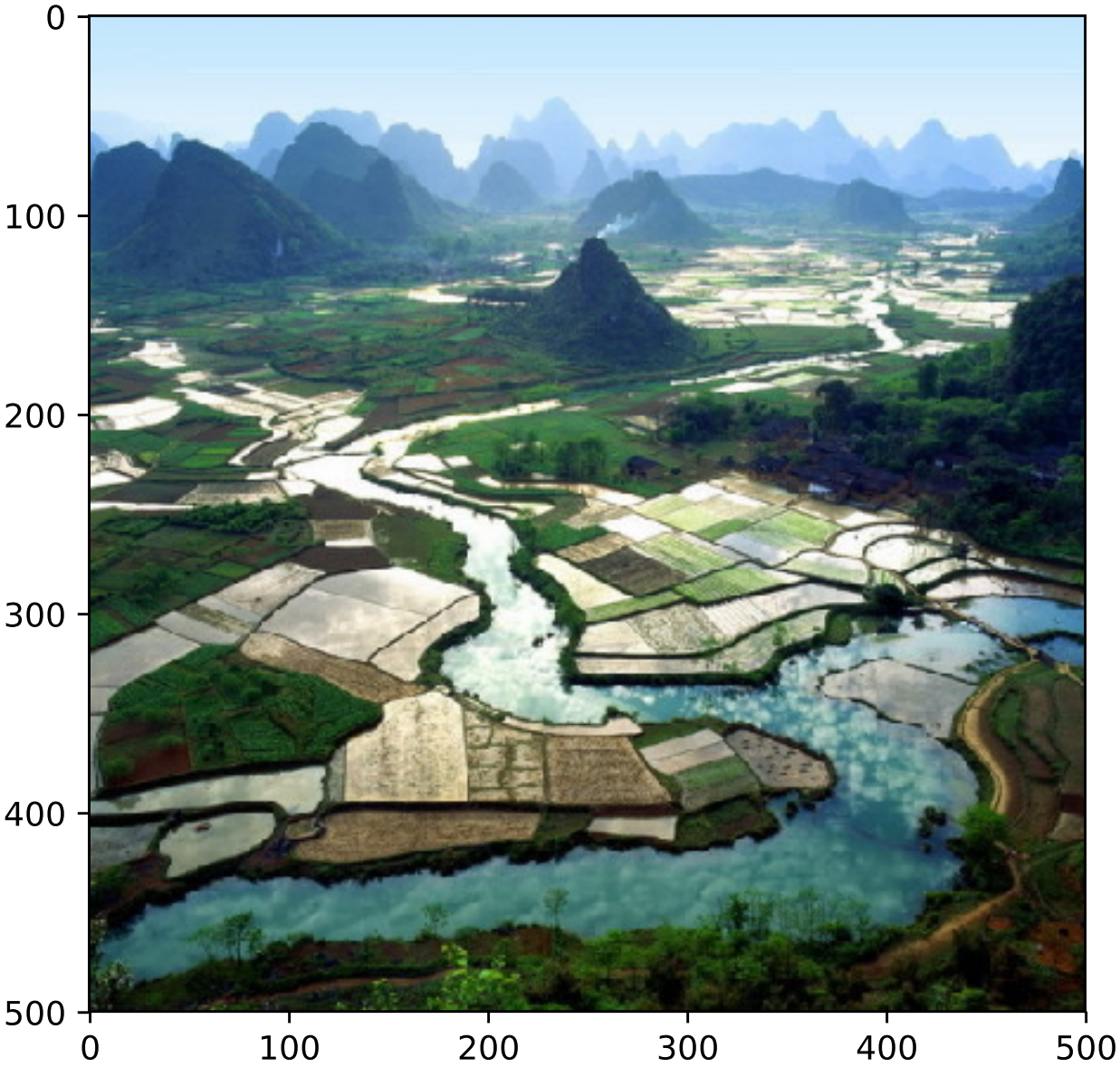
**Vector quantization** with $k$-means.

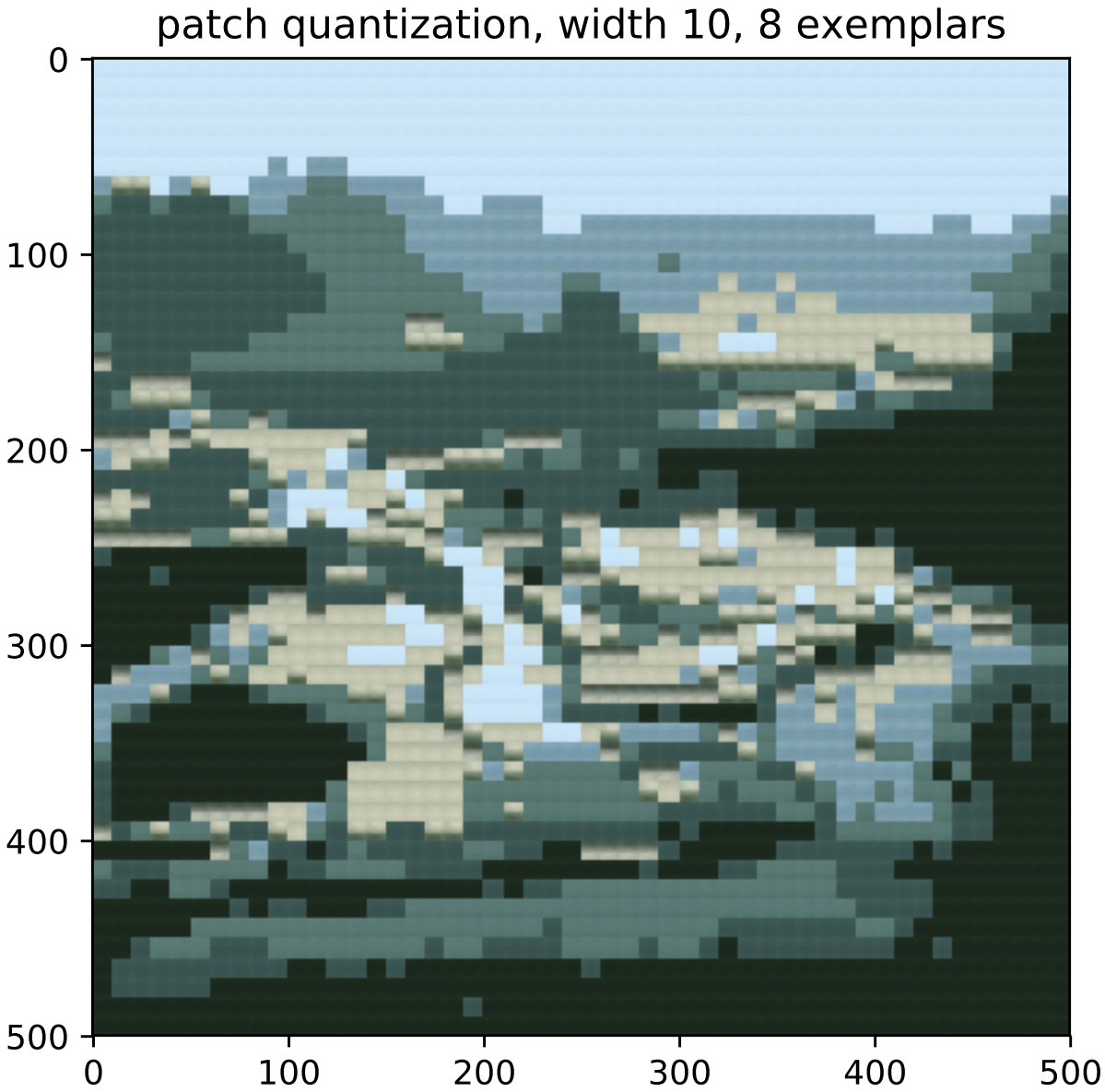
- Let $(\boldsymbol{x}_i)_{i=1}^n$ be given.
- run $k$-means to obtain $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k)$.
- Replace each $(\boldsymbol{x}_i)_{i=1}^n$ with $(\boldsymbol{\mu}(\boldsymbol{x}_i))_{i=1}^n$.

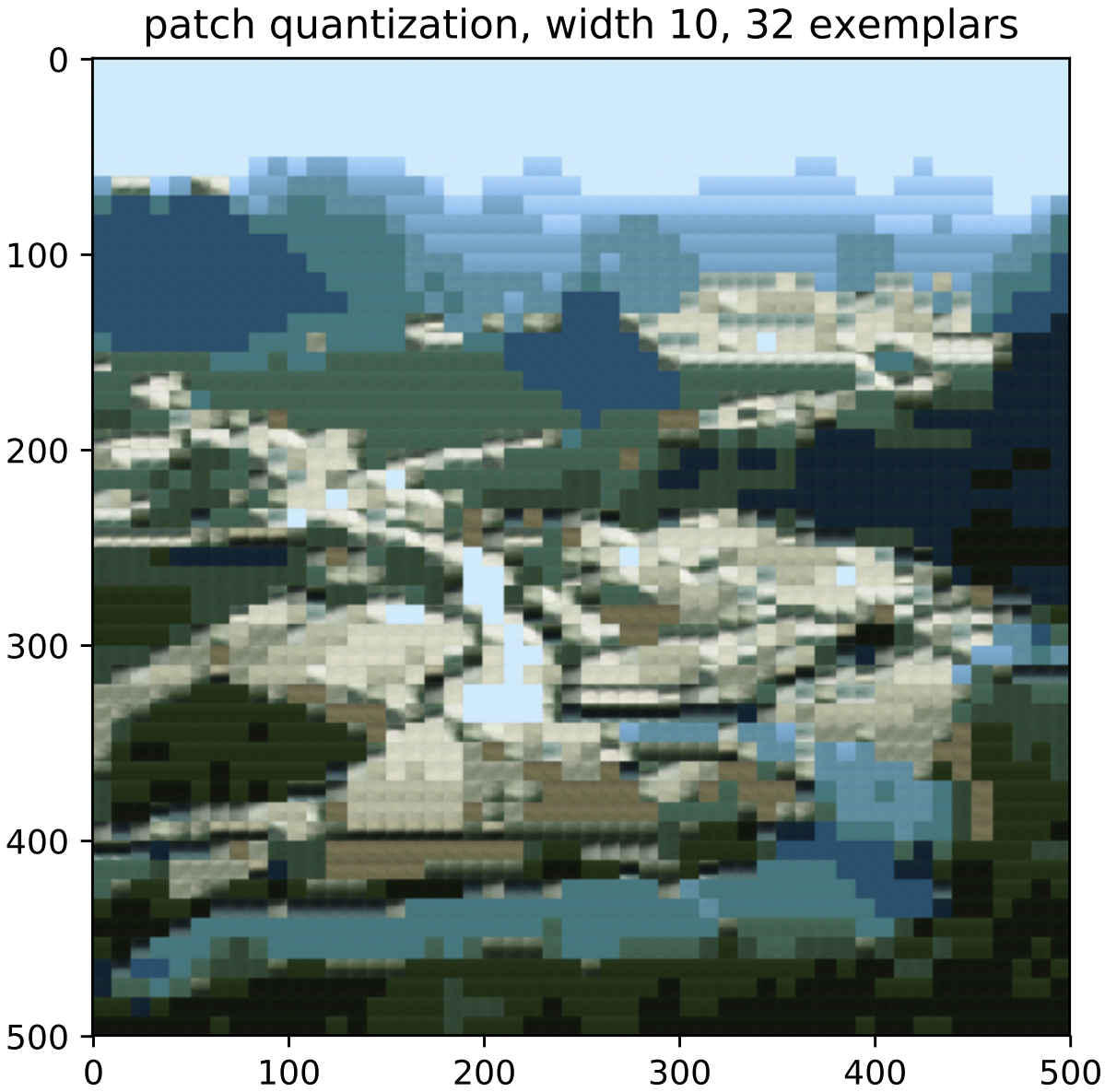
Encoding size reduces from $\mathcal{O}(nd)$ to $\mathcal{O}(kd + n\ln(k))$.

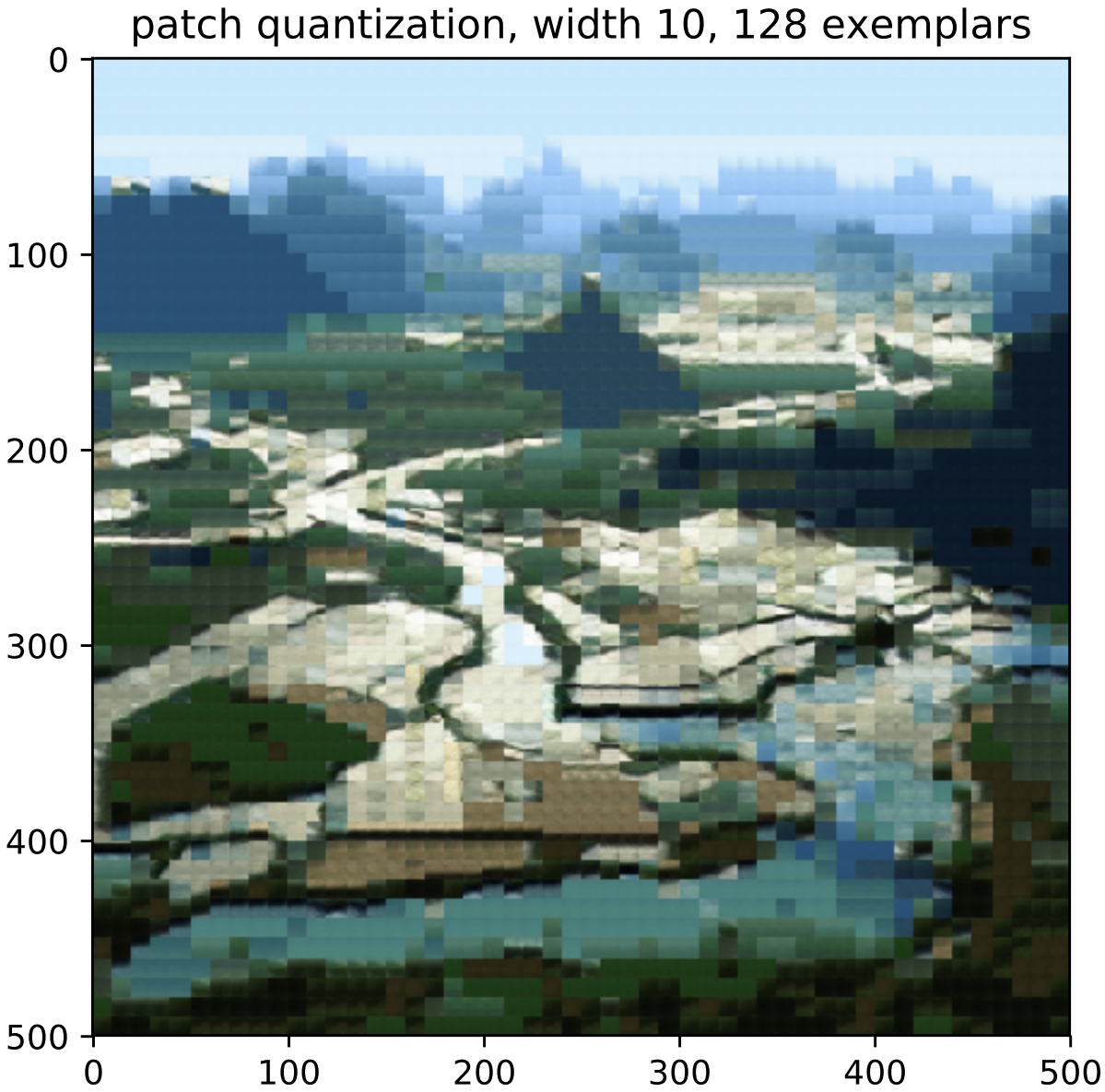
**Examples.**

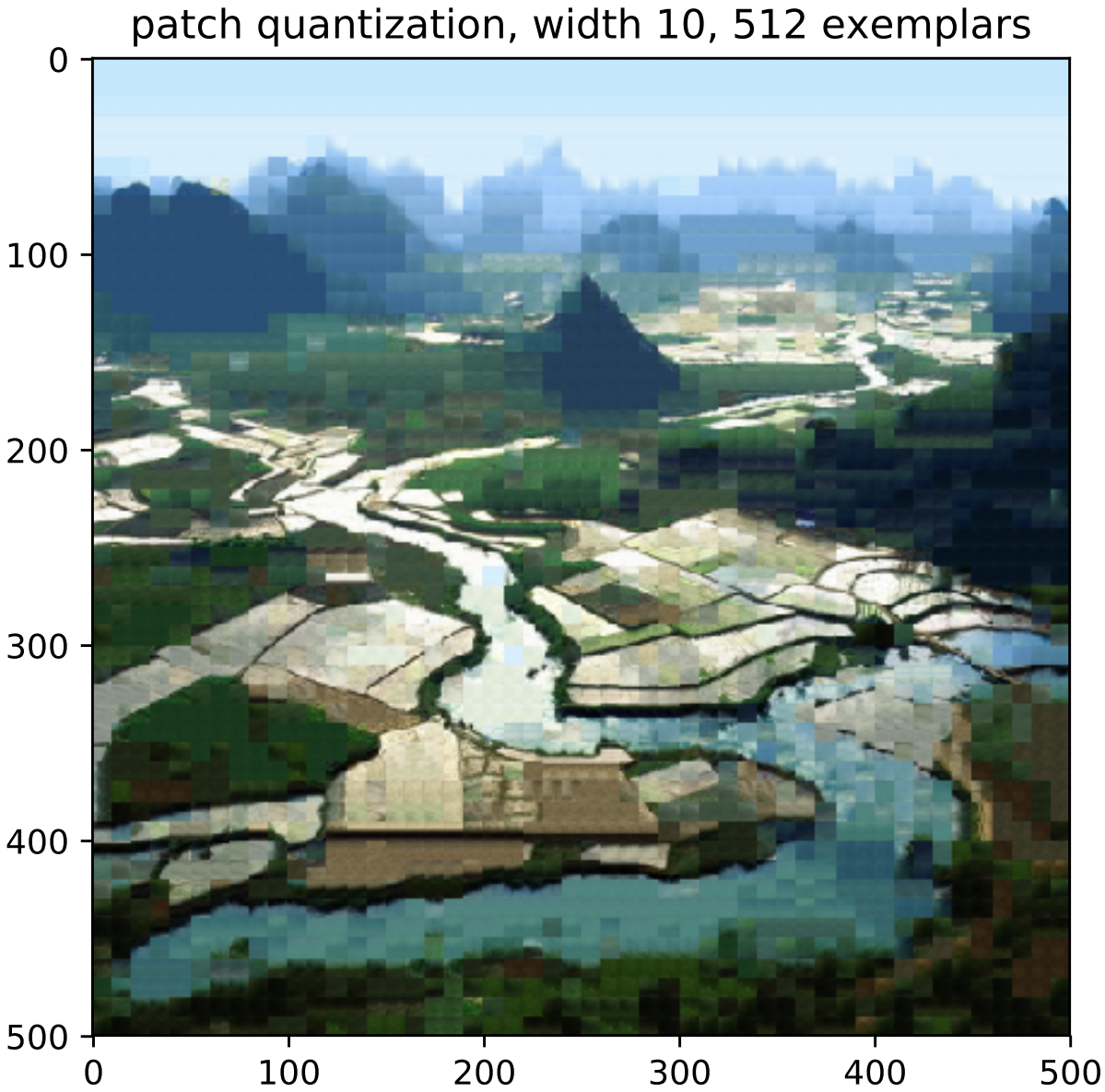
- Audio compression.
- Image compression.

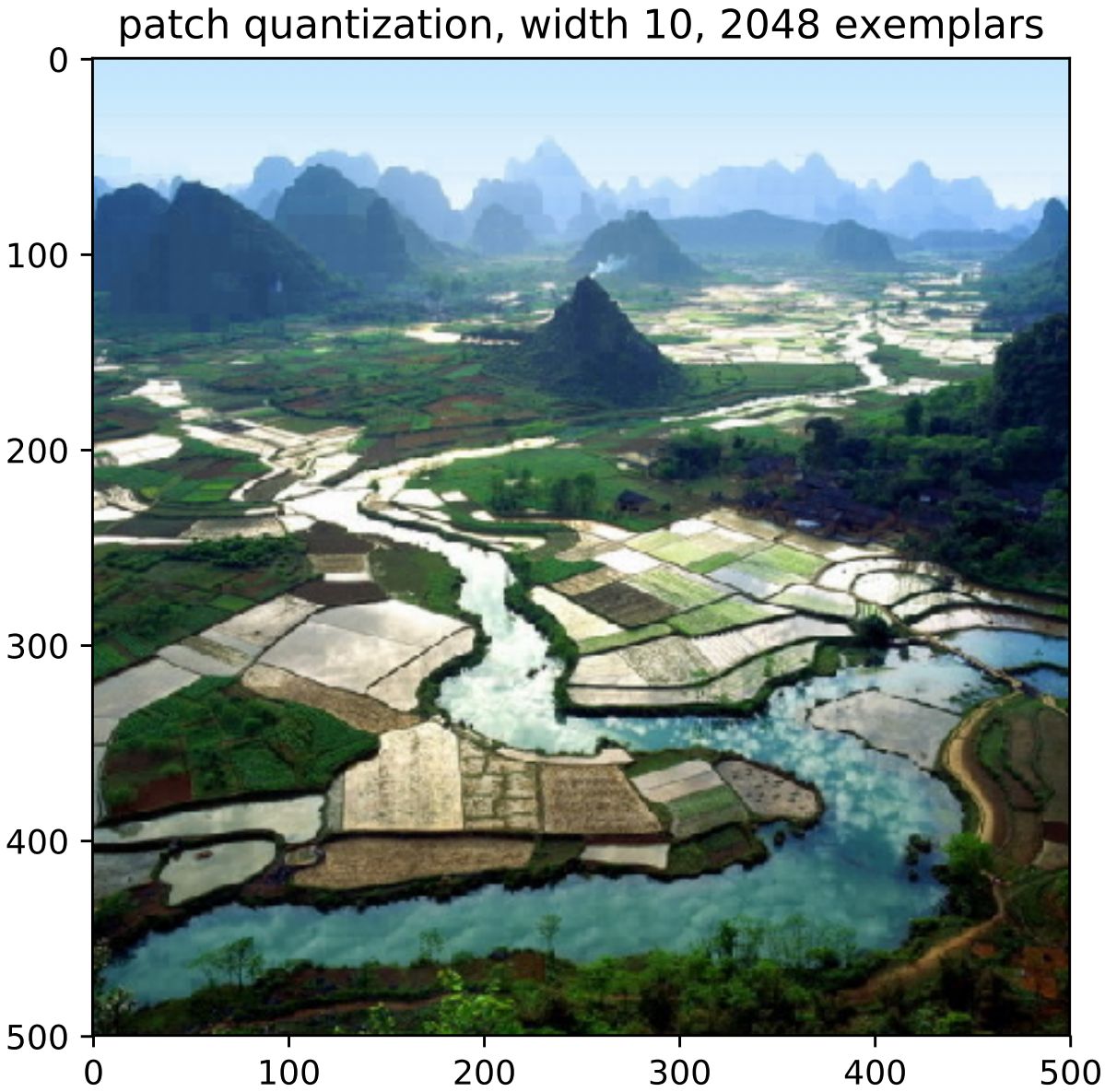

0
100
200
300
400
500
0
100
200
300
400
500

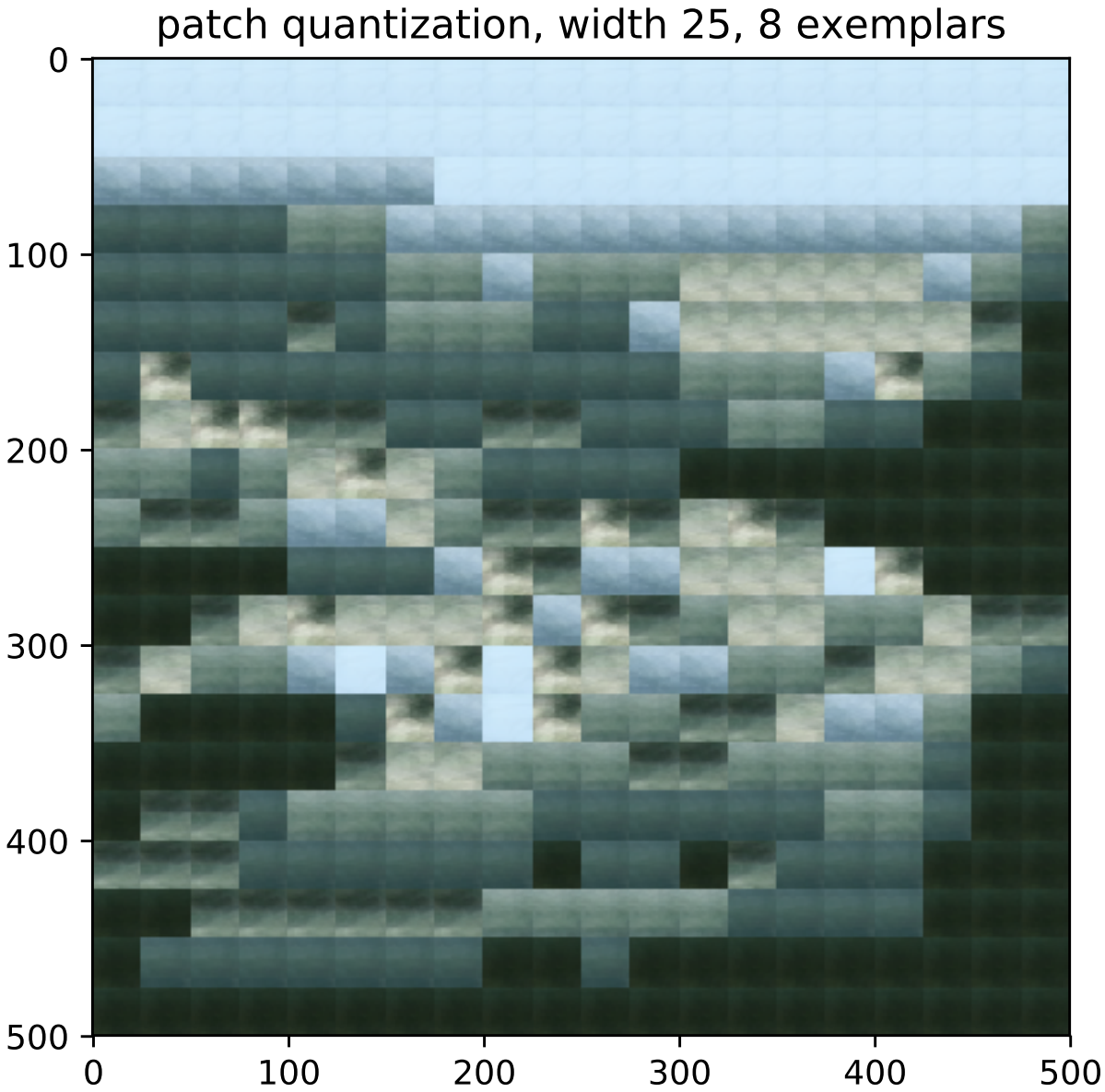

patch quantization, width 10, 8 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

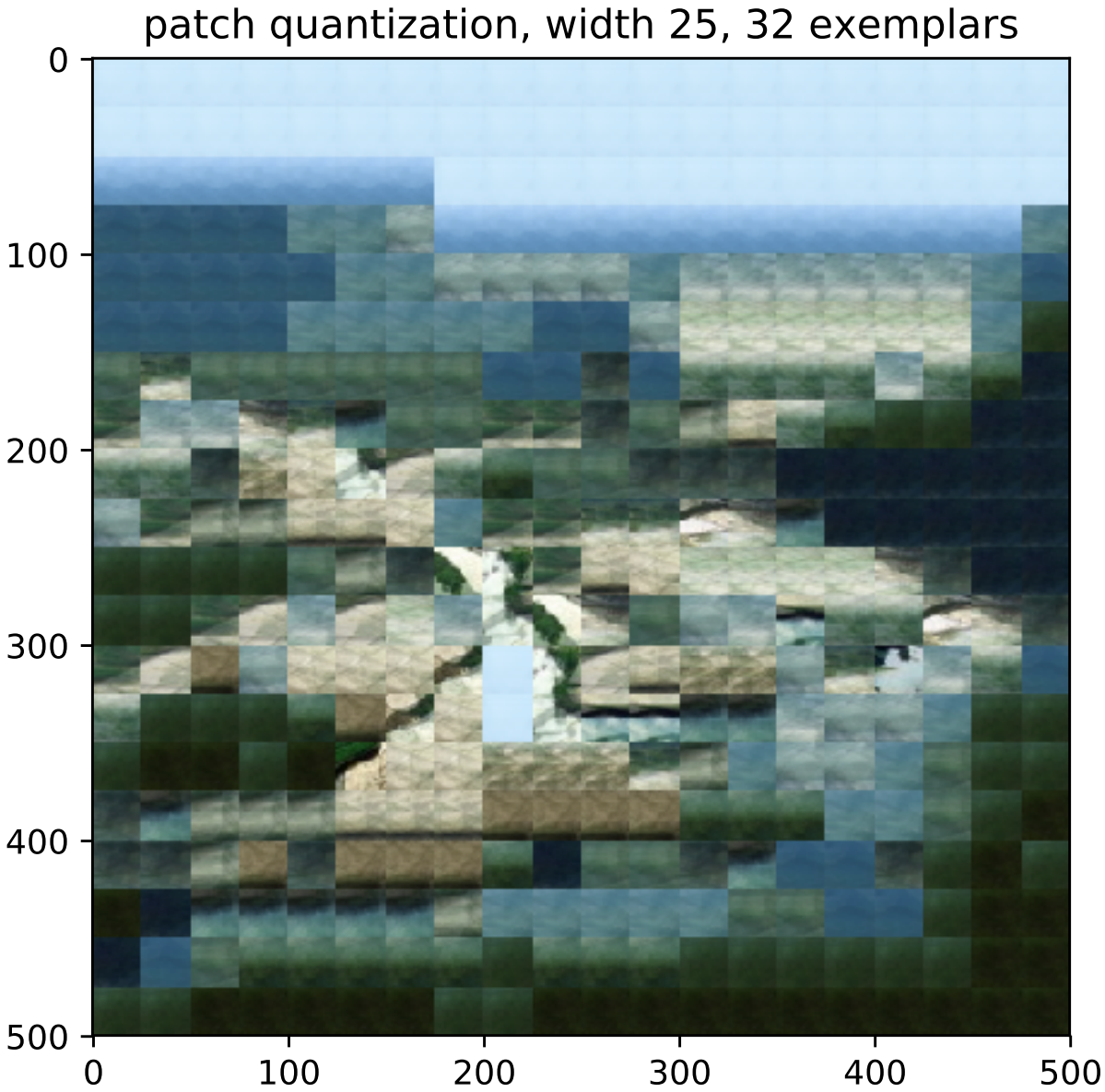

patch quantization, width 10, 32 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

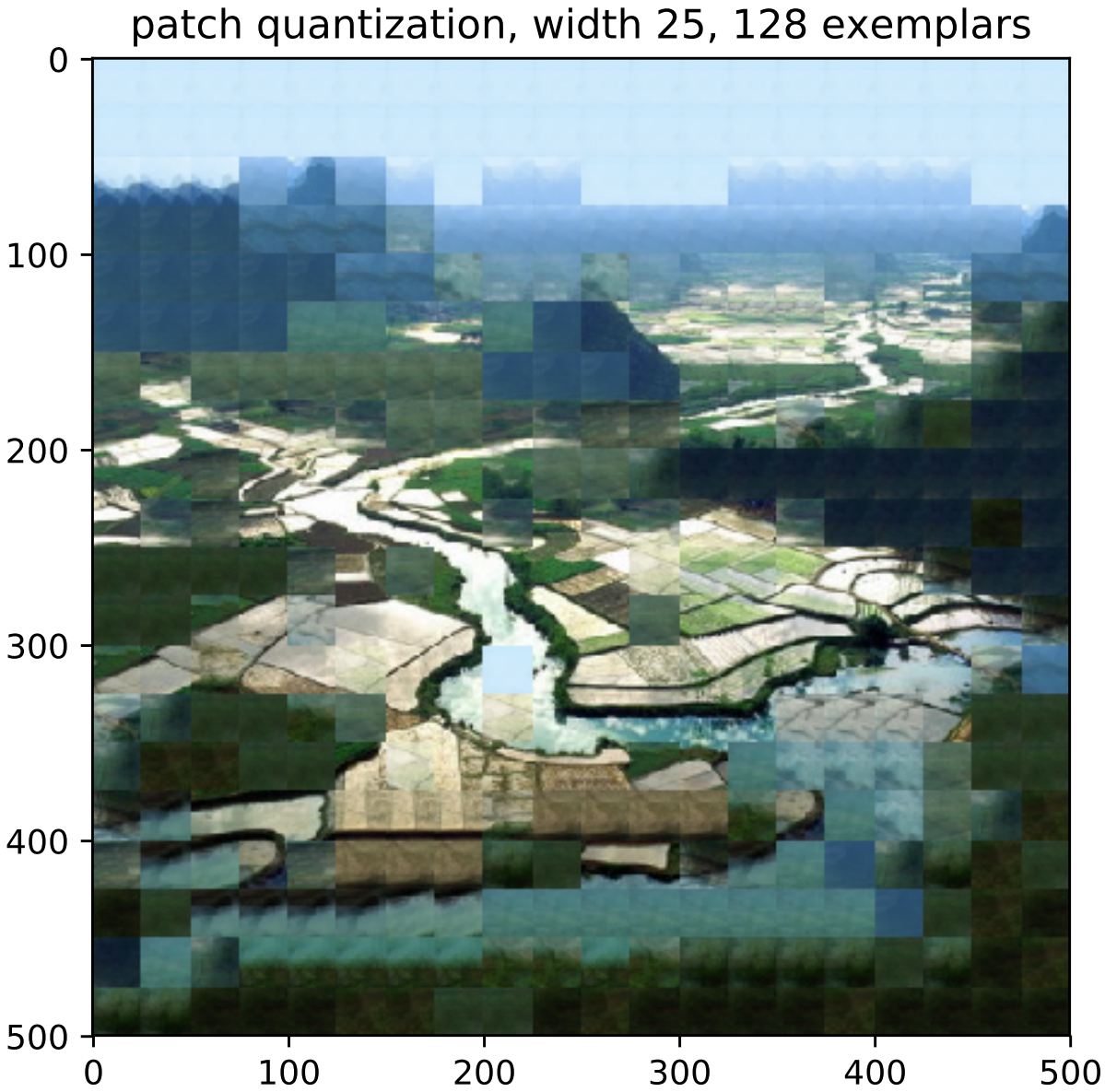

patch quantization, width 10, 128 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

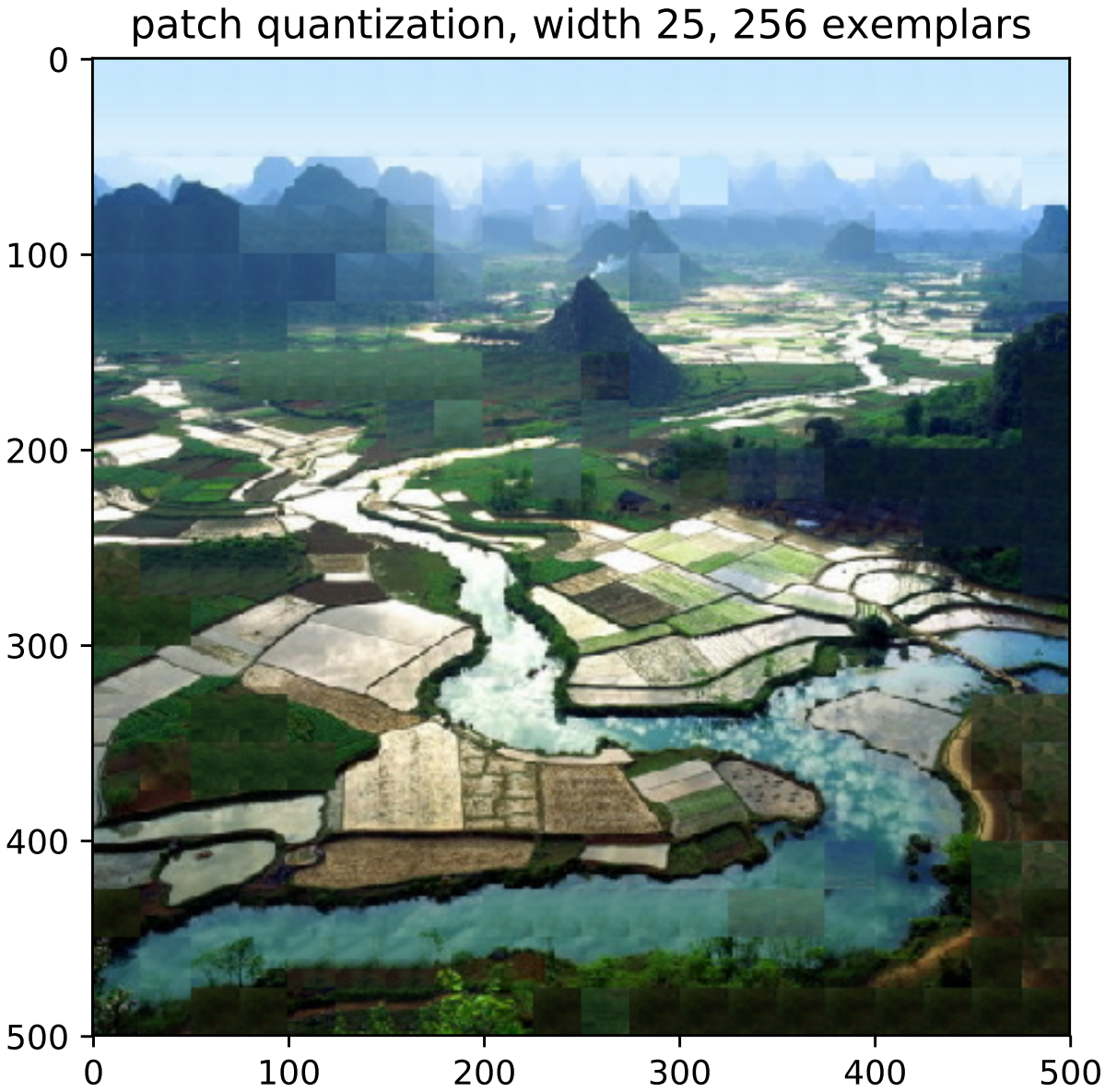

patch quantization, width 10, 512 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

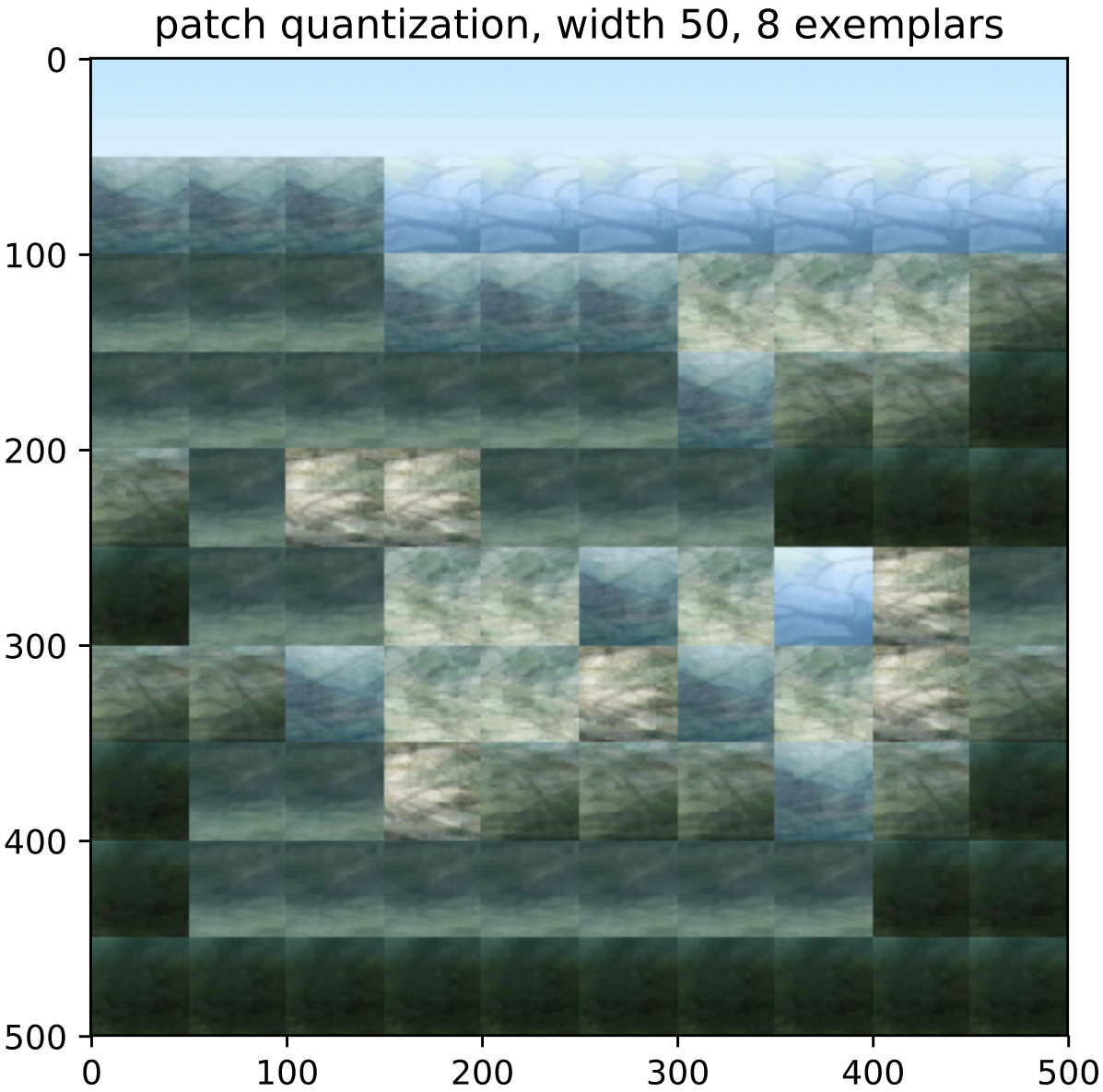

patch quantization, width 10, 2048 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

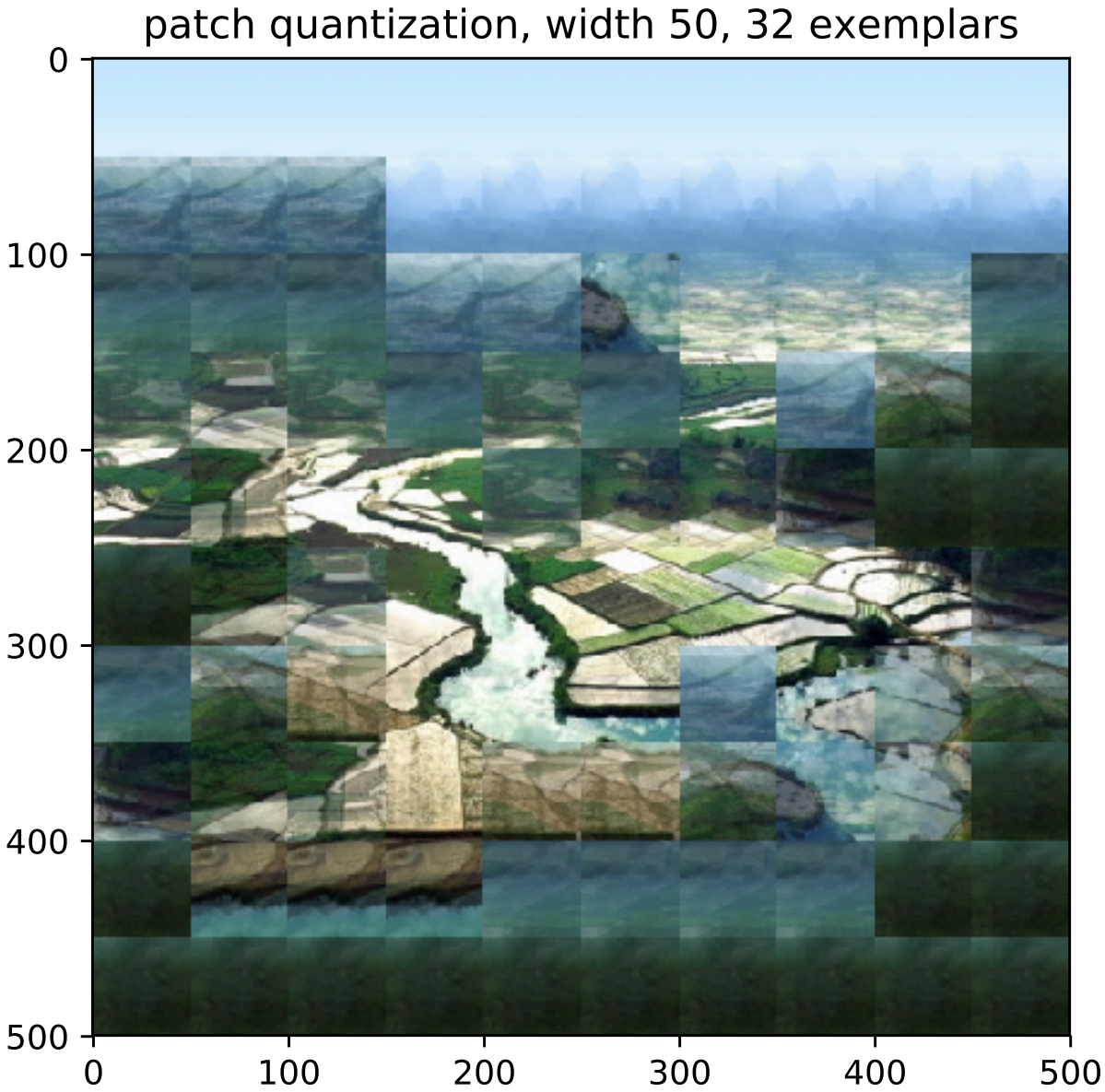

patch quantization, width 25, 8 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

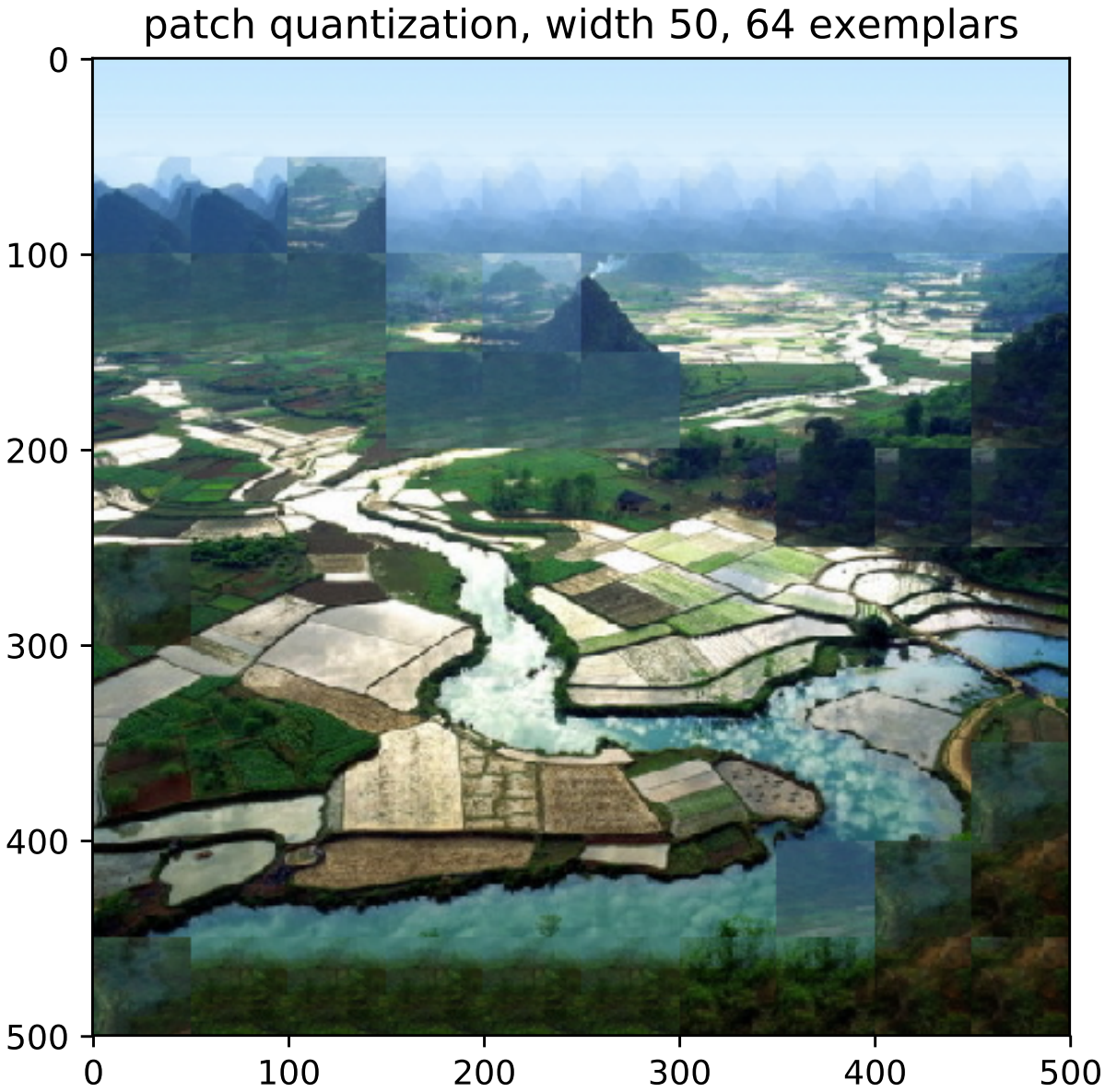

patch quantization, width 25, 32 exemplars
0
100
200
300
400
500
0
100
200
300
400
500


patch quantization, width 25, 128 exemplars
0
100
200
300
400
500
0
100
200
300
400
500


patch quantization, width 25, 256 exemplars
0
100
200
300
400
500
0
100
200
300
400
500


patch quantization, width 50, 8 exemplars
0
100
200
300
400
500
0
100
200
300
400
500


patch quantization, width 50, 32 exemplars
0
100
200
300
400
500
0
100
200
300
400
500


patch quantization, width 50, 64 exemplars
0
100
200
300
400
500
0
100
200
300
400
500

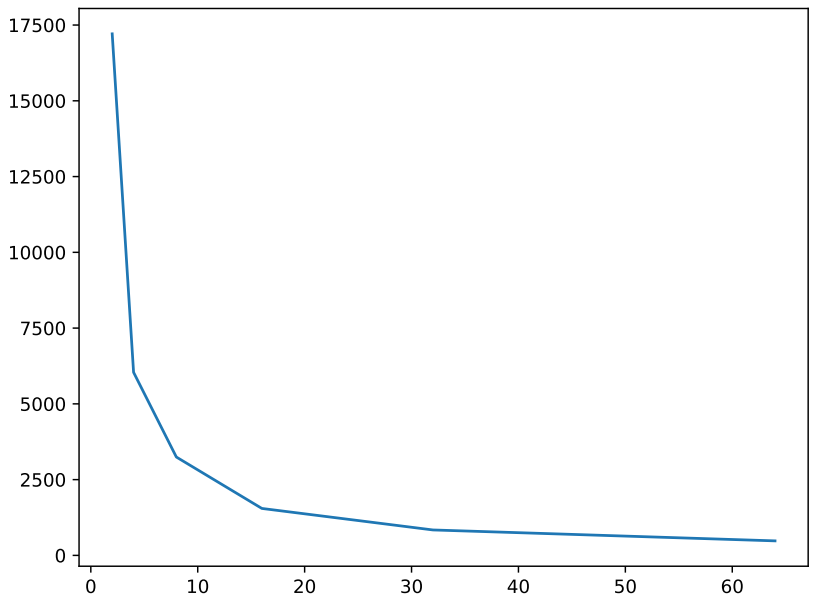
# 4. Other topics

# Initialization matters!

- **Easy choices:**
  - $k$ random points from dataset.
  - Random partition.
- **Standard choice** (theory and practice): "$D^2$-sampling"/`kmeans++`
  1. Choose $\mu_1$ uniformly at random from data.
  2. for $j \in (2, \ldots, k)$:
     2.1 Choose $x_i \propto \min_{l<j} \|x_i - \mu_l\|_2^2$.
- `kmeans++` is *randomized furthest-first traversal*; regular furthest-first fails with outliers.
- Scikits-learn and Matlab both default to `kmeans++`.

# How to choose $k$?

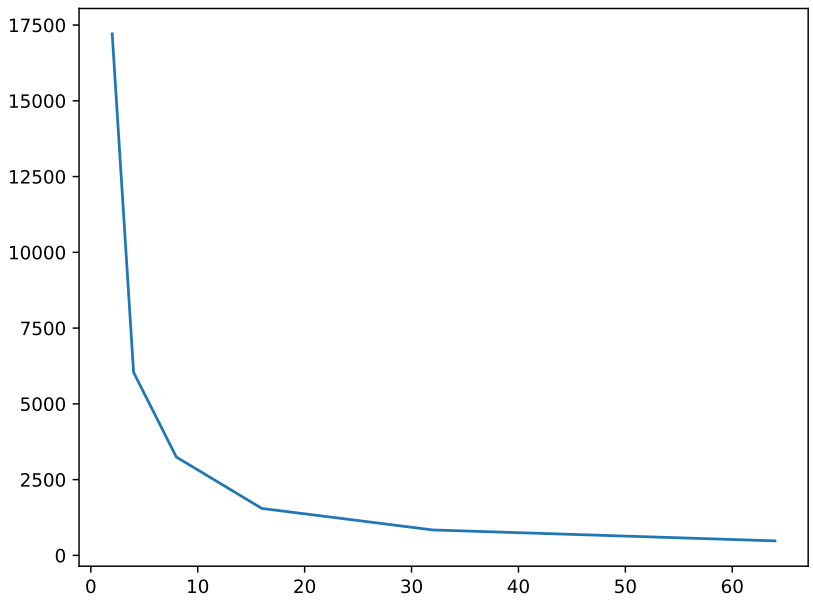
Costs when quantizing pixels of *Guilin*.




Reasonable to choose $k$ at "elbow";
trades off accuracy and complexity.

# How to choose $k$?

Costs when quantizing pixels of *Guilin*.




Reasonable to choose $k$ at "elbow";
trades off accuracy and complexity.

There are other, more complicated ways.
(E.g., "bayes information criterion", "Akaike information criterion", ...)

# Choice of $k$; sometimes no good choice.

Which of $k \in \{2, 3\}$ better on following data?

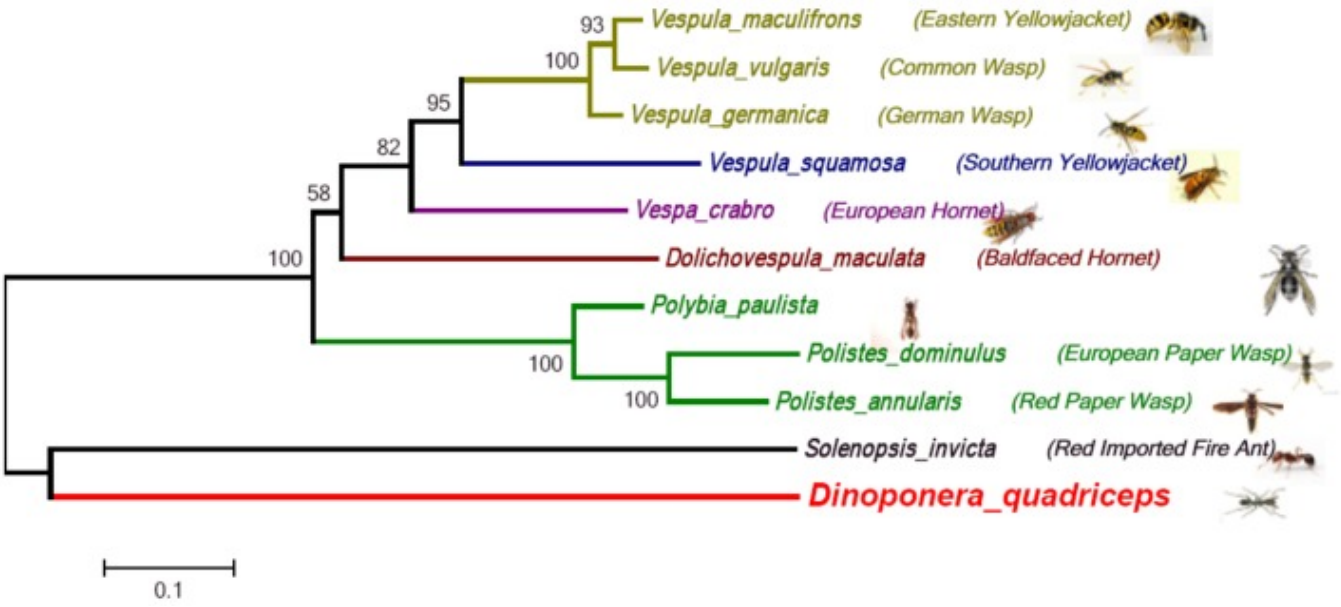
# Choice of $k$; sometimes no good choice.

Which of $k \in \{2, 3\}$ better on following data?

Perhaps **neither**; want 6. ("Elbow method" works here.)

# Choice of $k$; hierarchical clustering.

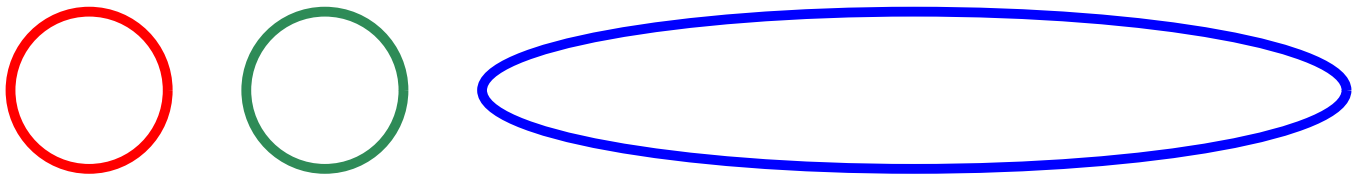
Sometimes *multiple* choices of $k$ make sense.
E.g., Phylogenetic trees have multiple notions of scales.




(Image credit: `https://www.researchgate.net/figure/Phylogenetic-tree-based-on-neighbor-joining-analyses-of-a-concatenated fig8_260108945`.)

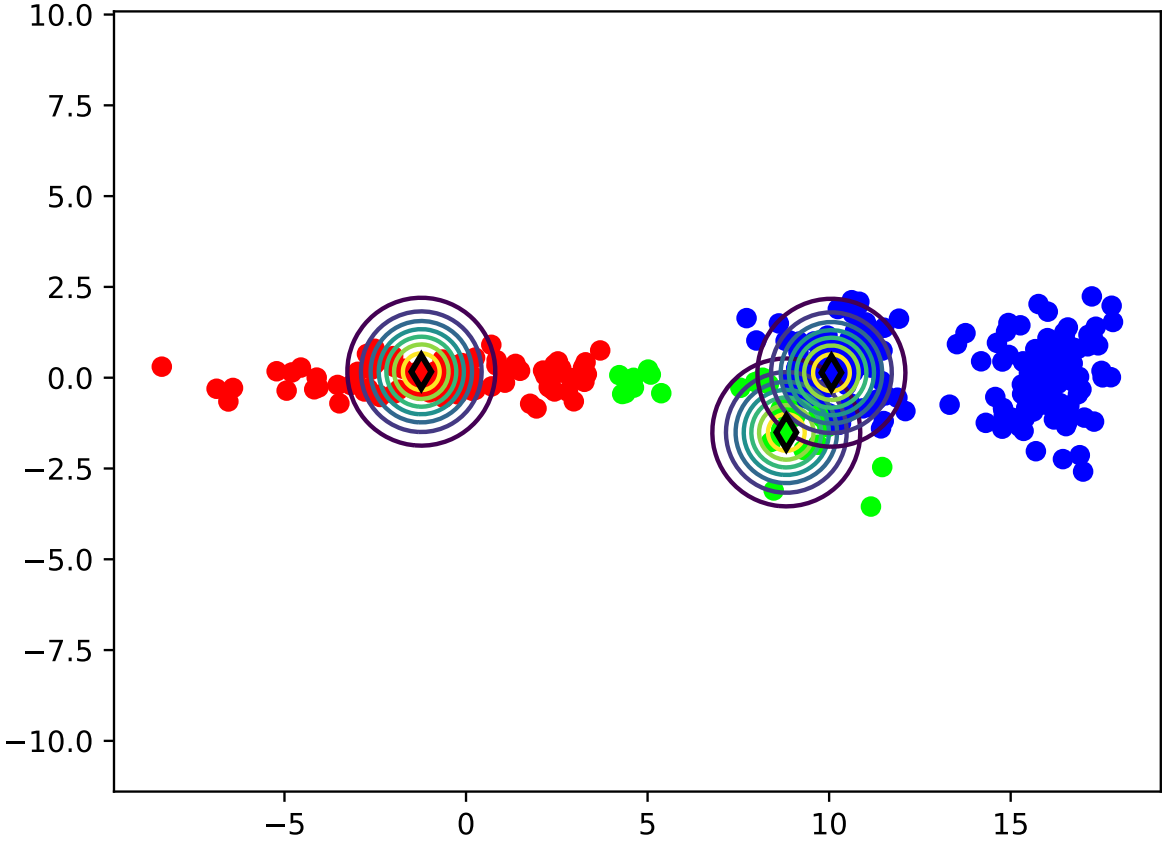
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters...



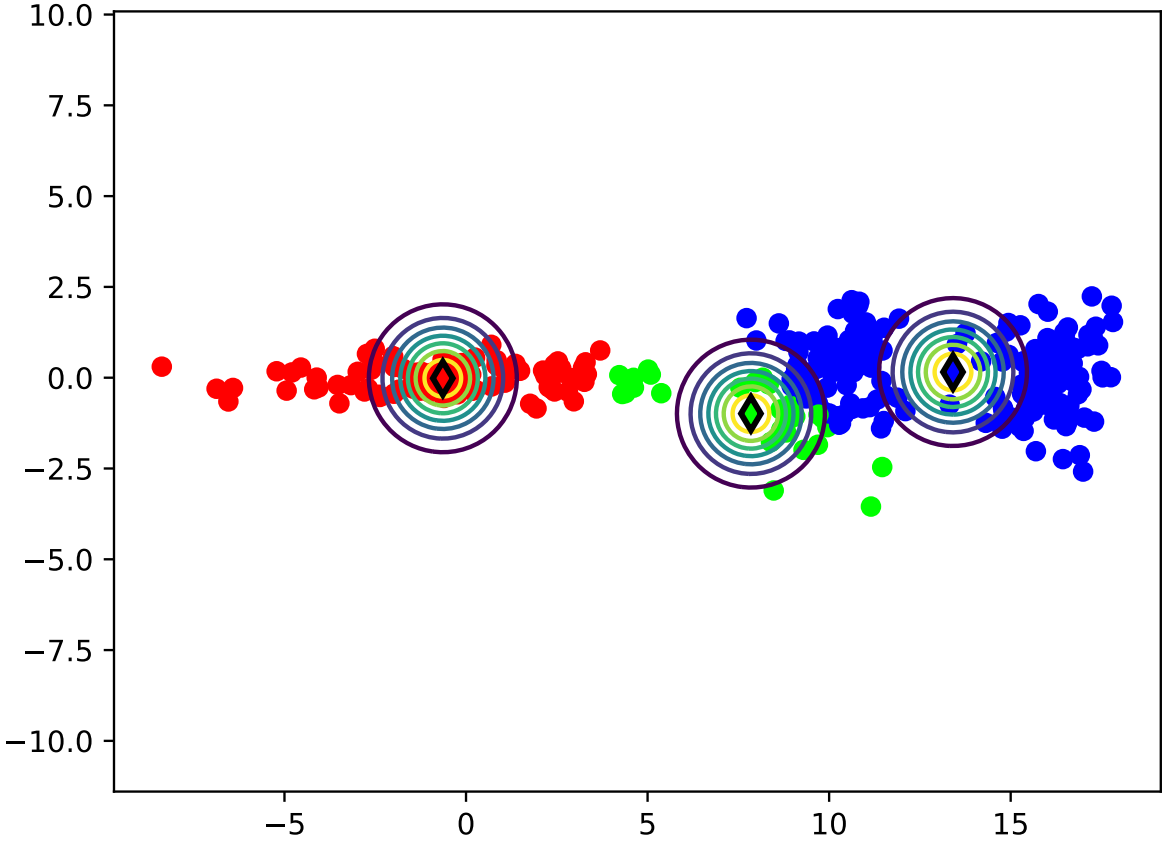
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



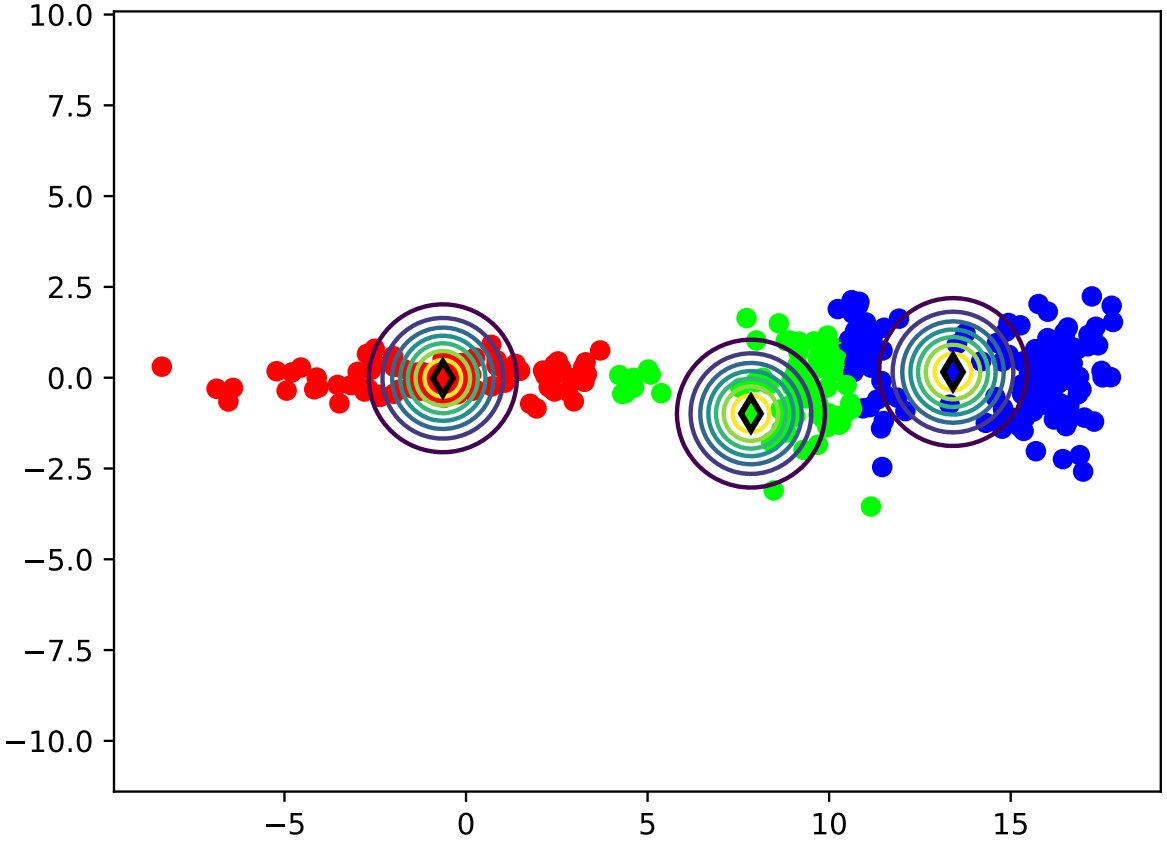
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



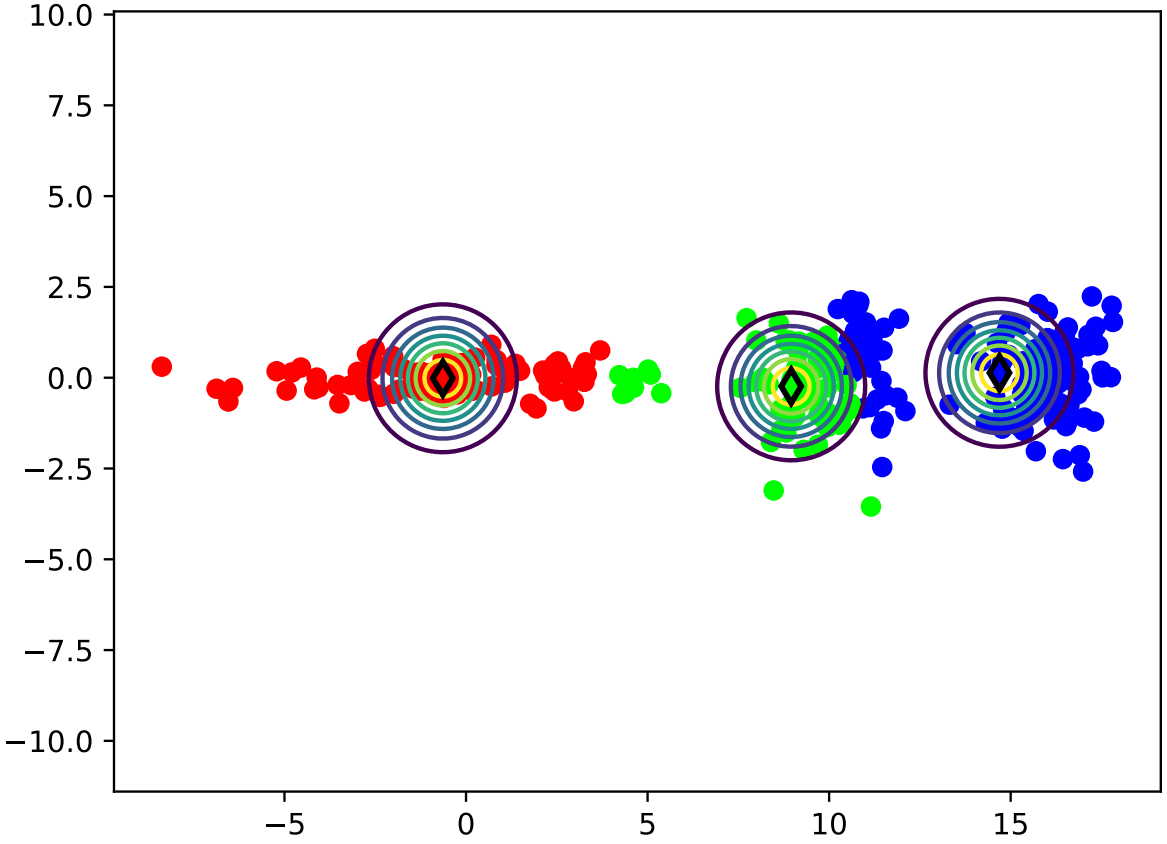
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



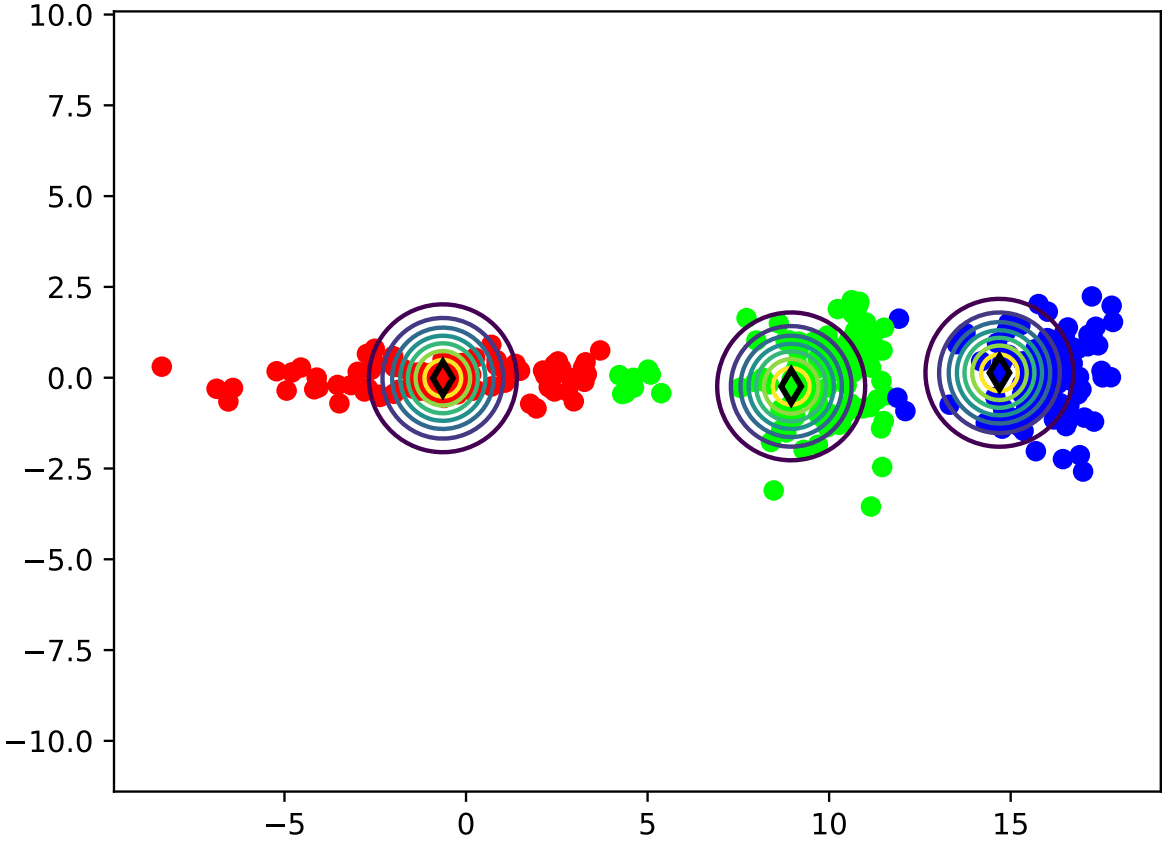
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



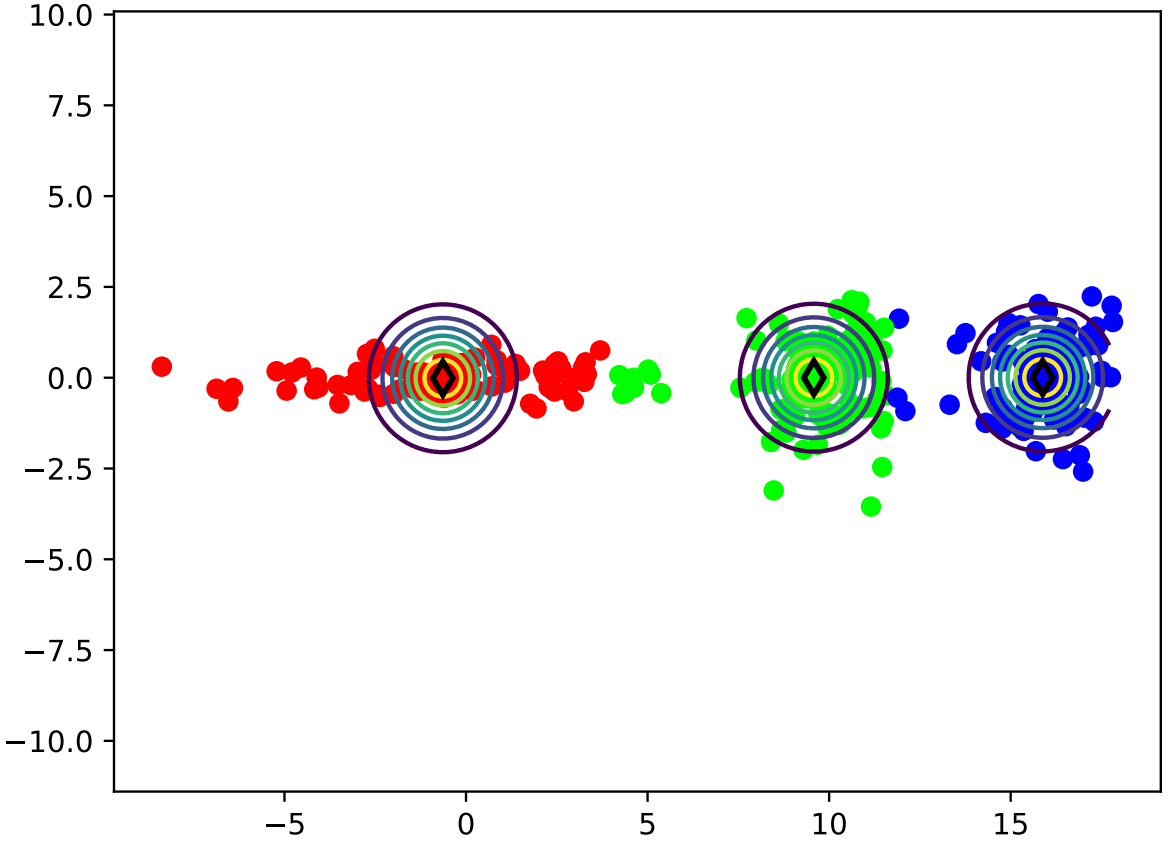
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



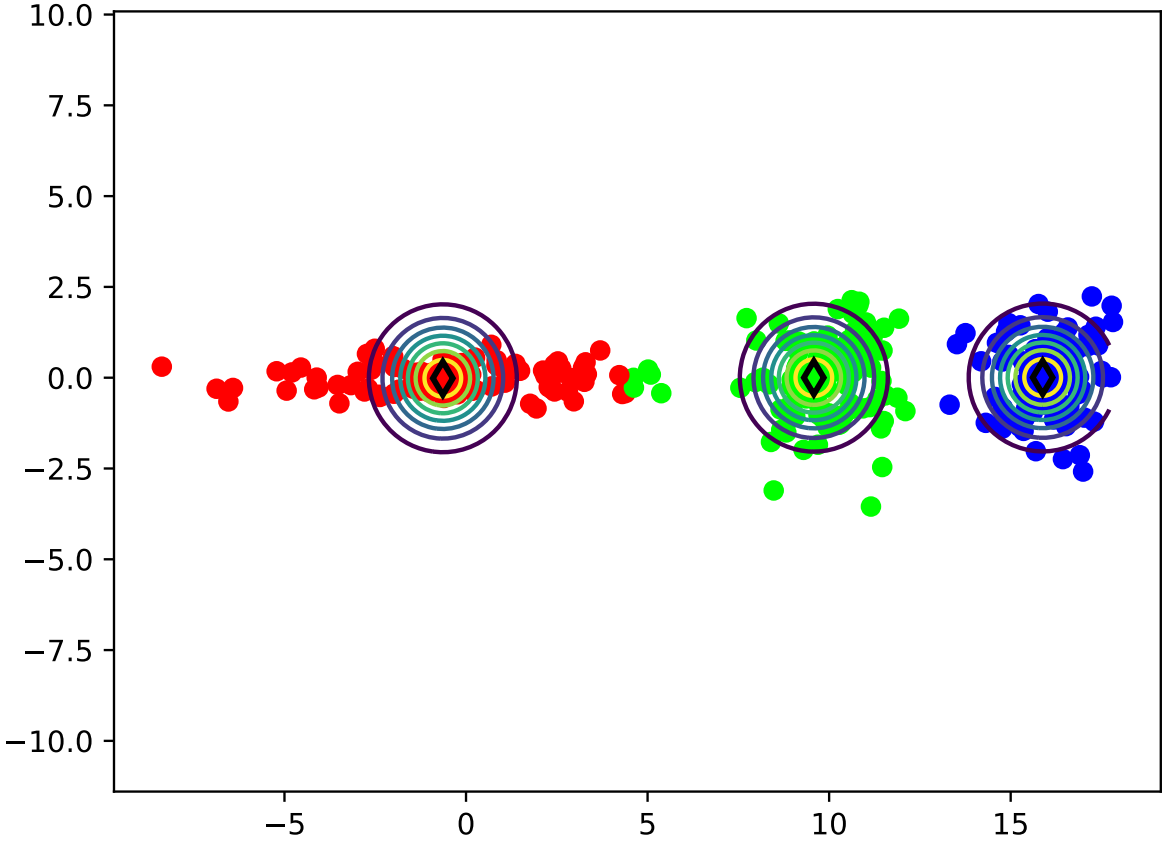
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



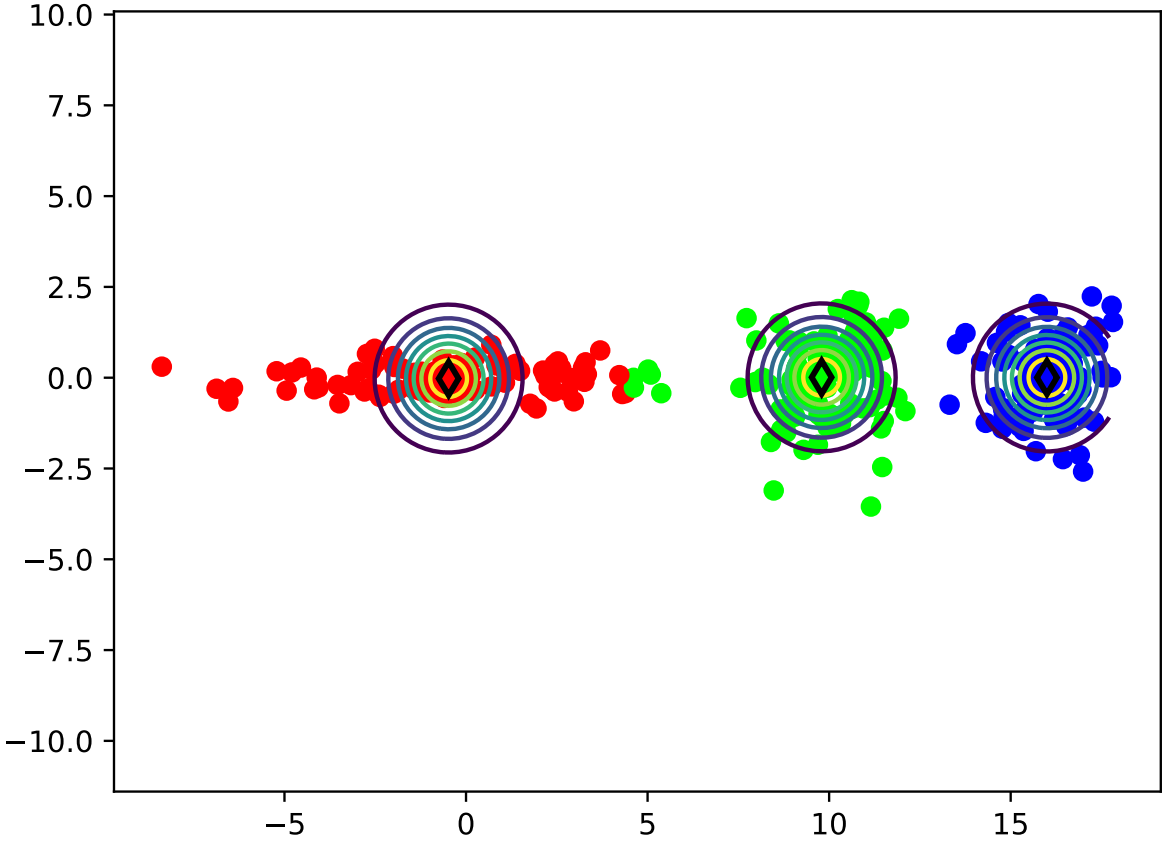
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



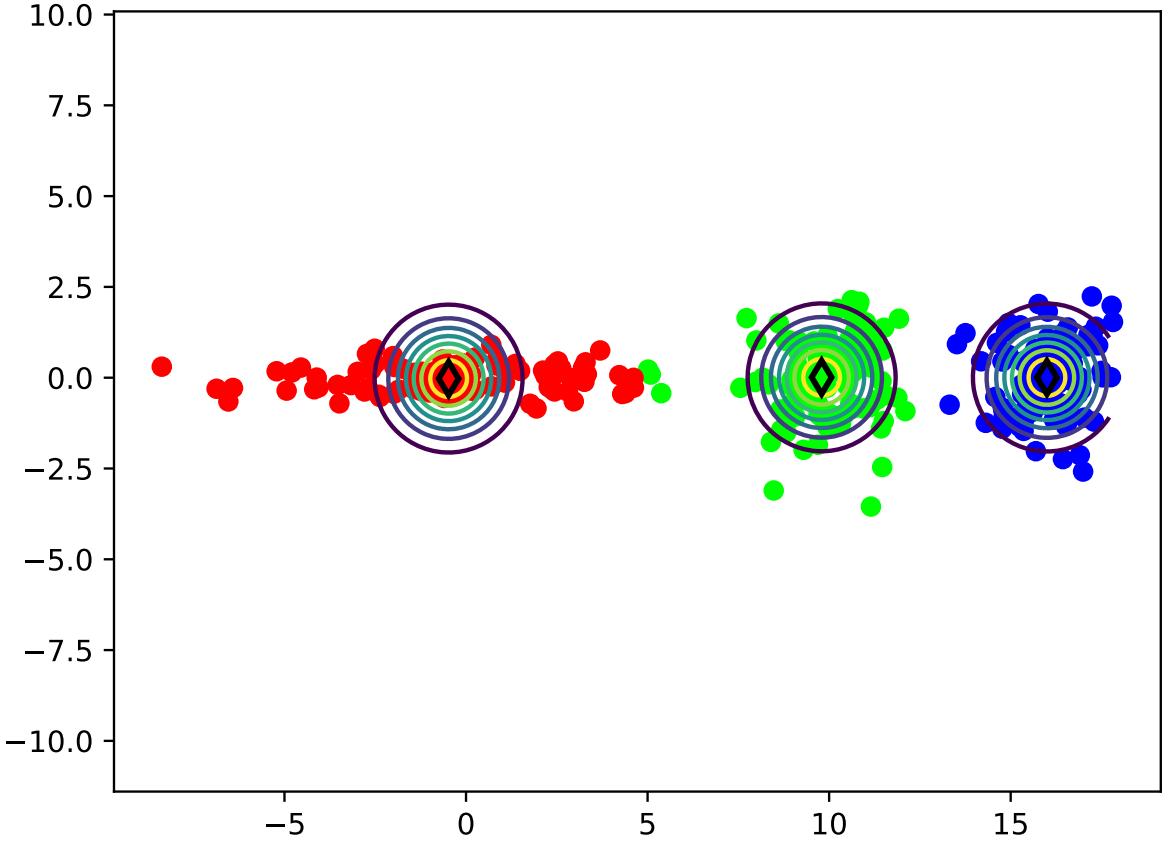
# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters. . .



# Non-spherical clusters.

The default choice $\|\cdot\|_2$ induces spherical clusters...

We can change the (dis)similarity function (from $\|\cdot\|_2$),
but still: all clusters must have same shape.
Gaussian Mixture Model provides another approach.

# Writing $k$-means with matrices

$\boldsymbol{C} \in \mathcal{C}_{d,k}$ has rows $\boldsymbol{\mu}_1^\top, \ldots, \boldsymbol{\mu}_k^\top$.
$\boldsymbol{A} \in \mathcal{A}_{n,k}$ has rows encoding cluster assignments.
**Therefore:** $\boldsymbol{AC}$ is matrix of assigned centers!

$$\begin{aligned}
\phi(\boldsymbol{C}; \boldsymbol{A}) &= \sum_{i=1}^{n} \sum_{j=1}^{k} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2 \\
&= \sum_{i=1}^{n} \|\mathbf{e}_i^\top \boldsymbol{X} - \mathbf{e}_i^\top (\boldsymbol{AC})\|^2 \\
&= \|\boldsymbol{X} - \boldsymbol{AC}\|_{\mathsf{F}}^2.
\end{aligned}$$

# 5. Summary

# $k$-means summary

- The $k$-means objective function:

$$\phi_S(\boldsymbol{C}) = \sum_{\boldsymbol{x}_i \in S} \min_j \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2.$$

- The alternate form with explicit assignments:

$$\phi_S(\boldsymbol{C}; \boldsymbol{A}) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2.$$

- Lloyd's method, and the meaning (and optimality) of its two steps.
- The "vector quantization" application.