Clustering algorithms

2IMM20 - Foundations of data mining
TU Eindhoven, Quartile 3, 2017-2018

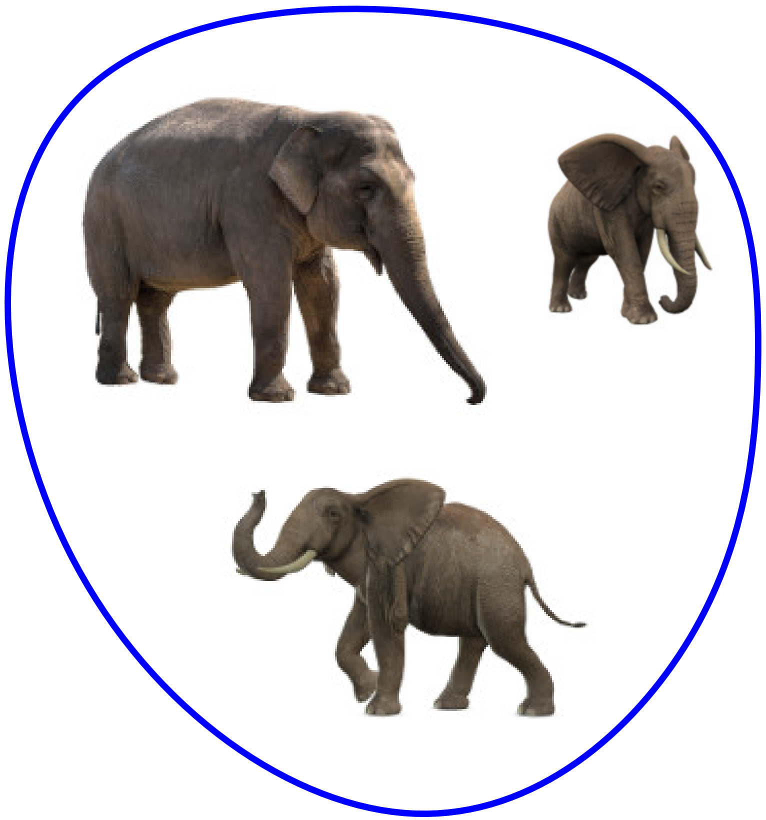
Anne Driemel

Overview of this lecture

- Clustering
- Facility Location
- Gonzales' algorithm
- Lloyd's algorithm (k-means)
- k-means++ algorithm
- Clustering in graphs

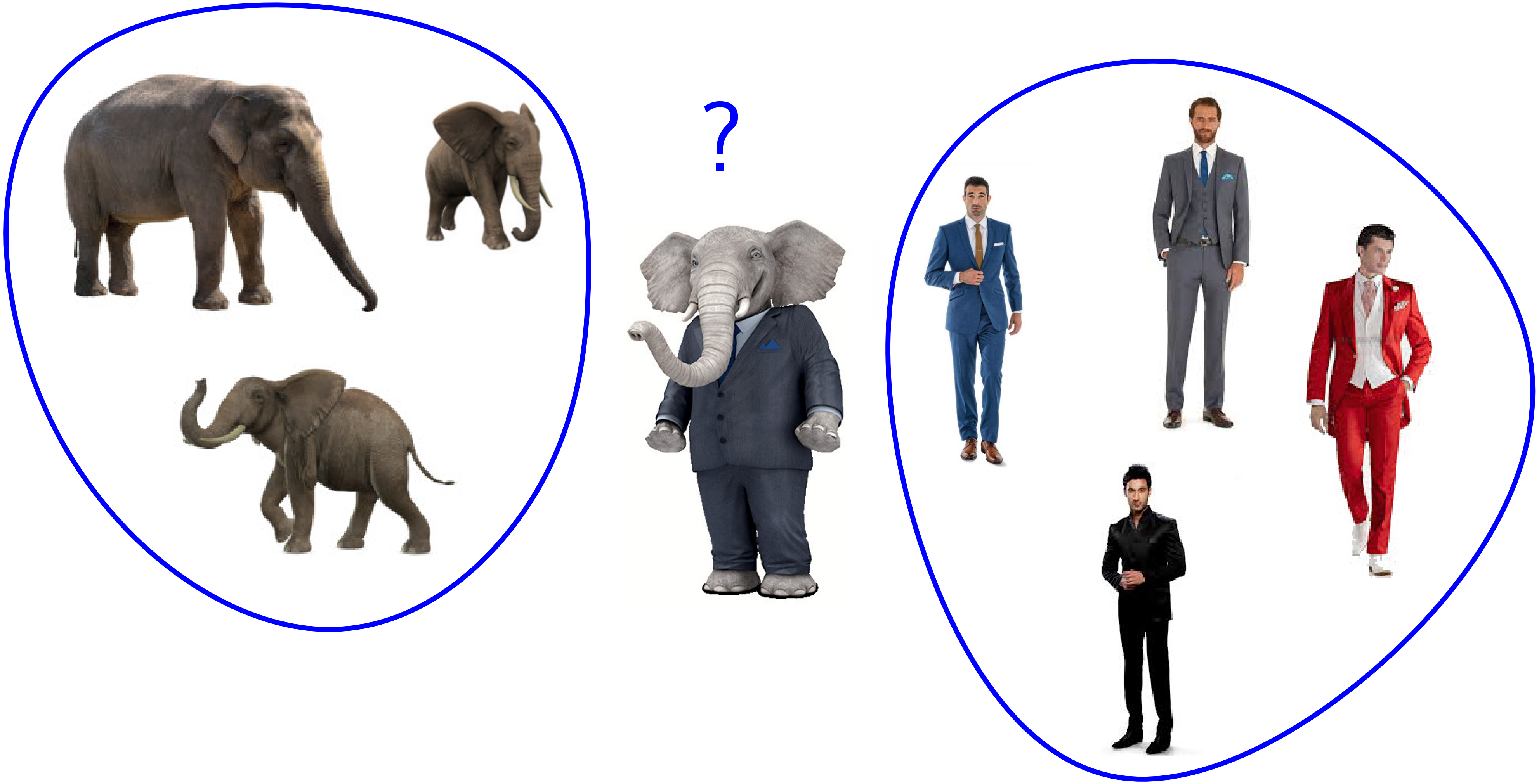
What is Clustering?

Clustering is the task of grouping similar objects into clusters



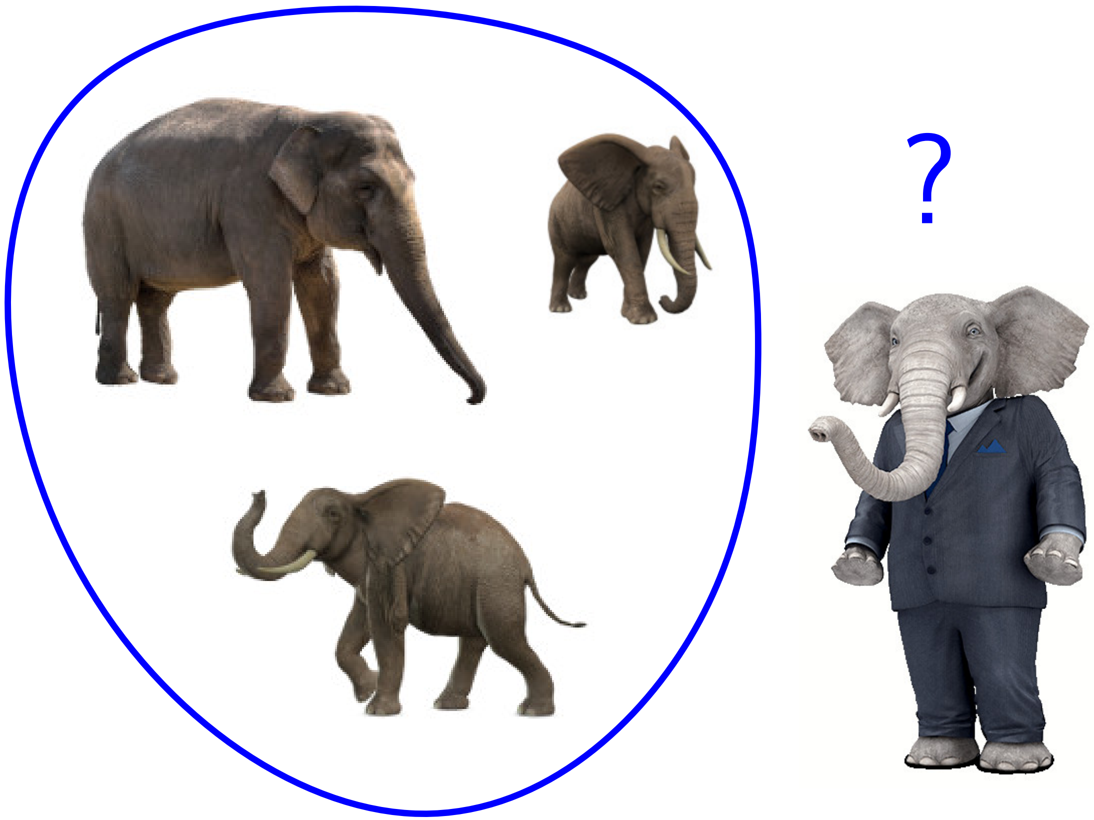
What is Clustering?

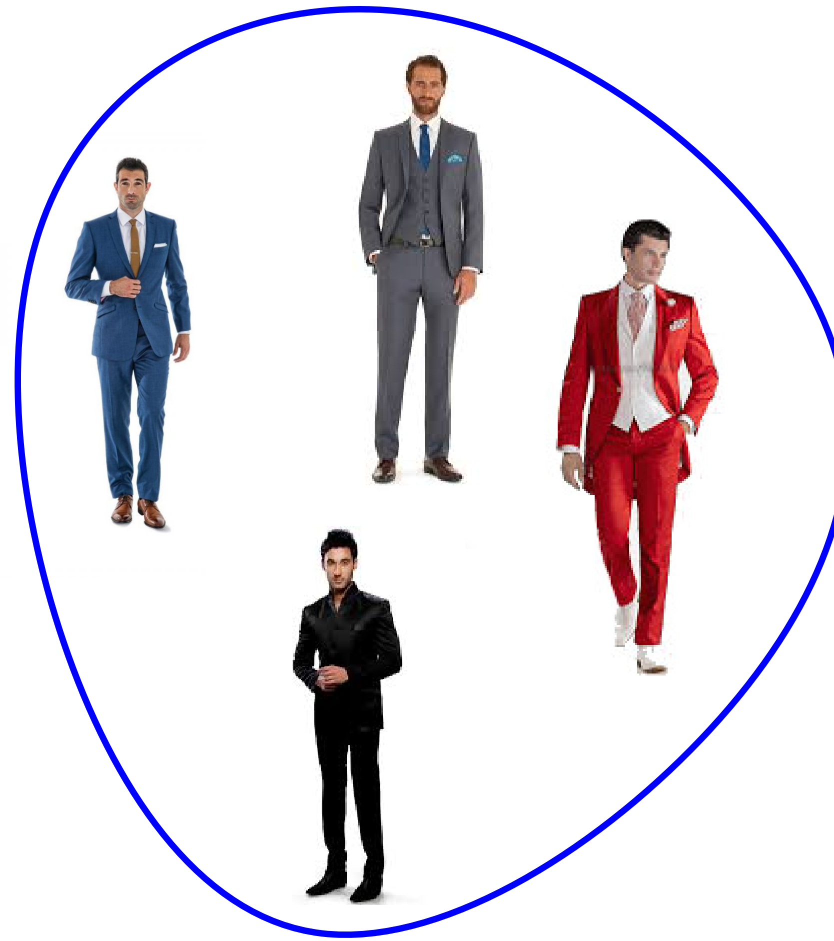
Clustering is the task of grouping similar objects into clusters



What is Clustering?

Clustering is the task of grouping similar objects into clusters



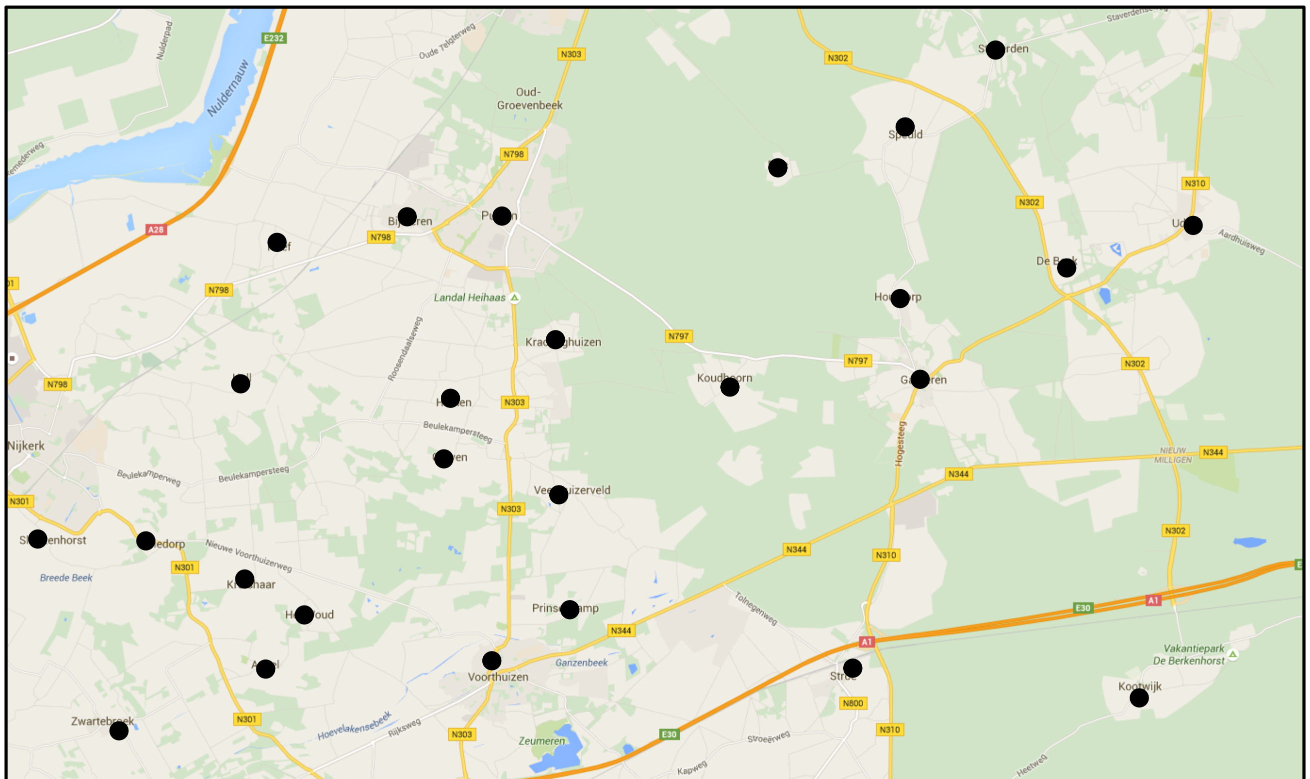


Issues:

- errors and unusual things (”noise”)
- what is the ”right” clustering?

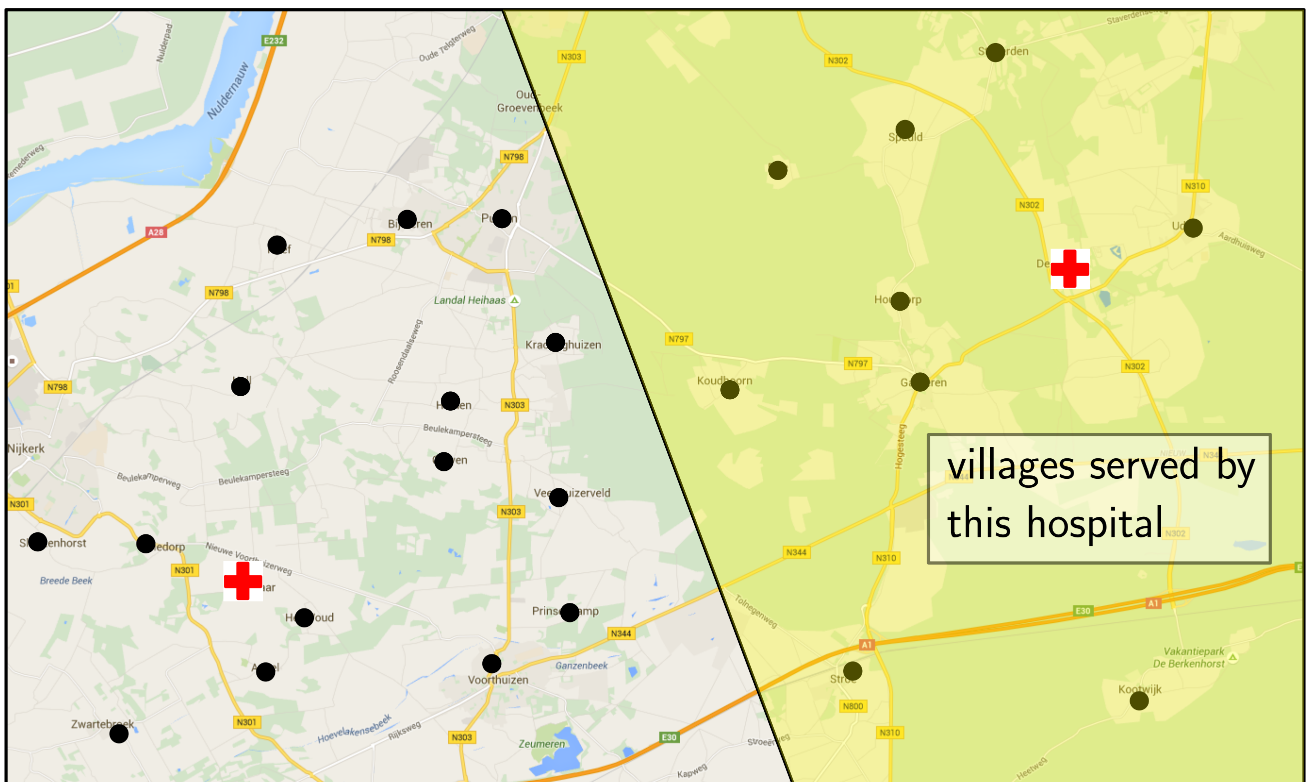
Facility Location

You may build two hospitals in two different villages serving the surrounding villages. Where do you place them to minimize the maximal distance from any village to its serving hospital?



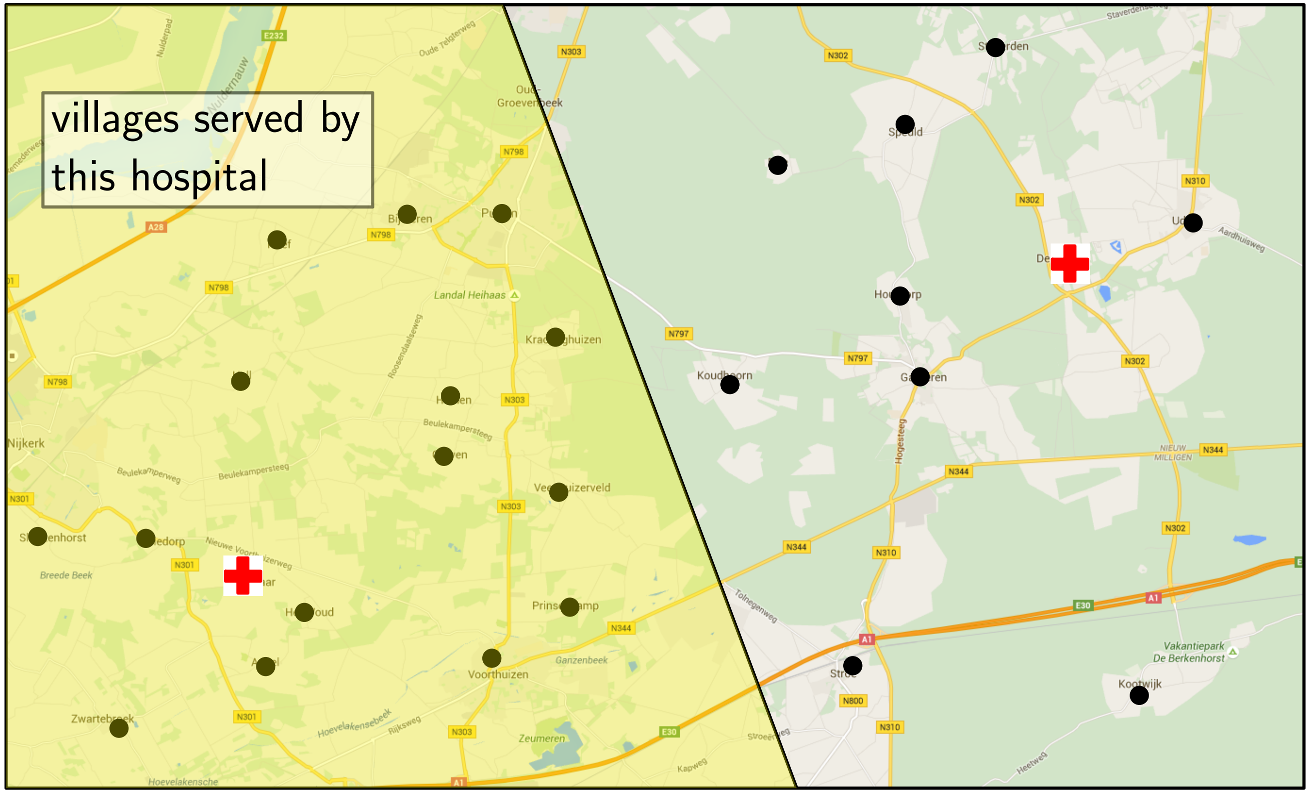
Facility Location

You may build two hospitals in two different villages serving the surrounding villages. Where do you place them to minimize the maximal distance from any village to its serving hospital?



Facility Location

You may build two hospitals in two different villages serving the surrounding villages. Where do you place them to minimize the maximal distance from any village to its serving hospital?



k-center clustering

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq P$

Problem:

- each $p_i \in P$ is "served by" its closest center

$$\operatorname*{argmin}_{c_j \in C} \|p_i - c_j\|$$

- all points served by a center c_j together form a "cluster"

- we want to choose $\{c_1, \ldots, c_k\}$ to minimize the cost function

$$\phi(P, C) = \max_{p_i \in P} \left\| p_i - \operatorname*{argmin}_{c_j \in C} \|p_i - c_j\| \right\|$$

Gonzales' algorithm for k-center

Input: set of points $P = \{p_1, \dots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \dots, c_k\} \subseteq P$

Algorithm:

- choose an arbitrary point $p_i \in P$ and set $c_1 = p_i$
- for $t = 2, \dots, k$ set

$$c_t = \underset{p_i \in P}{\operatorname{argmax}} \left\| p_i - \underset{c_j \in \{c_1, \dots, c_{t-1}\}}{\operatorname{argmin}} \|p_i - c_j\| \right\|$$

Gonzales' algorithm for k-center

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq P$

Algorithm:

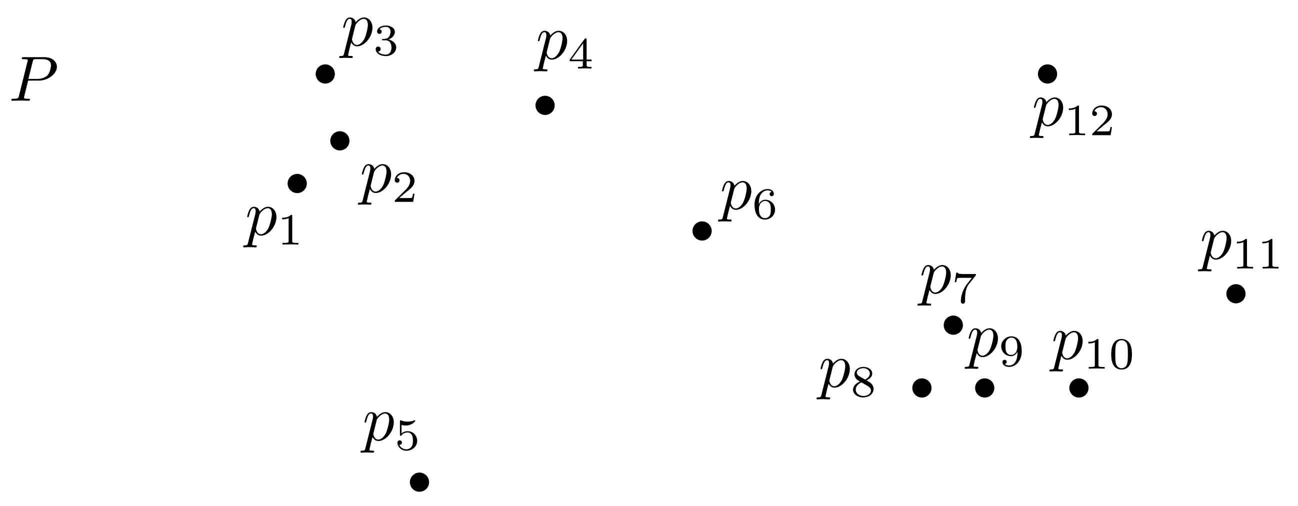
- choose an arbitrary point $p_i \in P$ and set $c_1 = p_i$
- for $t = 2, \ldots, k$ set

$$c_t = \operatorname*{argmax}_{p_i \in P} \left\| p_i - \operatorname*{argmin}_{c_j \in \{c_1, \ldots, c_{t-1}\}} \|p_i - c_j\| \right\|$$

"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

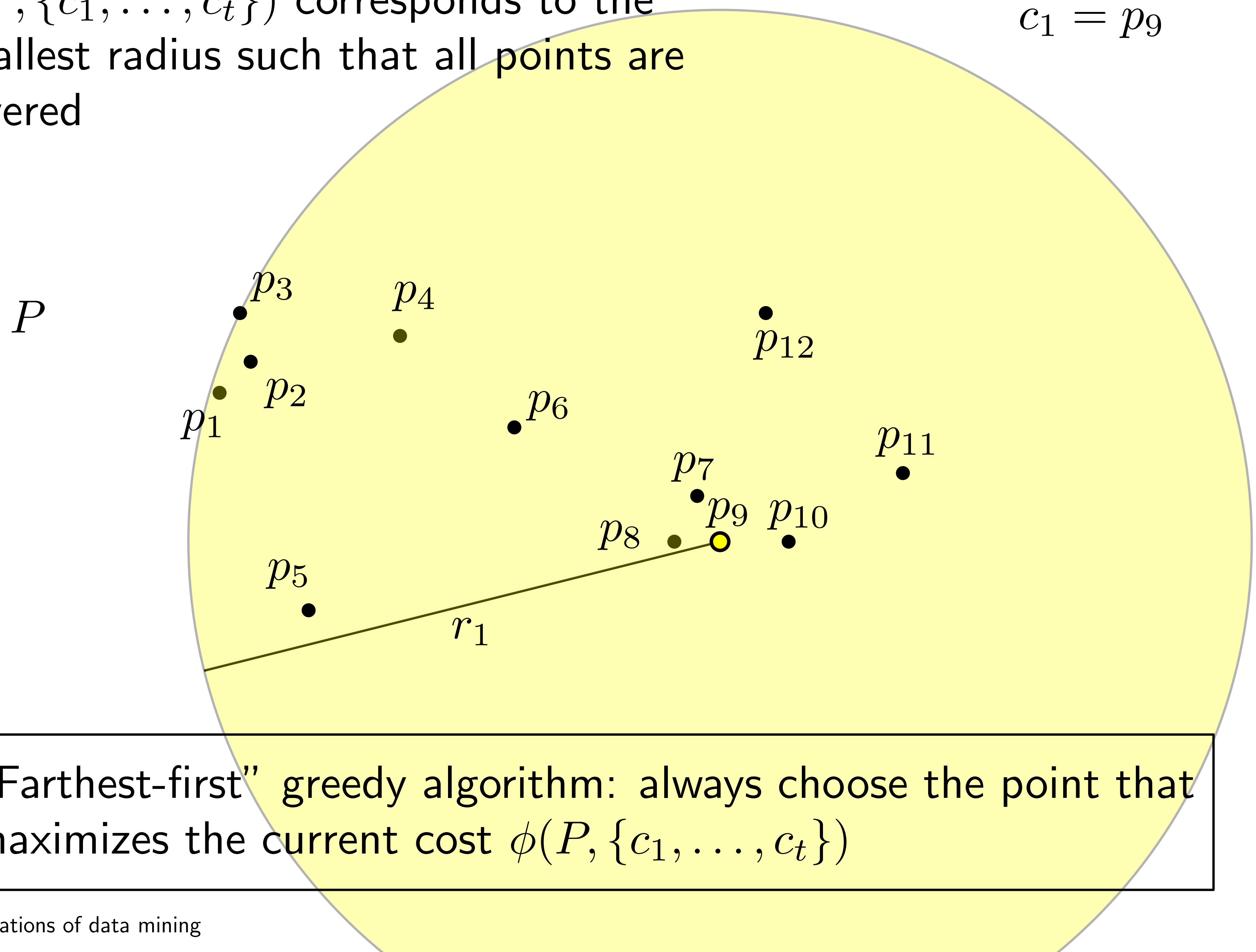


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$

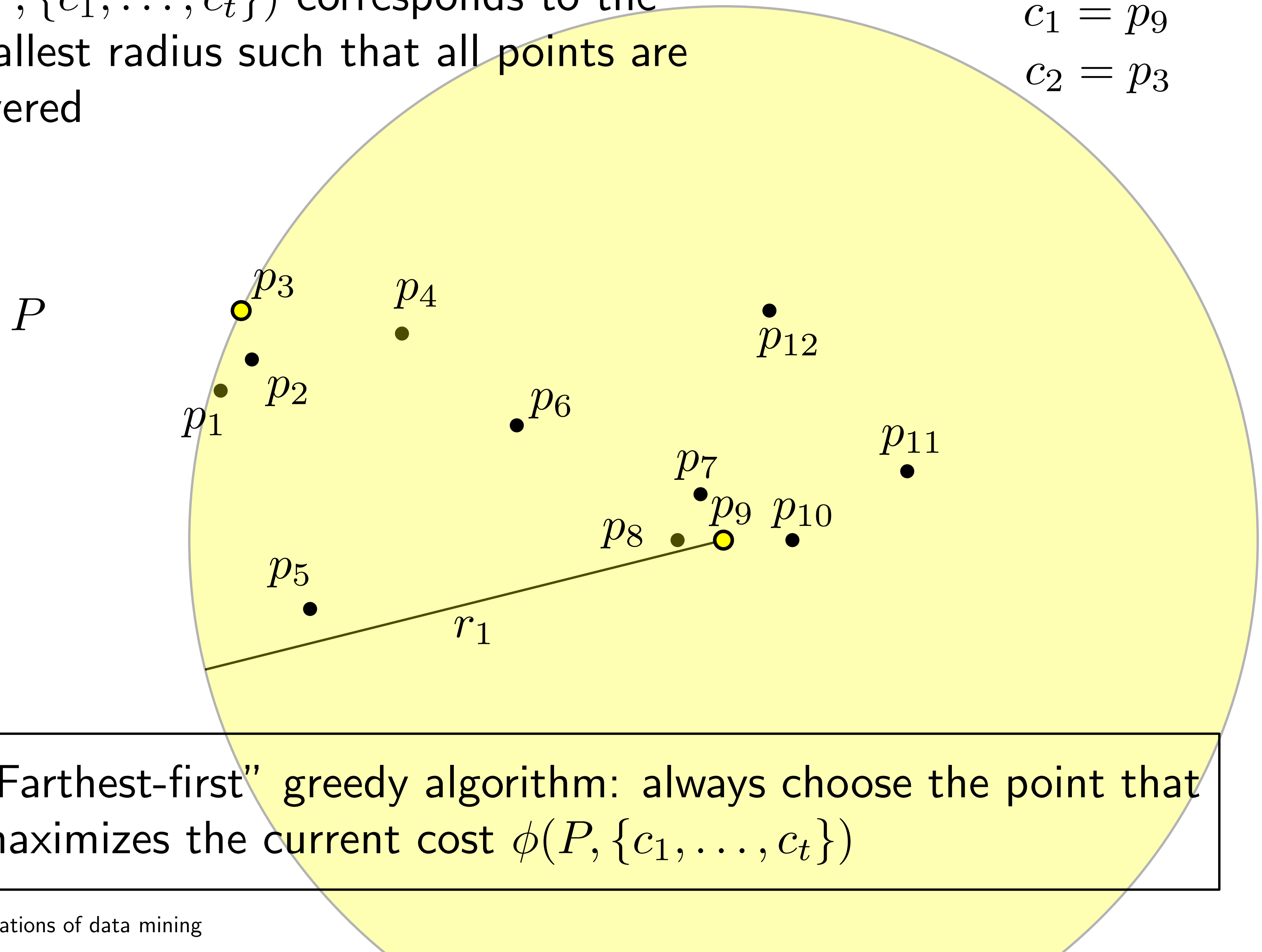


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$

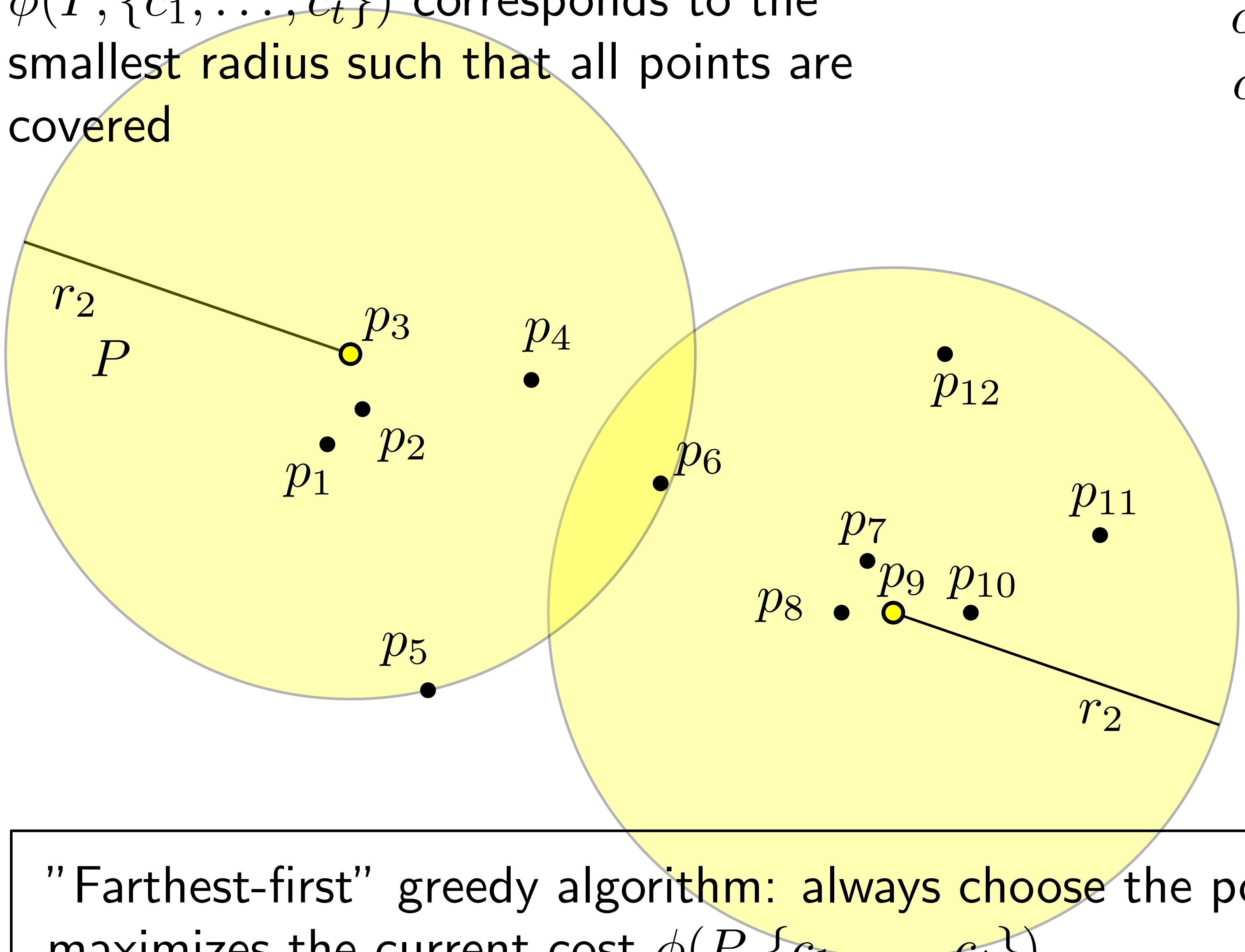


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$

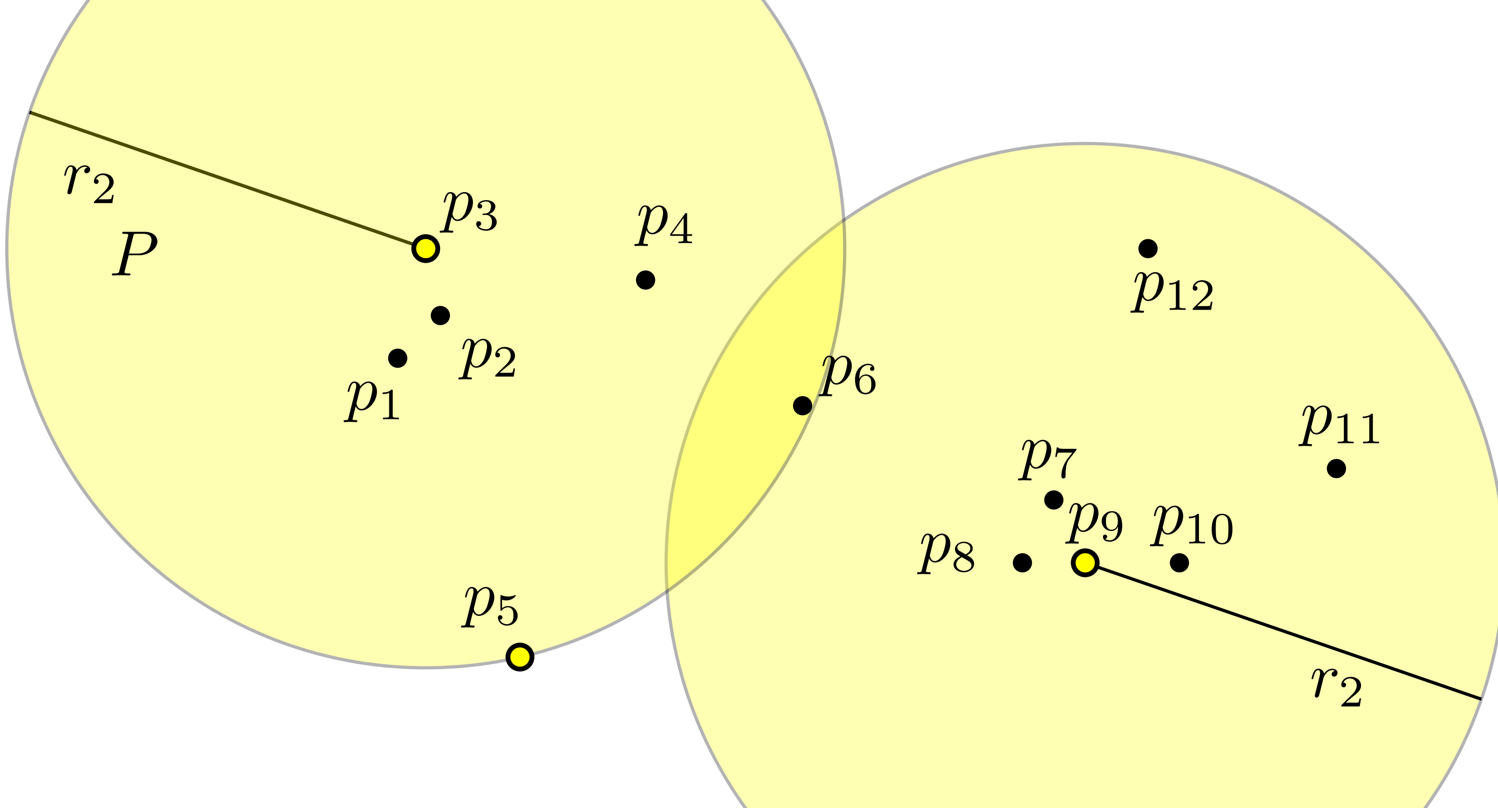


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$

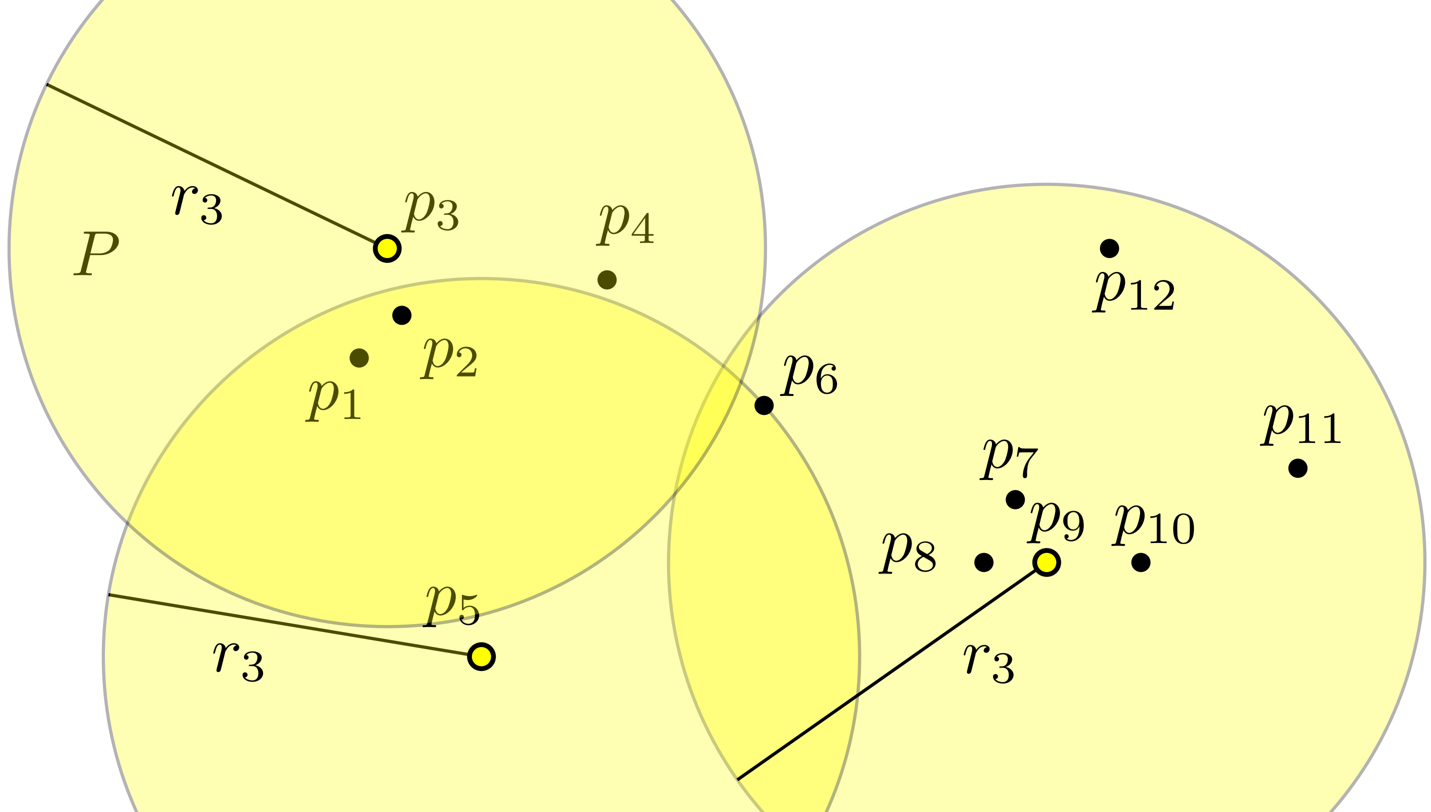


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$

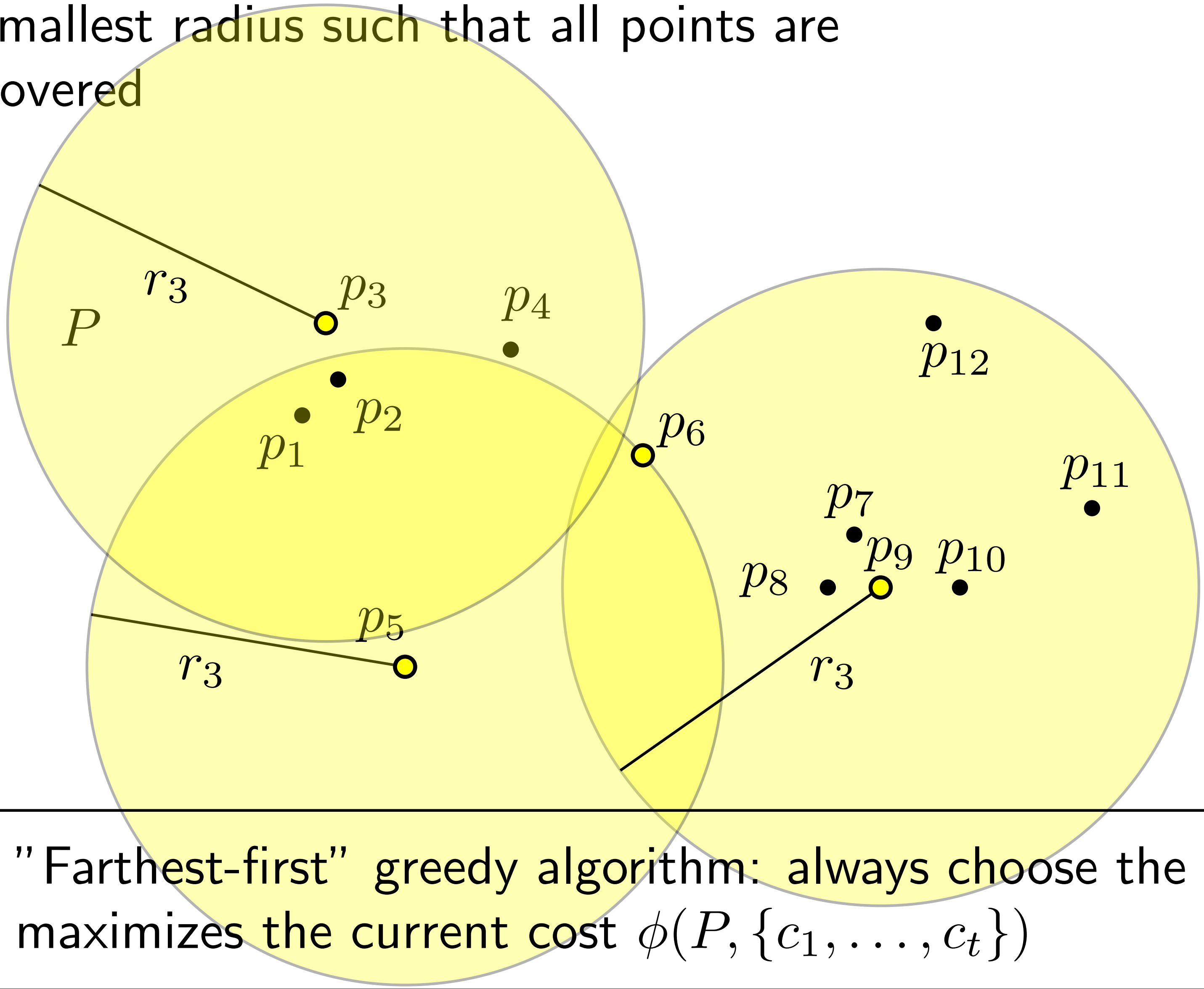


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$

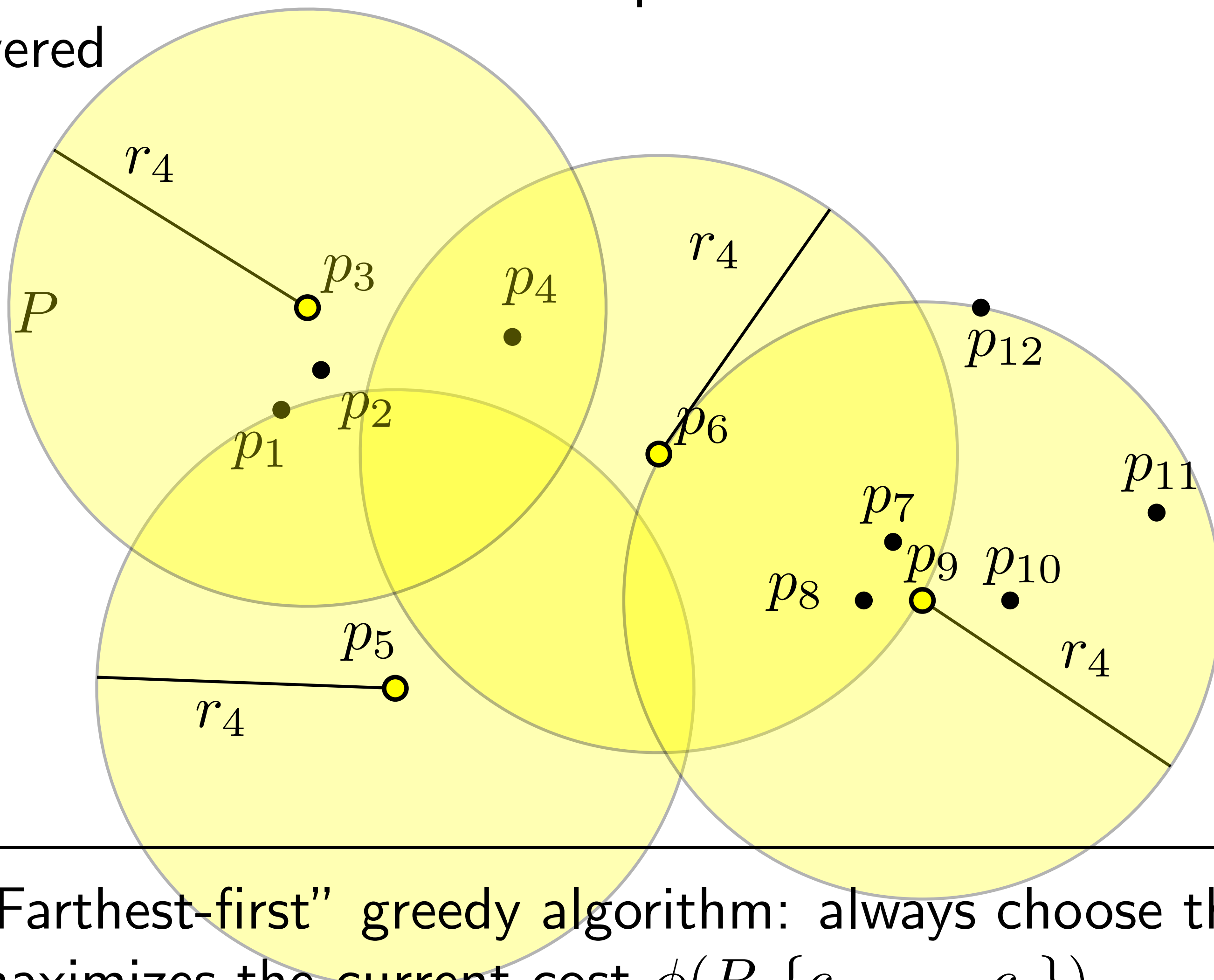


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$

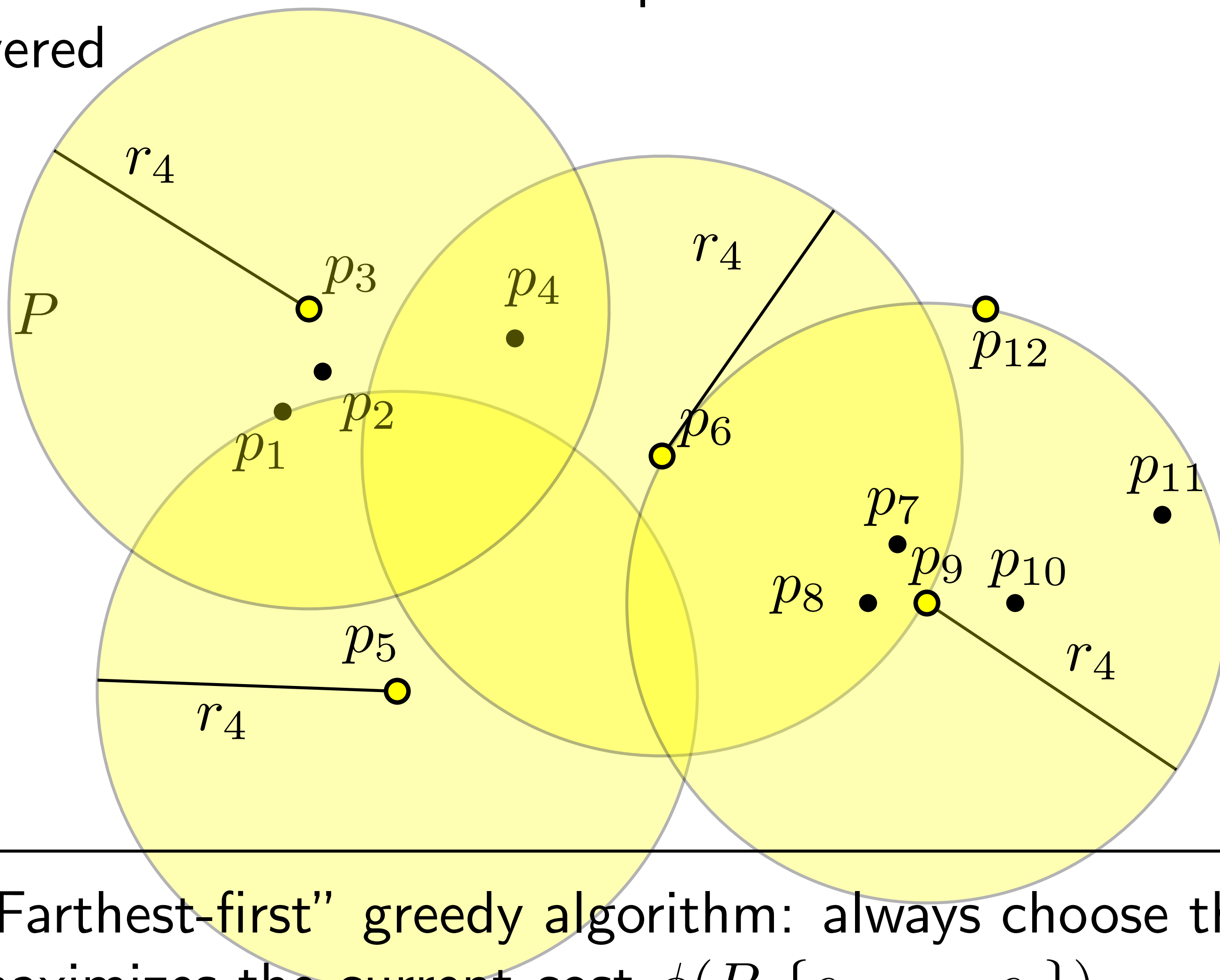


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$
$c_5 = p_{12}$

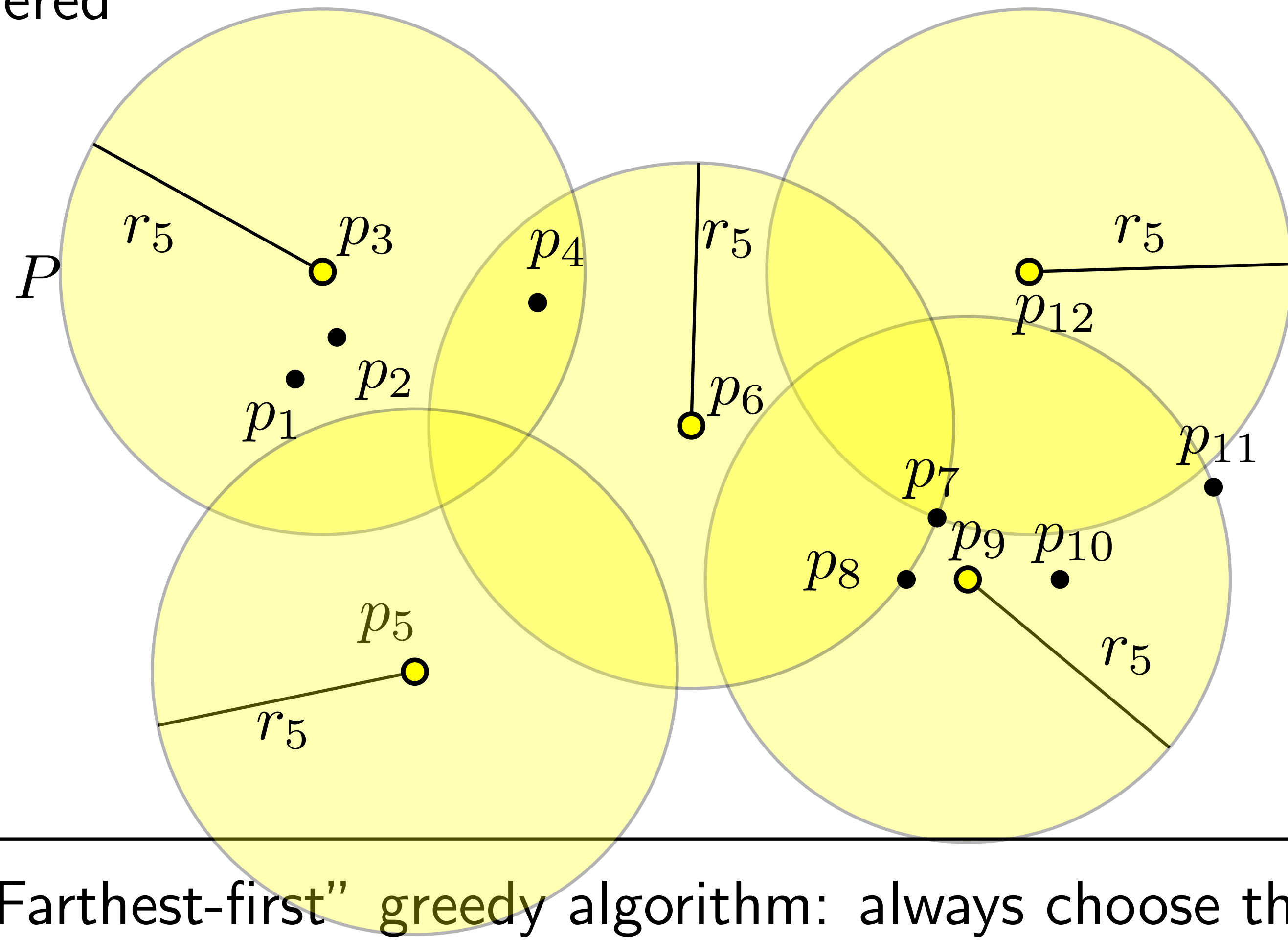


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$
$c_5 = p_{12}$

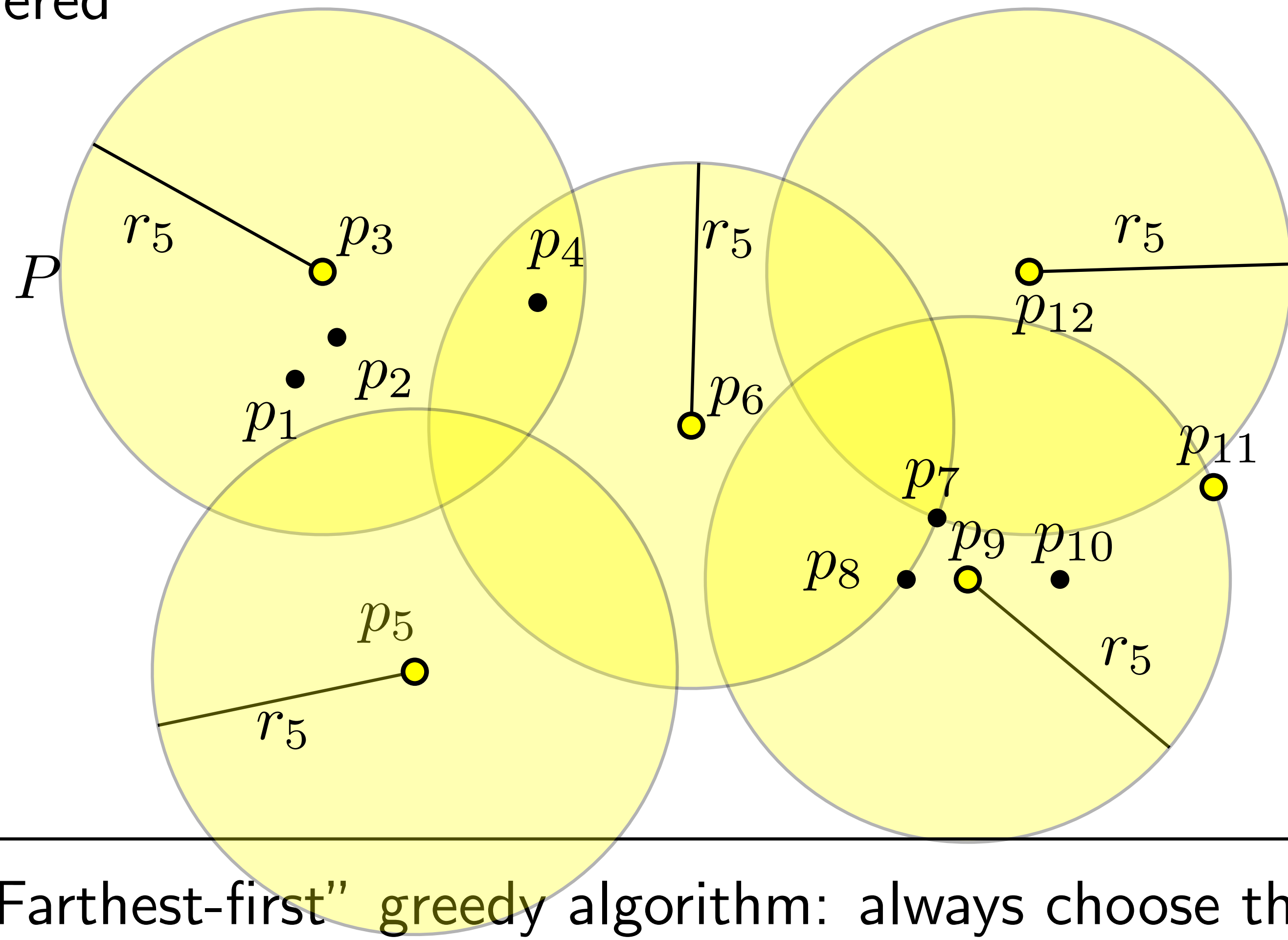


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \dots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$
$c_5 = p_{12}$
$c_6 = p_{11}$

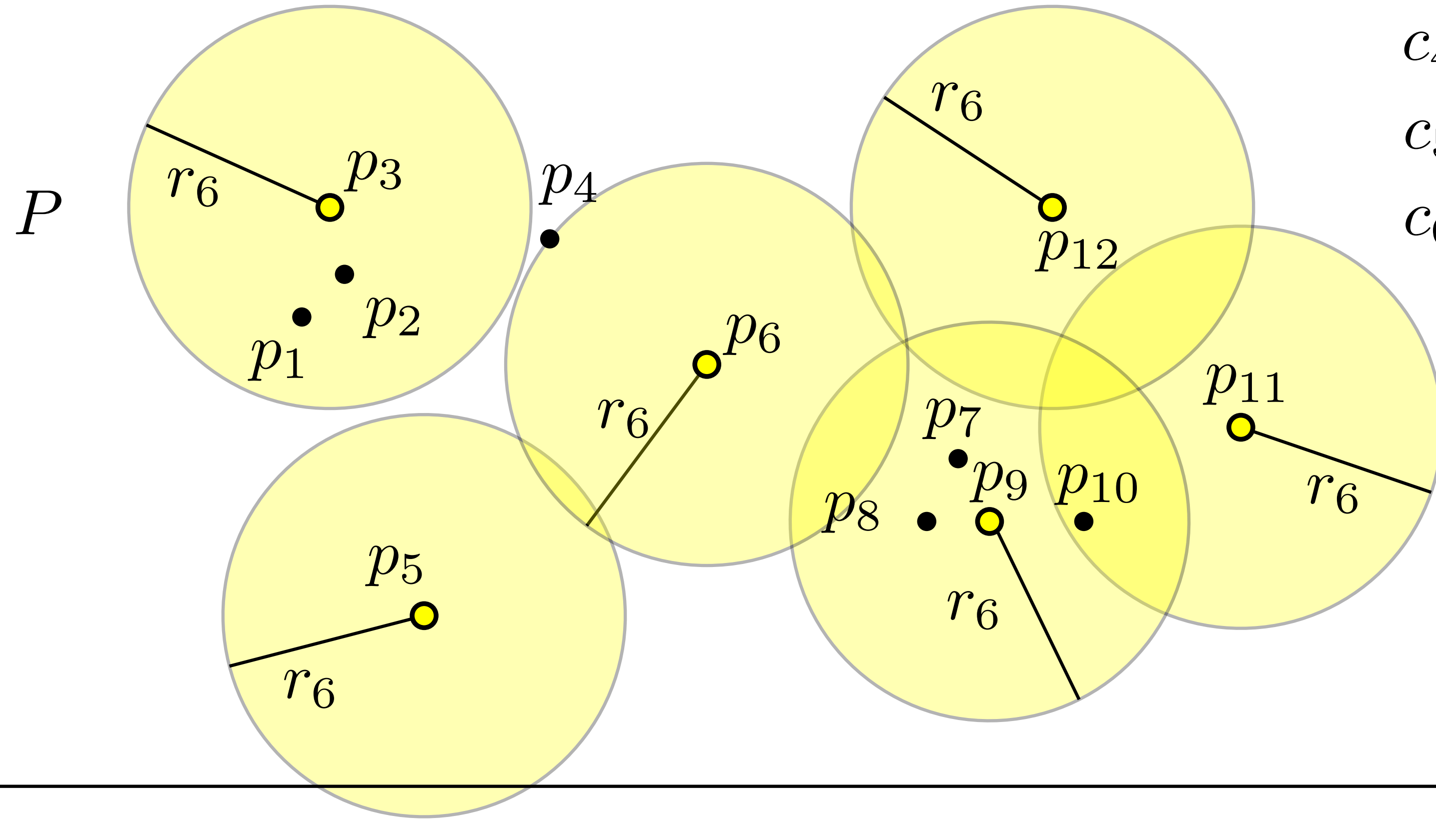


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \dots, c_t\})$

Gonzales' algorithm for k-center

$\phi(P, \{c_1, \ldots, c_t\})$ corresponds to the smallest radius such that all points are covered

$c_1 = p_9$
$c_2 = p_3$
$c_3 = p_5$
$c_4 = p_6$
$c_5 = p_{12}$
$c_6 = p_{11}$

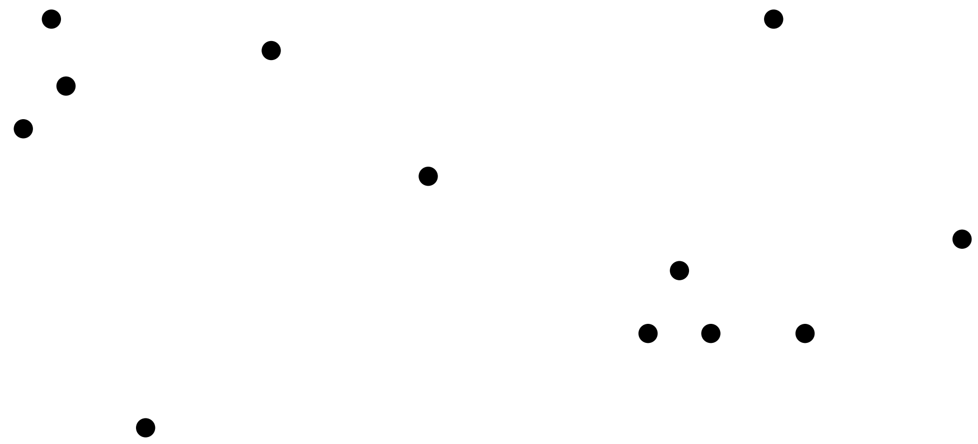


"Farthest-first" greedy algorithm: always choose the point that maximizes the current cost $\phi(P, \{c_1, \ldots, c_t\})$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

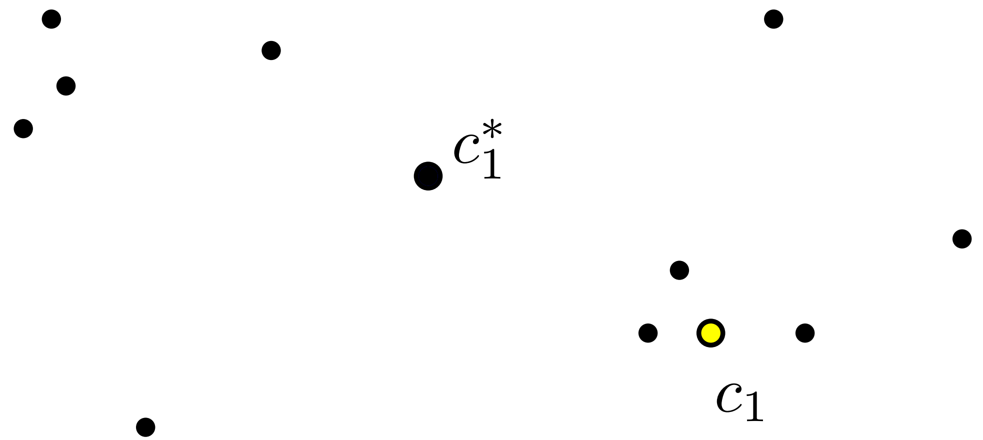


Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k = 1)$

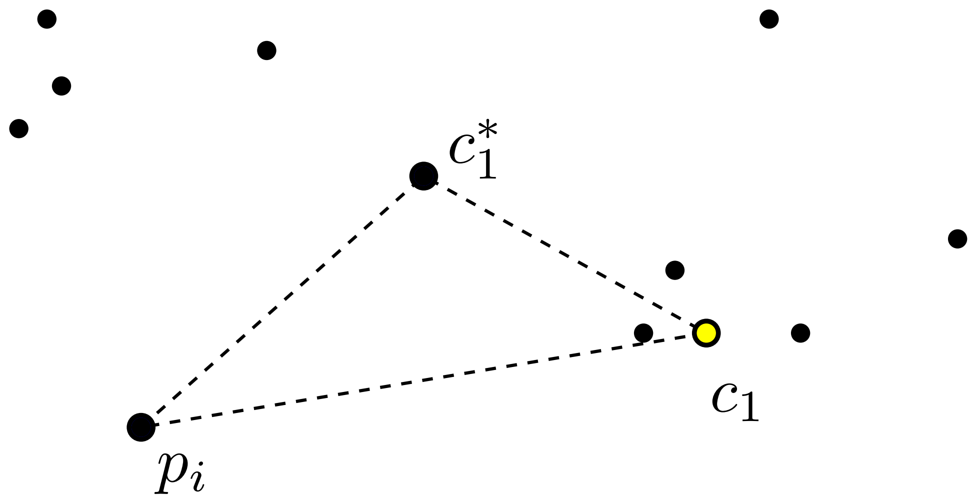


Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k = 1)$



(Triangle inequality)

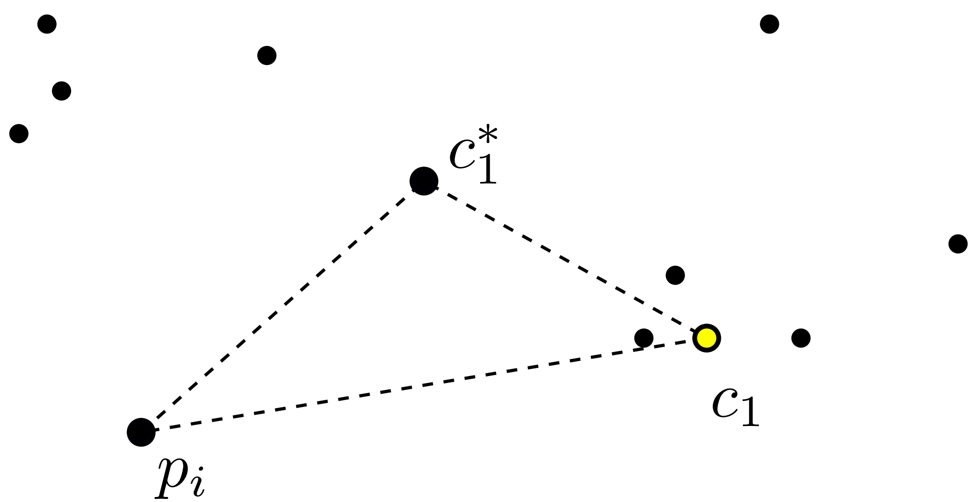
$$\forall p_i \in P : \qquad \|p_i - c_1\| \leq \|p_i - c_1^*\| + \|c_1^* - c_1\|$$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k = 1)$



(Triangle inequality)

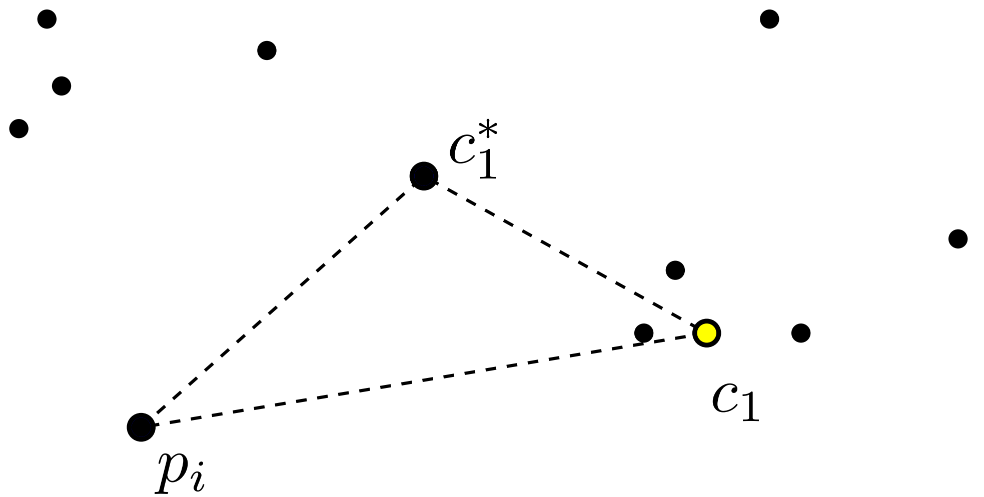
$$\forall p_i \in P: \qquad \|p_i - c_1\| \leq \underbrace{\|p_i - c_1^*\|}_{\leq \phi(P, C^*)} + \underbrace{\|c_1^* - c_1\|}_{\leq \phi(P, C^*)}$$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k = 1)$



(Triangle inequality)

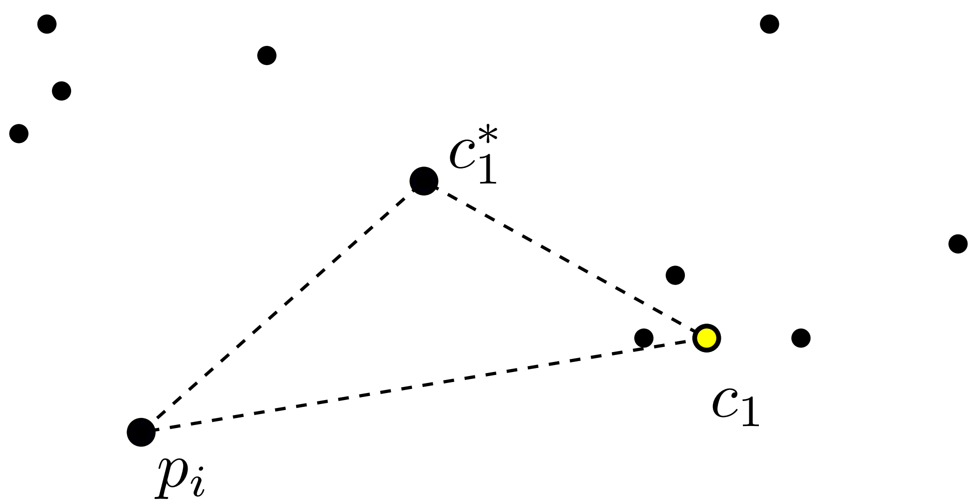
$$\forall p_i \in P : \quad \|p_i - c_1\| \leq \underbrace{\|p_i - c_1^*\|}_{\leq \phi(P, C^*)} + \underbrace{\|c_1^* - c_1\|}_{\leq \phi(P, C^*)} \leq 2\phi(P, C^*)$$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k = 1)$



(Triangle inequality)

$$\forall p_i \in P : \quad \|p_i - c_1\| \leq \underbrace{\|p_i - c_1^*\|}_{\leq \phi(P, C^*)} + \underbrace{\|c_1^* - c_1\|}_{\leq \phi(P, C^*)} \leq 2\phi(P, C^*)$$

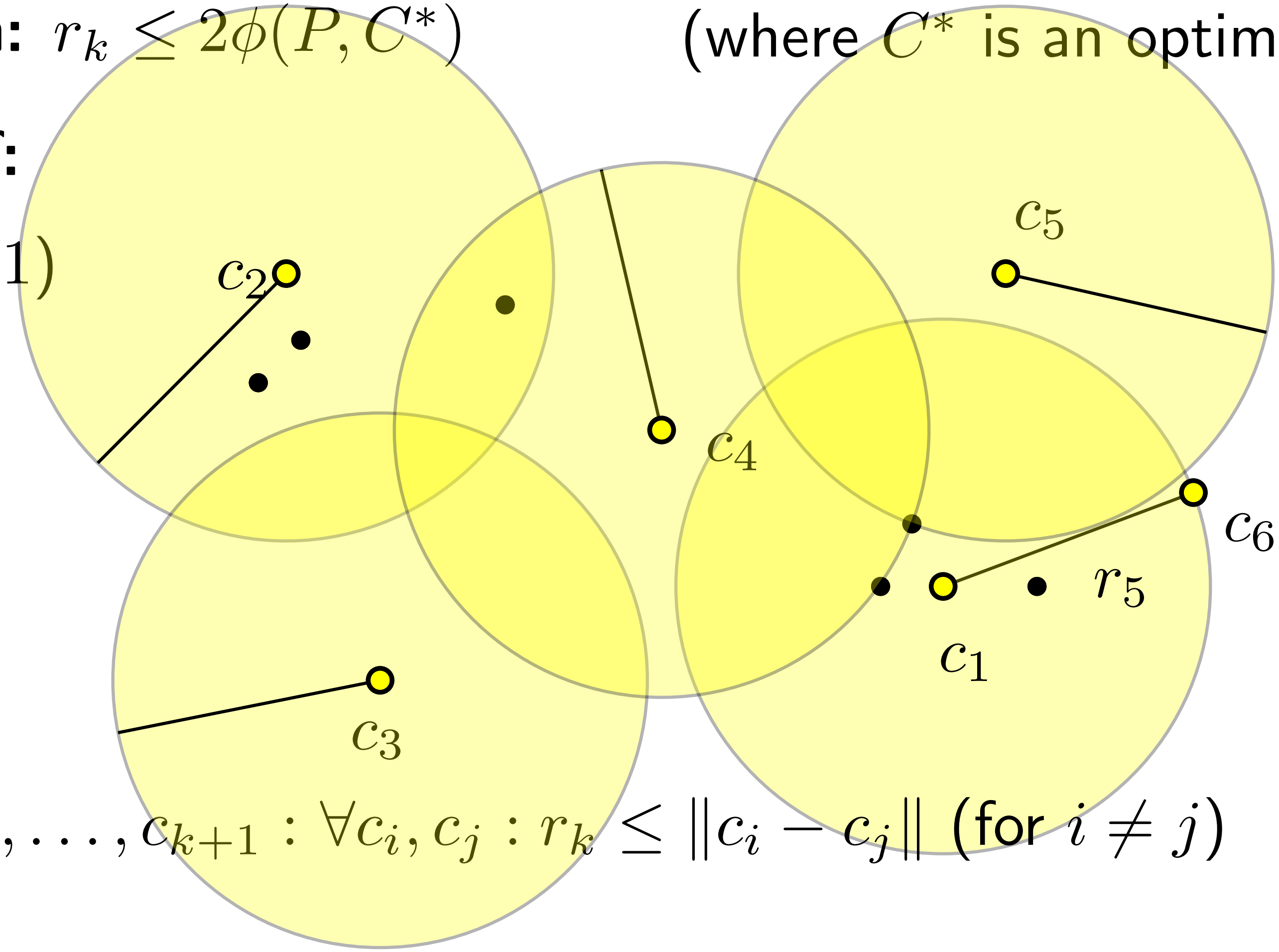
$$\Rightarrow r_1 \leq 2\phi(P, C^*)$$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k \geq 1)$



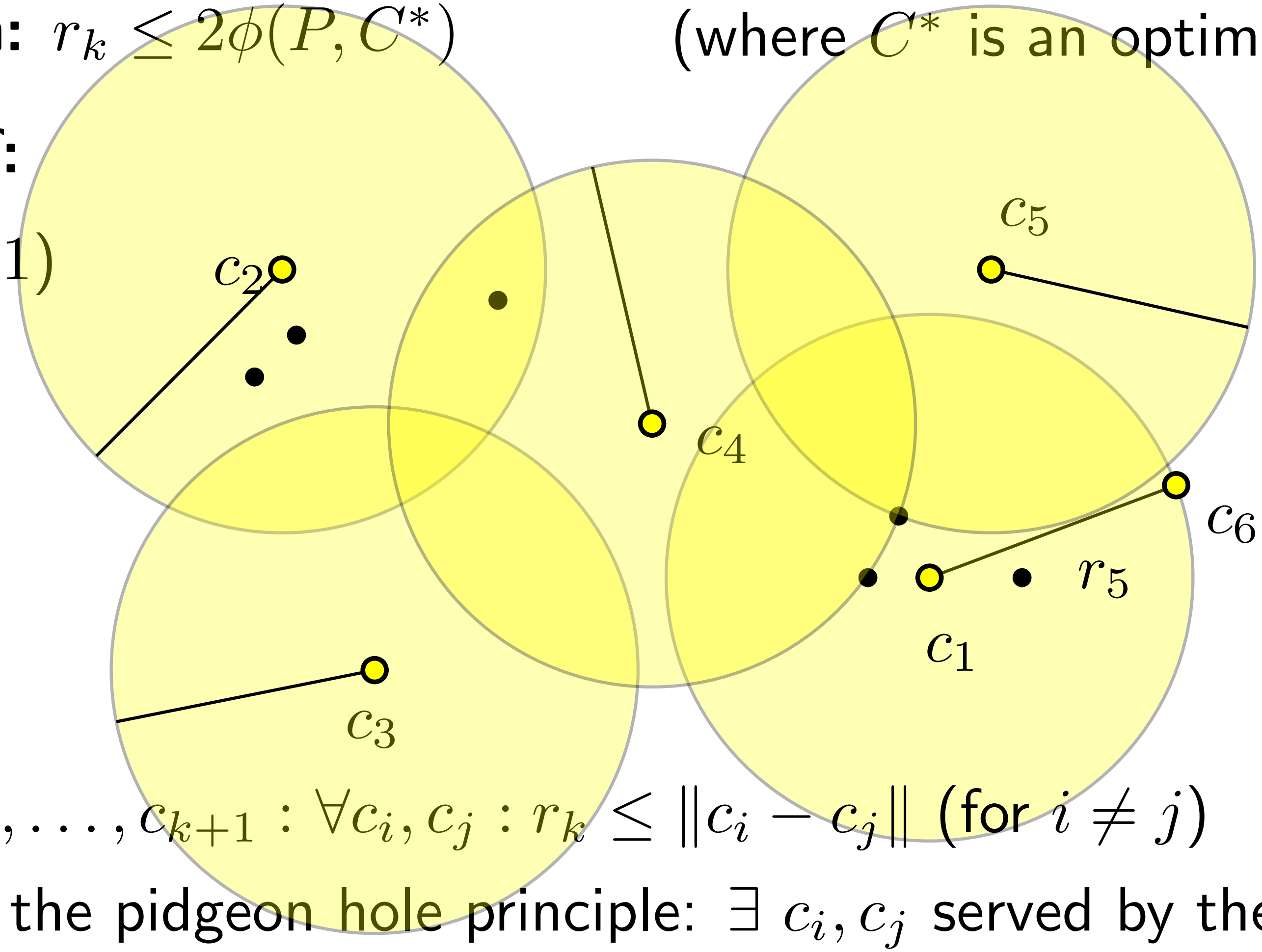
For $c_1, \ldots, c_{k+1} : \forall c_i, c_j : r_k \leq \|c_i - c_j\|$ (for $i \neq j$)

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k \geq 1)$



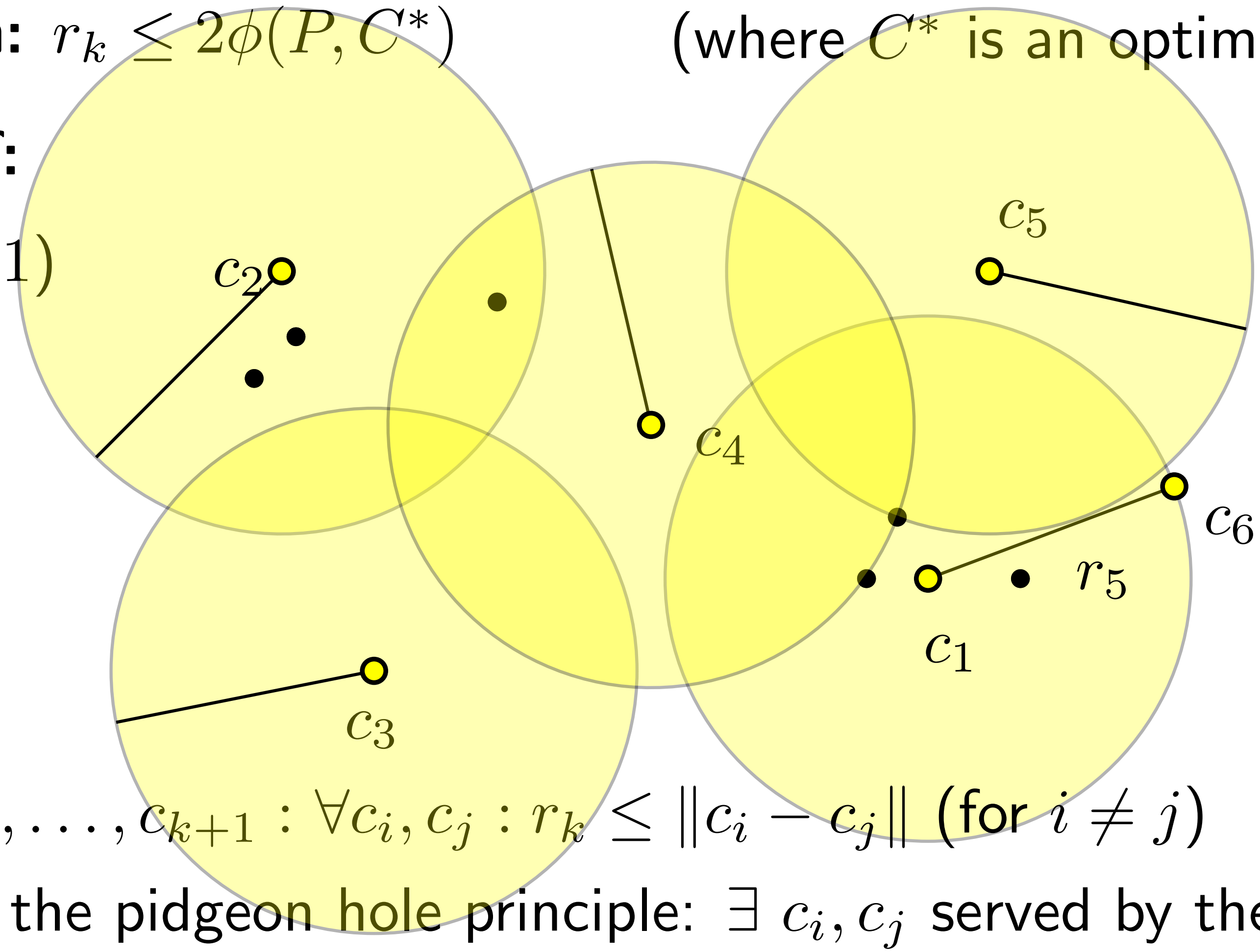
For $c_1, \ldots, c_{k+1} : \forall c_i, c_j : r_k \leq \|c_i - c_j\|$ (for $i \neq j$)

$\Rightarrow$ by the pidgeon hole principle: $\exists\ c_i, c_j$ served by the same c_s^*

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k \geq 1)$



For $c_1, \ldots, c_{k+1} : \forall c_i, c_j : r_k \leq \|c_i - c_j\|$ (for $i \neq j$)

$\Rightarrow$ by the pidgeon hole principle: $\exists\ c_i, c_j$ served by the same c_s^*

$\Rightarrow$ by the triangle inequality for c_i, c_j, c_s^*:

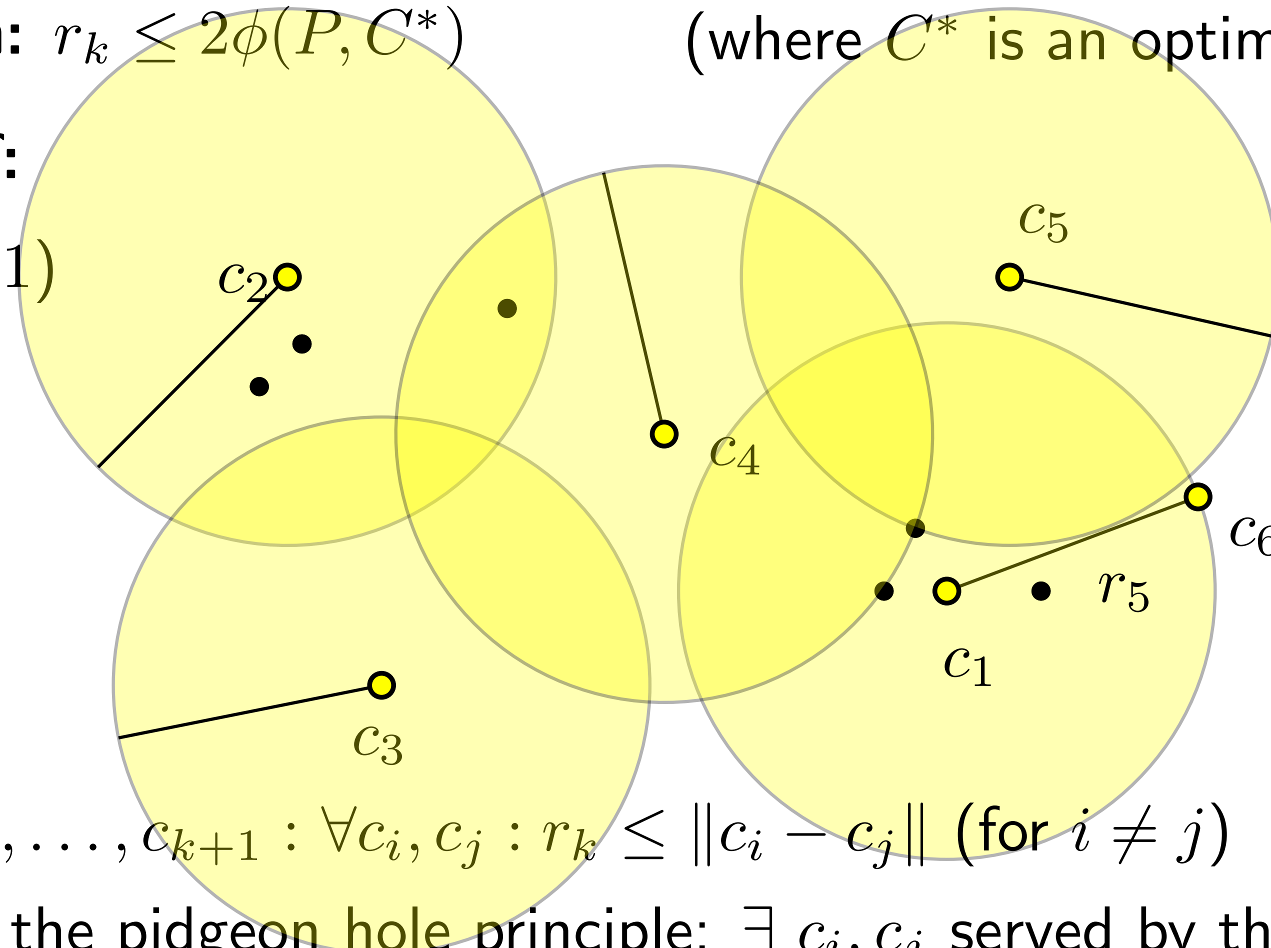
$$\|c_i - c_j\| \leq \|c_i - c_s^*\| + \|c_s^* - c_j\| \leq 2\phi(P, C^*)$$

Gonzales' algorithm (Analysis)

Claim: $r_k \leq 2\phi(P, C^*)$ (where C^* is an optimal solution)

Proof:

$(k \geq 1)$



For $c_1, \ldots, c_{k+1} : \forall c_i, c_j : r_k \leq \|c_i - c_j\|$ (for $i \neq j$)

$\Rightarrow$ by the pidgeon hole principle: $\exists\ c_i, c_j$ served by the same c_s^*

$\Rightarrow$ by the triangle inequality for c_i, c_j, c_s^*:

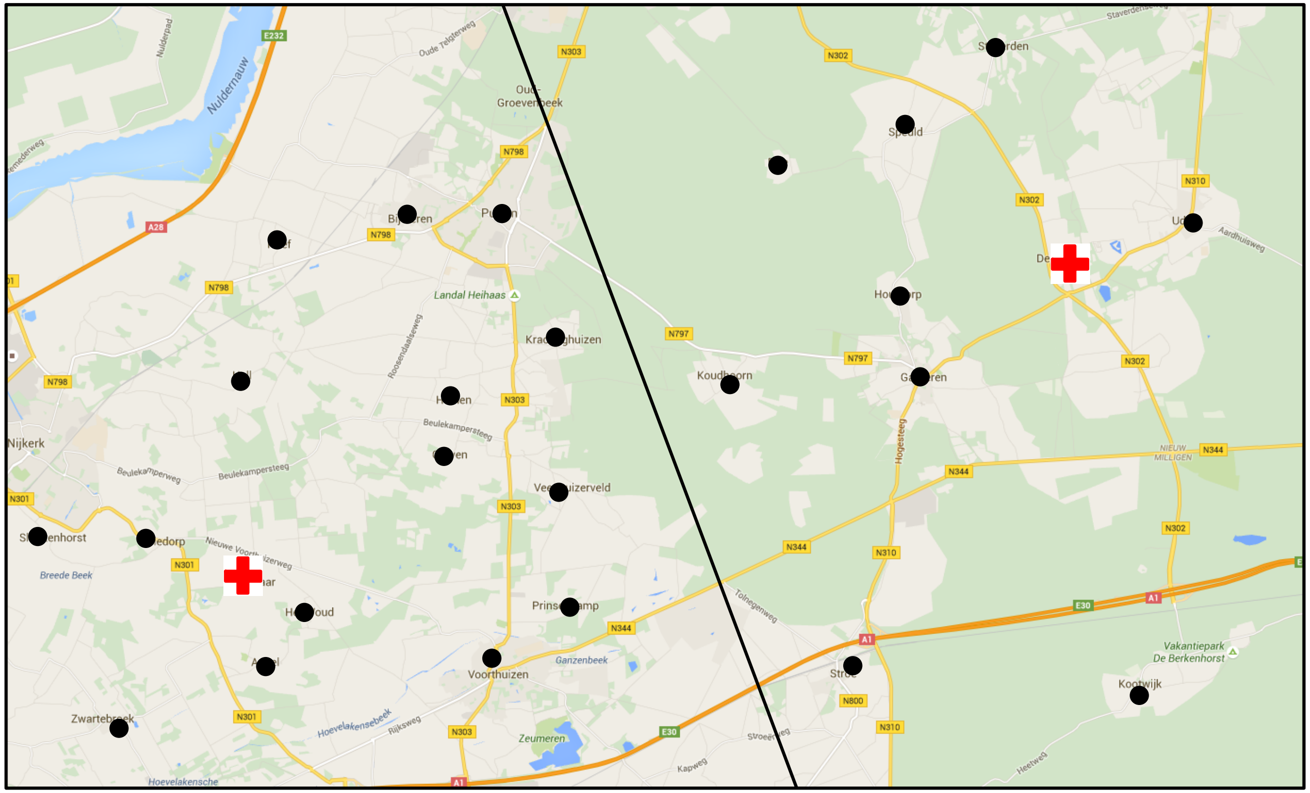
$$\|c_i - c_j\| \leq \|c_i - c_s^*\| + \|c_s^* - c_j\| \leq 2\phi(P, C^*)$$

So we have $r_k \leq 2\phi(P, C^*)$

Facility Location (Variant)

You may build two hospitals in two different villages serving the surrounding villages. Where do you place them to minimize the maximal distance from any village to its serving hospital?

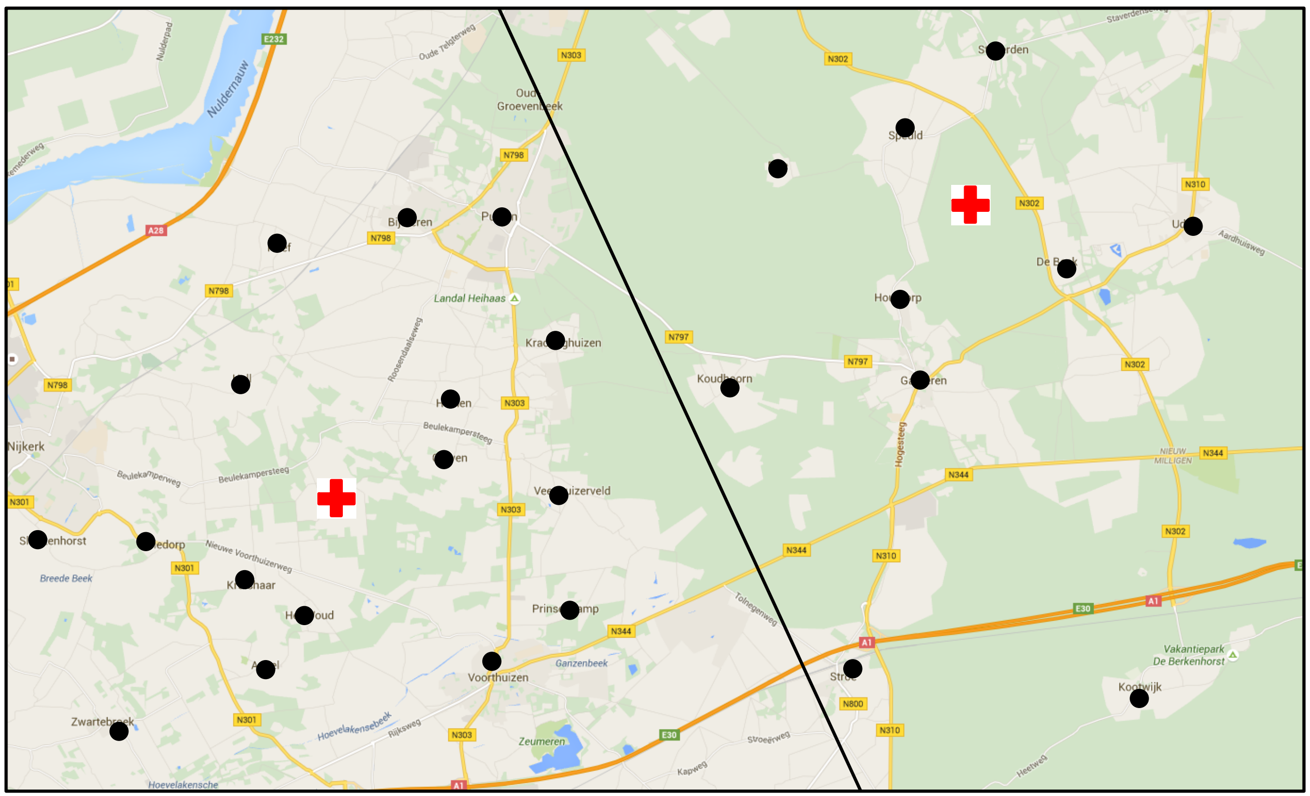
Variant:
- minimize the (squared) average distance



Facility Location (Variant)

You may build two hospitals in two different villages serving the surrounding villages. Where do you place them to minimize the maximal distance from any village to its serving hospital?

Variant:
- minimize the (squared) average distance
- hospitals may be built "in the middle of nowhere"



k-means clustering

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq \mathbb{R}^d$

centers may be in the middle of nowhere

Problem:

- each $p_i \in P$ is associated with its closest center

$$\operatorname*{argmin}_{c_j \in C} \|p_i - c_j\|$$

- points associated with a center c_j together form a "cluster".

- we want to choose $\{c_1, \ldots, c_k\}$ to minimize the cost function

average instead of maximum

$$\phi(P, C) = \sum_{p_i \in P} \left\| p_i - \operatorname*{argmin}_{c_j \in C} \|p_i - c_j\| \right\|^2$$

(squared distance)

Lloyd's algorithm for k-means

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq \mathbb{R}^d$

Algorithm:

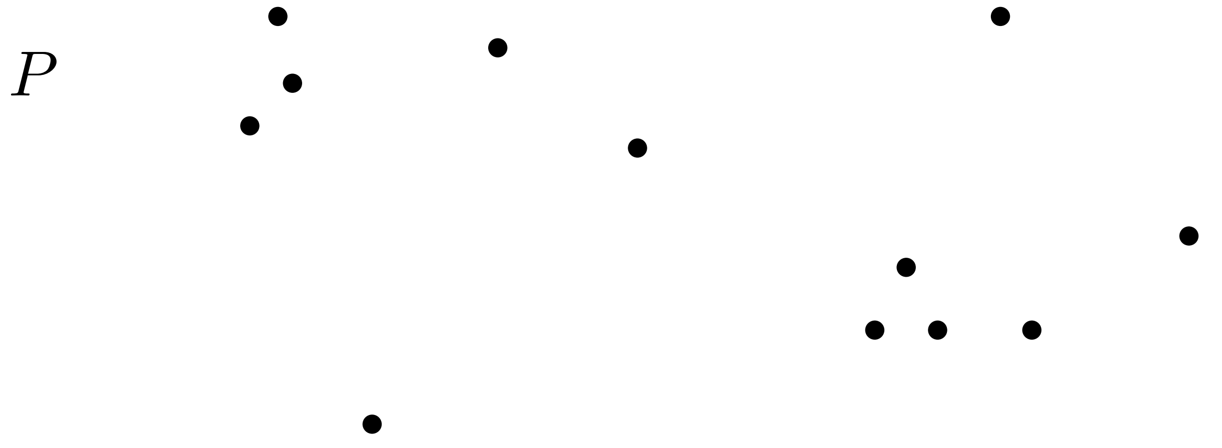
- choose initial centers arbitrarily $\{c_1, \ldots, c_k\}$ from P
- until $\{c_1, \ldots, c_k\}$ does not change anymore:

  (1) assign each $p_i \in P$ to its closest center

  (2) update center for each cluster Θ_j:

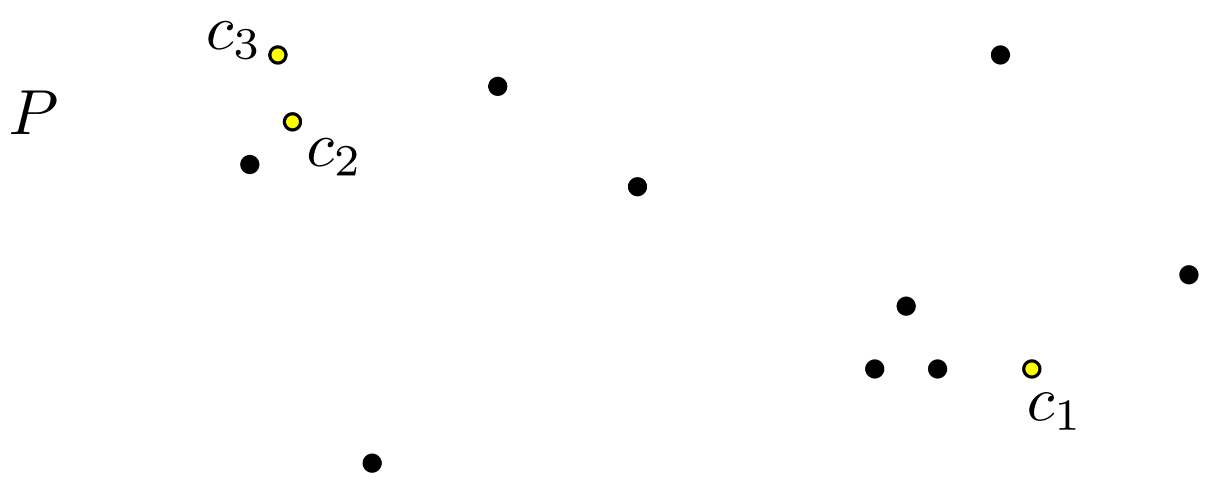
$$c_j := \frac{1}{m} \sum_{p_i \in \Theta_j} p_i$$

Lloyd's algorithm for k-means



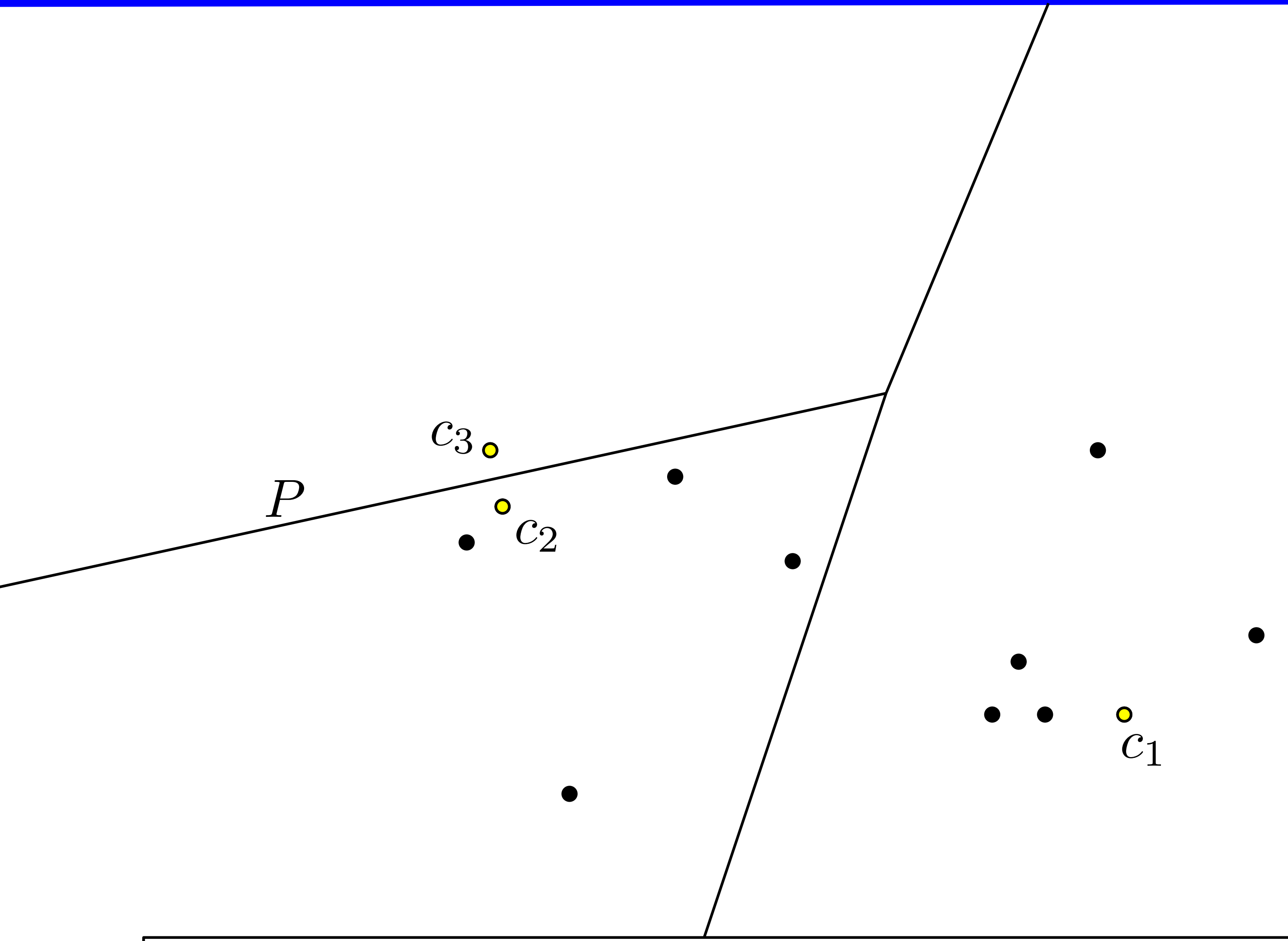
Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



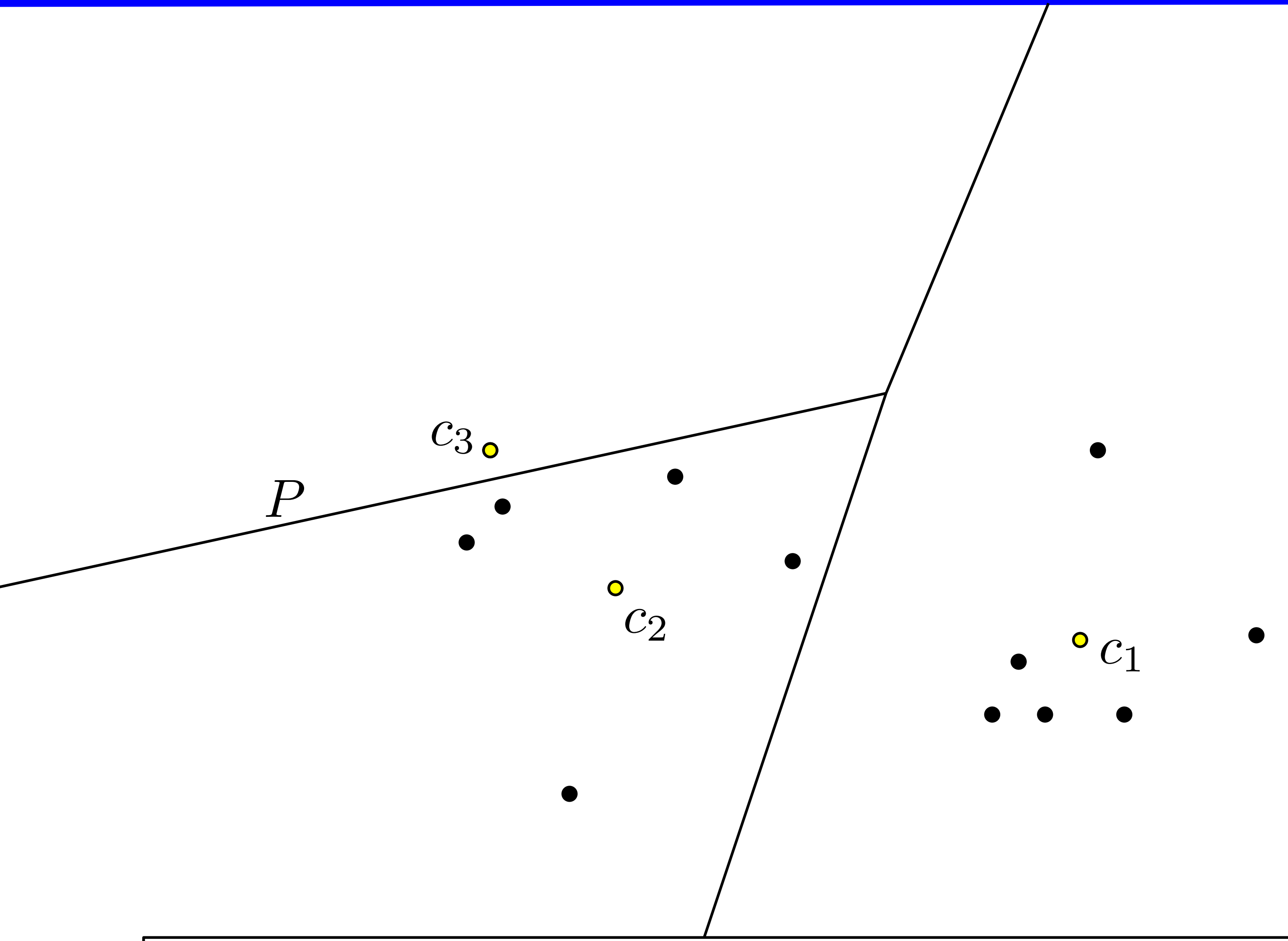
Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



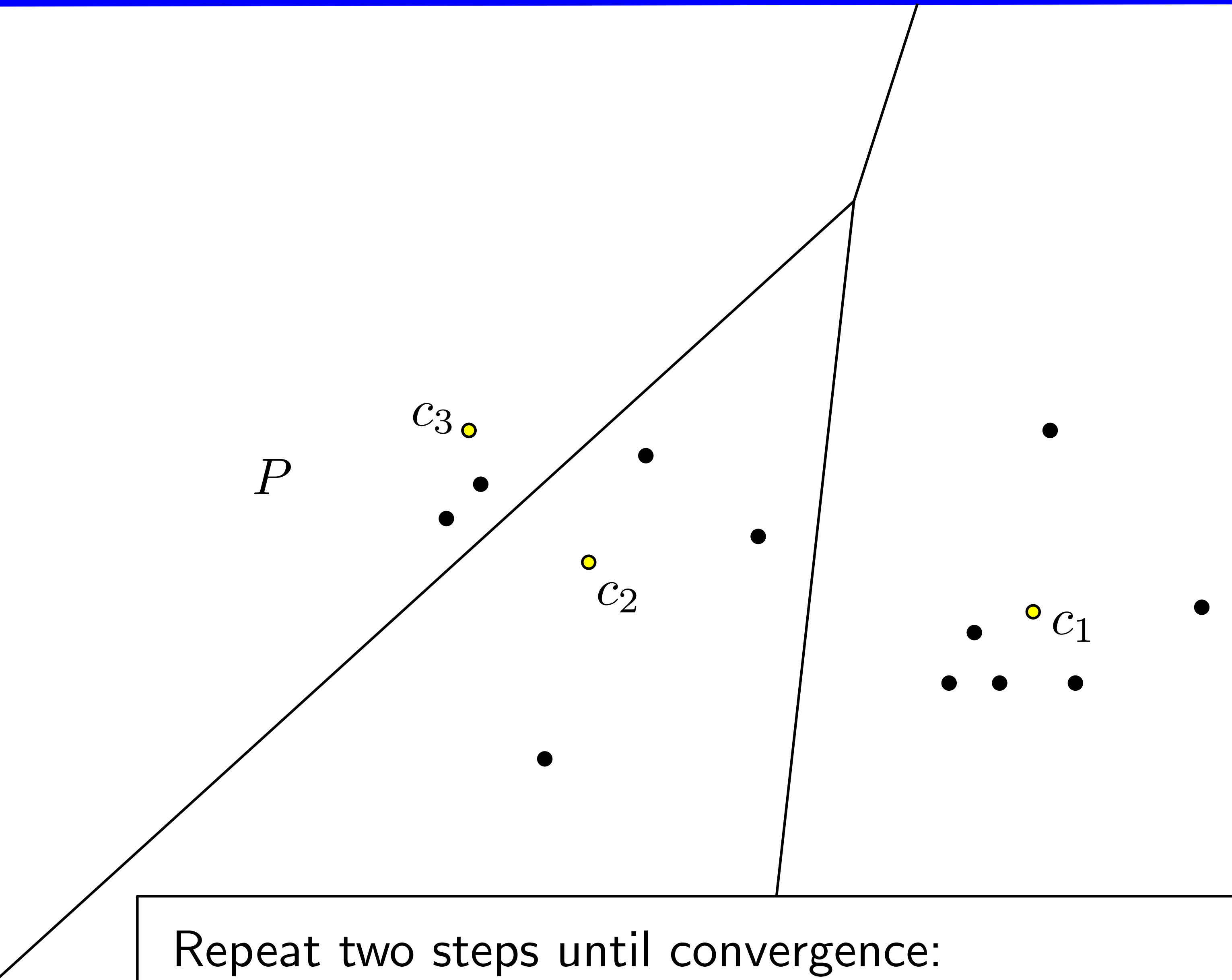
Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



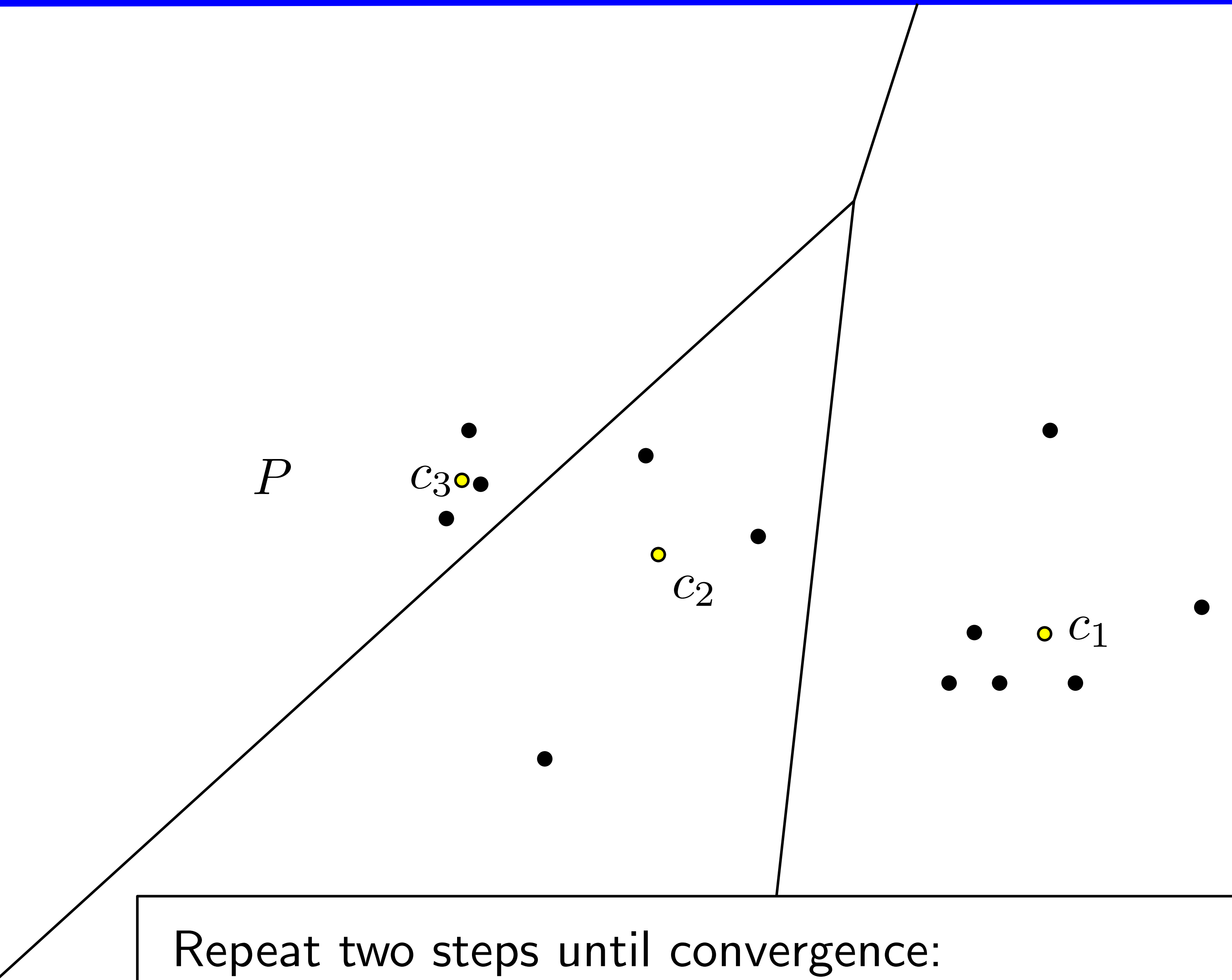
Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



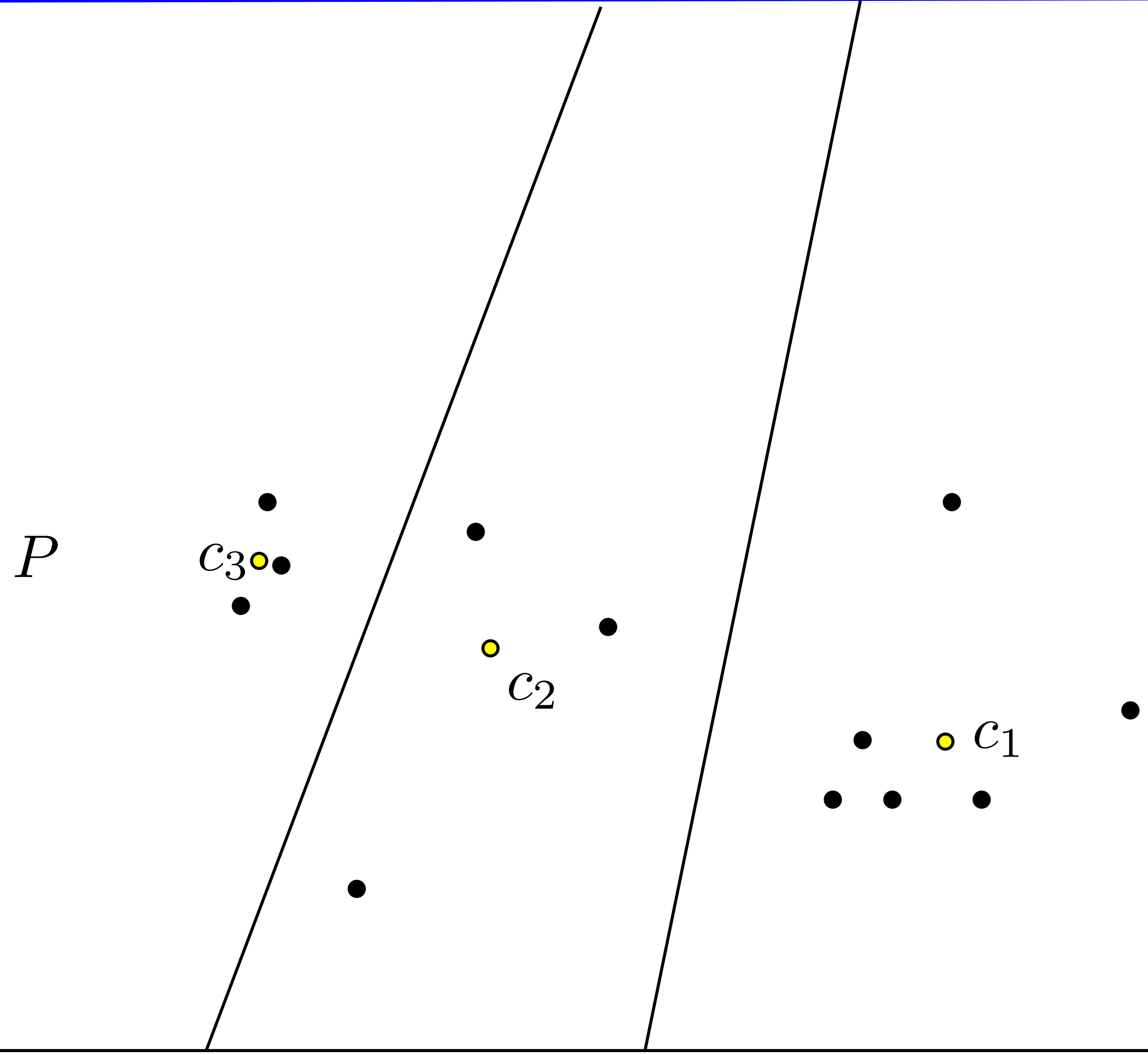
Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm for k-means



Repeat two steps until convergence:
(step 1) update assignment (step 2) update centers

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\sum_{i=1}^{n} \|p_i - c\|^2 = \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\underset{c \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\begin{aligned}
\sum_{i=1}^{n} \|p_i - c\|^2 &= \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} (p_{ij} - \overline{p}_j + \overline{p}_j - c_j)^2
\end{aligned}$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\begin{aligned}
\sum_{i=1}^{n} \|p_i - c\|^2 &= \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} (p_{ij} - \overline{p}_j + \overline{p}_j - c_j)^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} \left((p_{ij} - \overline{p}_j)^2 + 2(p_{ij} - \overline{p}_j)(\overline{p}_j - c_j) + (\overline{p}_j - c_j)^2 \right)
\end{aligned}$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\begin{aligned}
\sum_{i=1}^{n} \|p_i - c\|^2 &= \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} (p_{ij} - \overline{p}_j + \overline{p}_j - c_j)^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} \left((p_{ij} - \overline{p}_j)^2 + 2(p_{ij} - \overline{p}_j)(\overline{p}_j - c_j) + (\overline{p}_j - c_j)^2\right) \\
&= \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + 2 \sum_{i=1}^{n} \langle p_i - \overline{p}, \overline{p} - c \rangle + n \cdot \|\overline{p} - c\|^2
\end{aligned}$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\begin{aligned}
\sum_{i=1}^{n} \|p_i - c\|^2 &= \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} (p_{ij} - \overline{p}_j + \overline{p}_j - c_j)^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} \left((p_{ij} - \overline{p}_j)^2 + 2(p_{ij} - \overline{p}_j)(\overline{p}_j - c_j) + (\overline{p}_j - c_j)^2 \right) \\
&= \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + 2 \sum_{i=1}^{n} \langle p_i - \overline{p}, \overline{p} - c \rangle + n \cdot \|\overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + 2 \left\langle \sum_{i=1}^{n} p_i - n \cdot \overline{p}, \overline{p} - c \right\rangle + n \cdot \|\overline{p} - c\|^2
\end{aligned}$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: Let $\overline{p} := \frac{1}{n} \sum_{i=1}^{n} p_i$

$$\begin{aligned}
\sum_{i=1}^{n} \|p_i - c\|^2 &= \sum_{i=1}^{n} \|p_i - \overline{p} + \overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} (p_{ij} - \overline{p}_j + \overline{p}_j - c_j)^2 \\
&= \sum_{i=1}^{n} \sum_{j=1}^{d} \left((p_{ij} - \overline{p}_j)^2 + 2(p_{ij} - \overline{p}_j)(\overline{p}_j - c_j) + (\overline{p}_j - c_j)^2 \right) \\
&= \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + 2 \sum_{i=1}^{n} \langle p_i - \overline{p}, \overline{p} - c \rangle + n \cdot \|\overline{p} - c\|^2 \\
&= \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + 2 \left\langle \underbrace{\sum_{i=1}^{n} p_i - n \cdot \overline{p}}_{\mathbf{0}}, \overline{p} - c \right\rangle + n \cdot \|\overline{p} - c\|^2
\end{aligned}$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: (continued)

$$\sum_{i=1}^{n} \|p_i - c\|^2 = \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + n \cdot \|\overline{p} - c\|^2$$

Lloyd's algorithm (Analysis for $k = 1$)

Claim:

$$\operatorname*{argmin}_{c \in \mathbb{R}^d} \sum_{p_i \in P} \|p_i - c\|^2 = \frac{1}{n} \sum_{i=1}^{n} p_i$$

Proof: (continued)

$$\sum_{i=1}^{n} \|p_i - c\|^2 = \sum_{i=1}^{n} \|p_i - \overline{p}\|^2 + n \cdot \|\overline{p} - c\|^2$$

right hand side is minimized for $c = \overline{p}$

$\square$

Lloyd's algorithm (Analysis)

Does the algorithm always terminate?

Lloyd's algorithm (Analysis)

Does the algorithm always terminate?

Yes, because:

- each step decreases the cost $\phi(P, C)$
- there exists only a finite number of cluster assignments

Lloyd's algorithm (Analysis)

Does the algorithm always terminate?

Yes, because:

- each step decreases the cost $\phi(P, C)$
- there exists only a finite number of cluster assignments

Does it always converge to an optimal solution?

Lloyd's algorithm (Analysis)

Does the algorithm always terminate?

Yes, because:

- each step decreases the cost $\phi(P, C)$
- there exists only a finite number of cluster assignments

Does it always converge to an optimal solution?

No, can get stuck in local minima.

Lloyd's algorithm (Analysis)

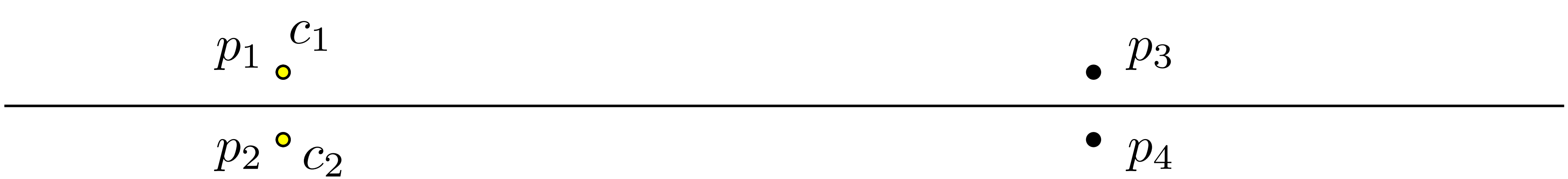
Does the algorithm always terminate?

Yes, because:

- each step decreases the cost $\phi(P, C)$
- there exists only a finite number of cluster assignments

Does it always converge to an optimal solution?

No, can get stuck in local minima.



Lloyd's algorithm (Analysis)

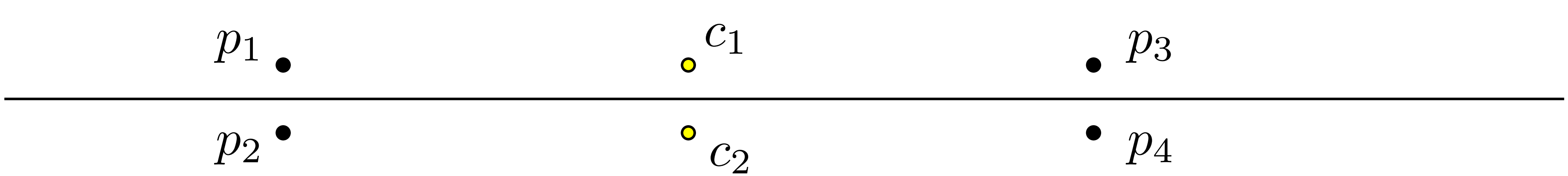
Does the algorithm always terminate?

Yes, because:

- each step decreases the cost $\phi(P, C)$
- there exists only a finite number of cluster assignments

Does it always converge to an optimal solution?

No, can get stuck in local minima.



k-means++ Algorithm

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq \mathbb{R}^2$

Algorithm:

new

- choose c_1 uniformly at random from P
- for $t = 2, \ldots, k$: choose $c_t = p_i$ with probability α_i

$$\alpha_i := \frac{\left\| p_i - \operatorname{argmin}_{c_j \in \{c_1, \ldots, c_{t-1}\}} \|p_i - c_j\| \right\|^2}{\phi(P, \{c_1, \cdots, c_{t-1}\})}$$

k-means++ Algorithm

Input: set of points $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^d$, value of k
Output: set of centers $C = \{c_1, \ldots, c_k\} \subseteq \mathbb{R}^2$

Algorithm:

new

- choose c_1 uniformly at random from P
- for $t = 2, \ldots, k$: choose $c_t = p_i$ with probability α_i

$$\alpha_i := \frac{\left\| p_i - \operatorname{argmin}_{c_j \in \{c_1, \ldots, c_{t-1}\}} \|p_i - c_j\| \right\|^2}{\phi(P, \{c_1, \cdots, c_{t-1}\})}$$

as before

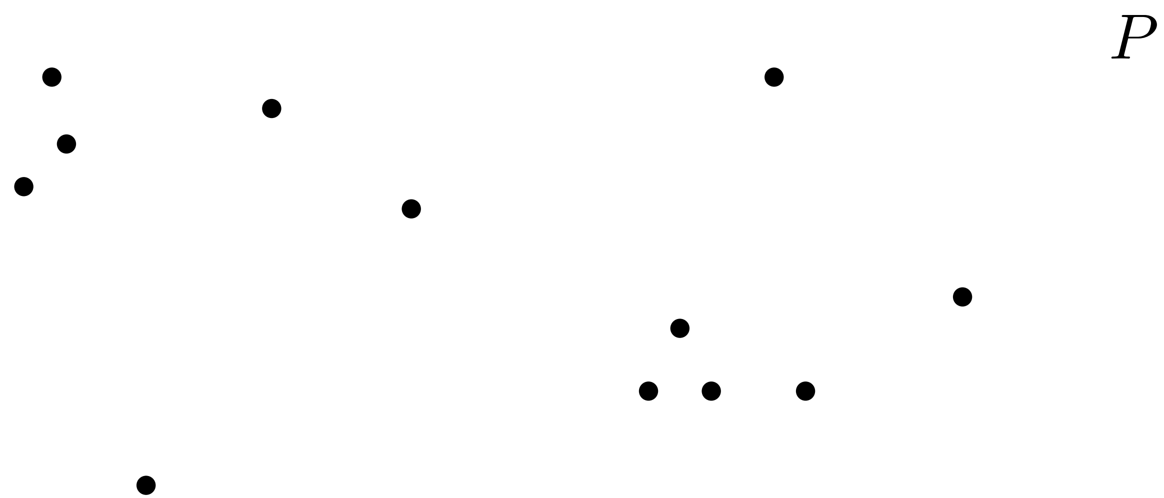
- until $\{c_1, \ldots, c_k\}$ does not change anymore:

(1) assign each $p_i \in P$ to its closest center

(2) update center for each cluster Θ_j:

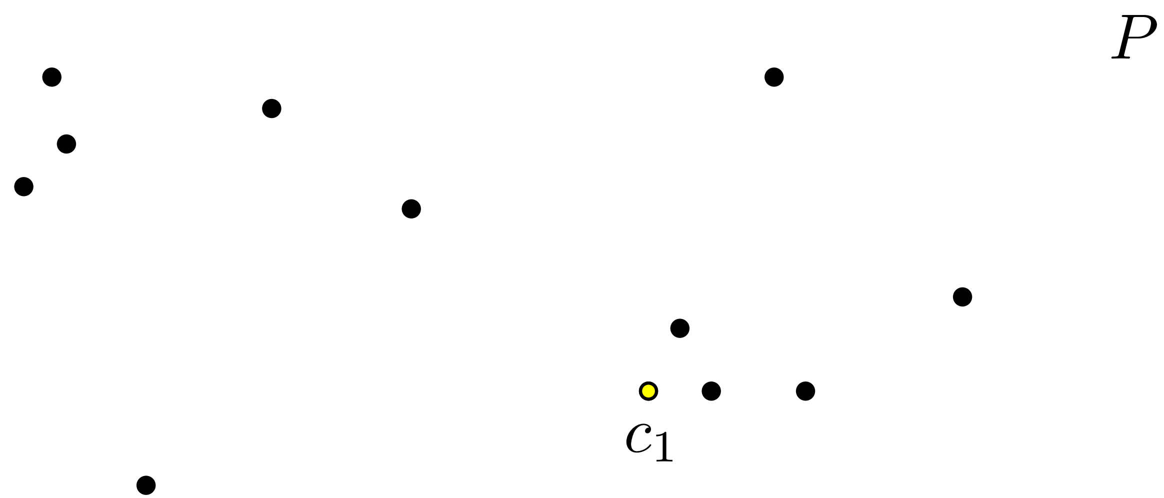
$$c_j := \frac{1}{m} \sum_{p_i \in \Theta_j} p_i$$

k-means++ Algorithm



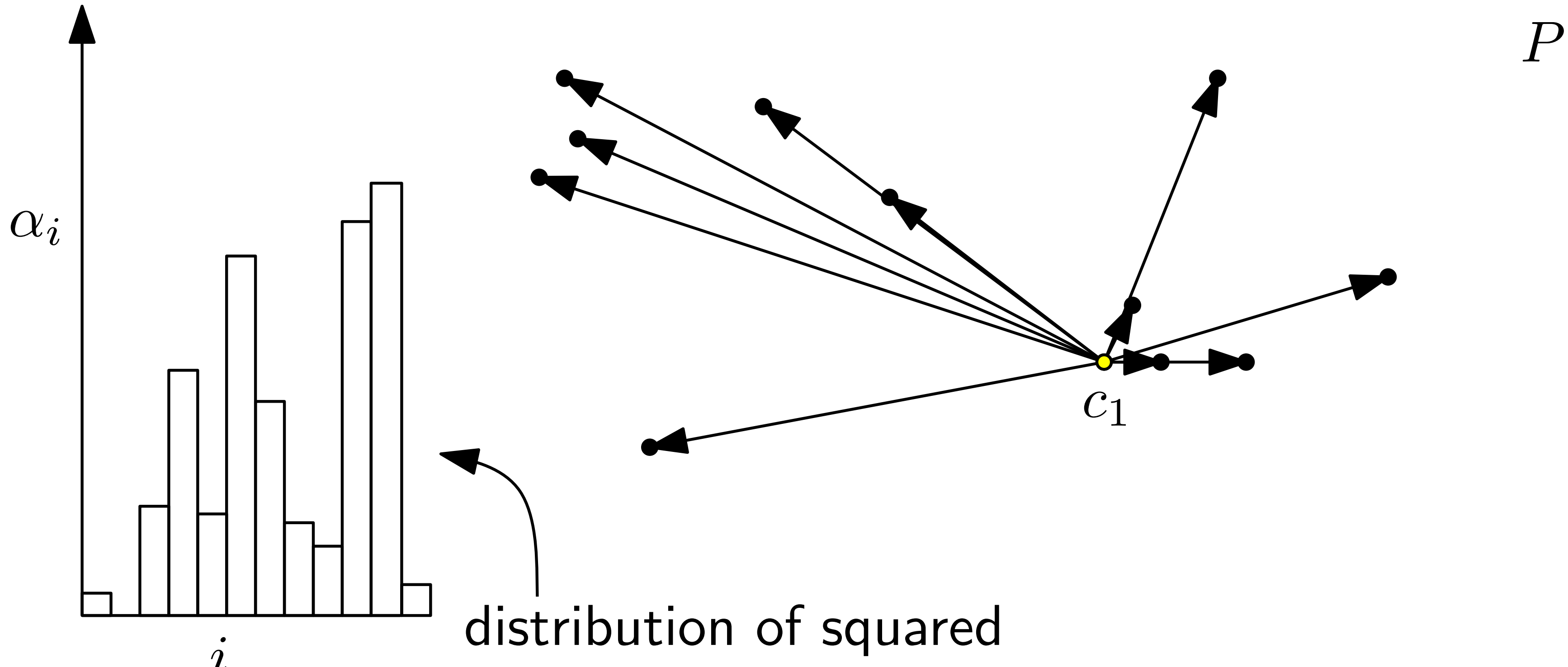
k-means++ Algorithm

new



k-means++ Algorithm

new

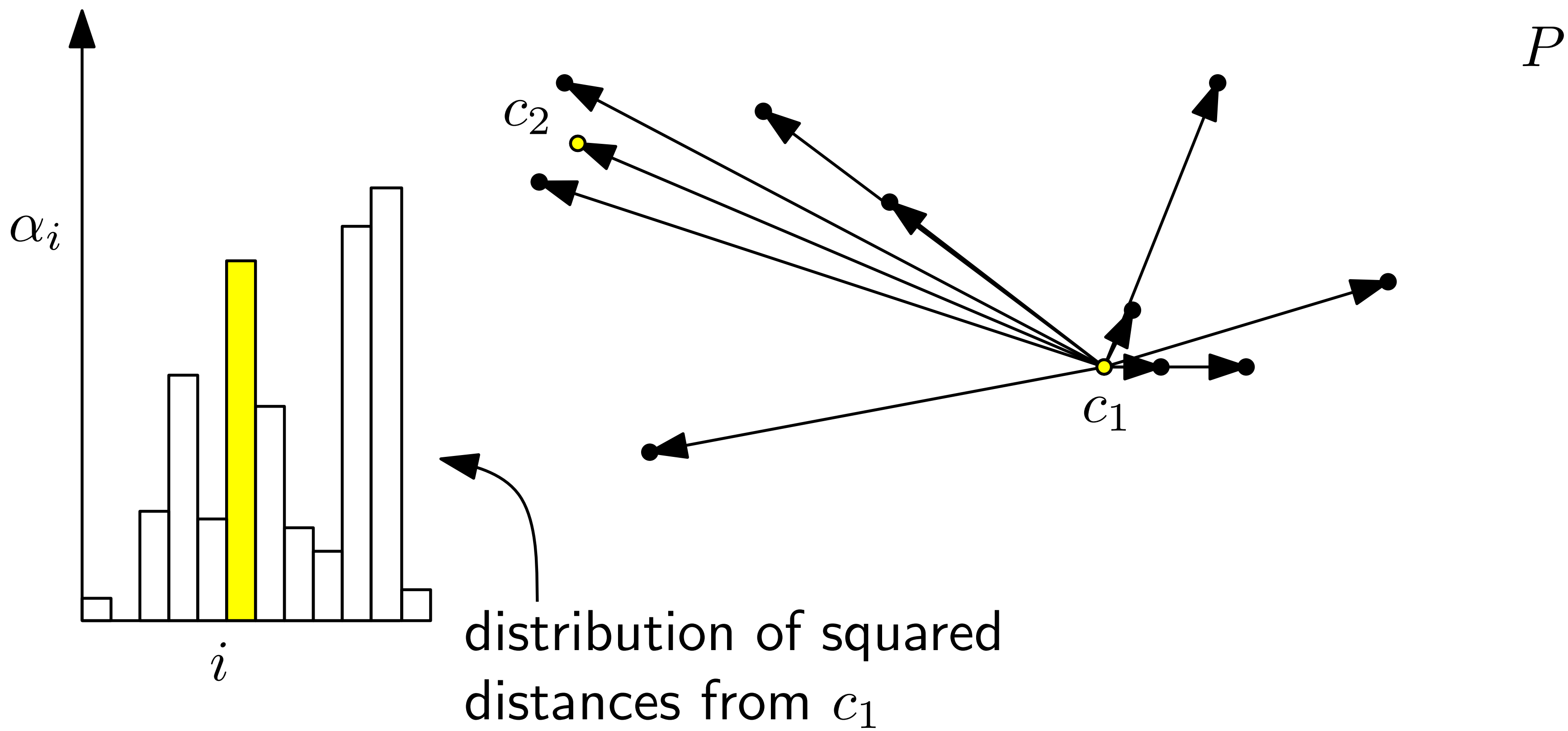


distribution of squared distances from c_1

k-means++ Algorithm

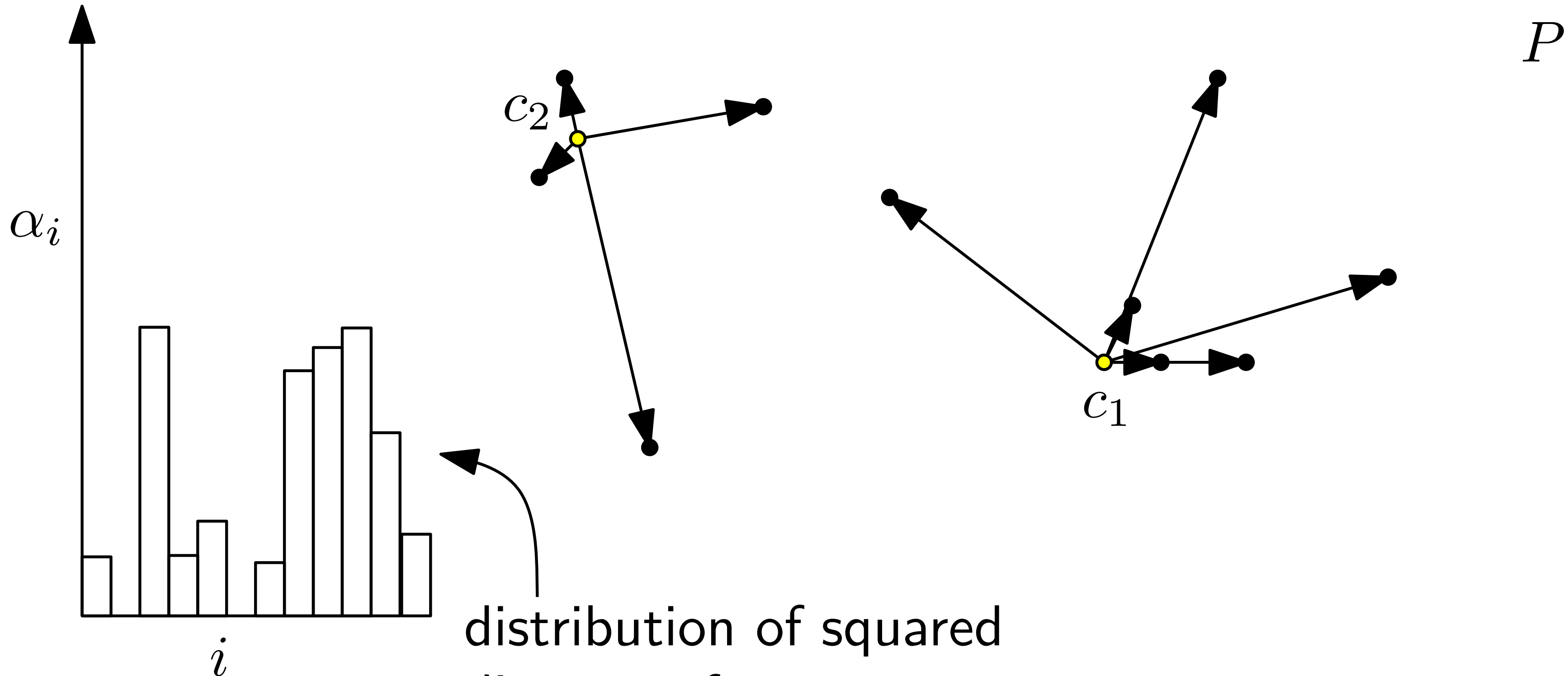
new

random sample



k-means++ Algorithm

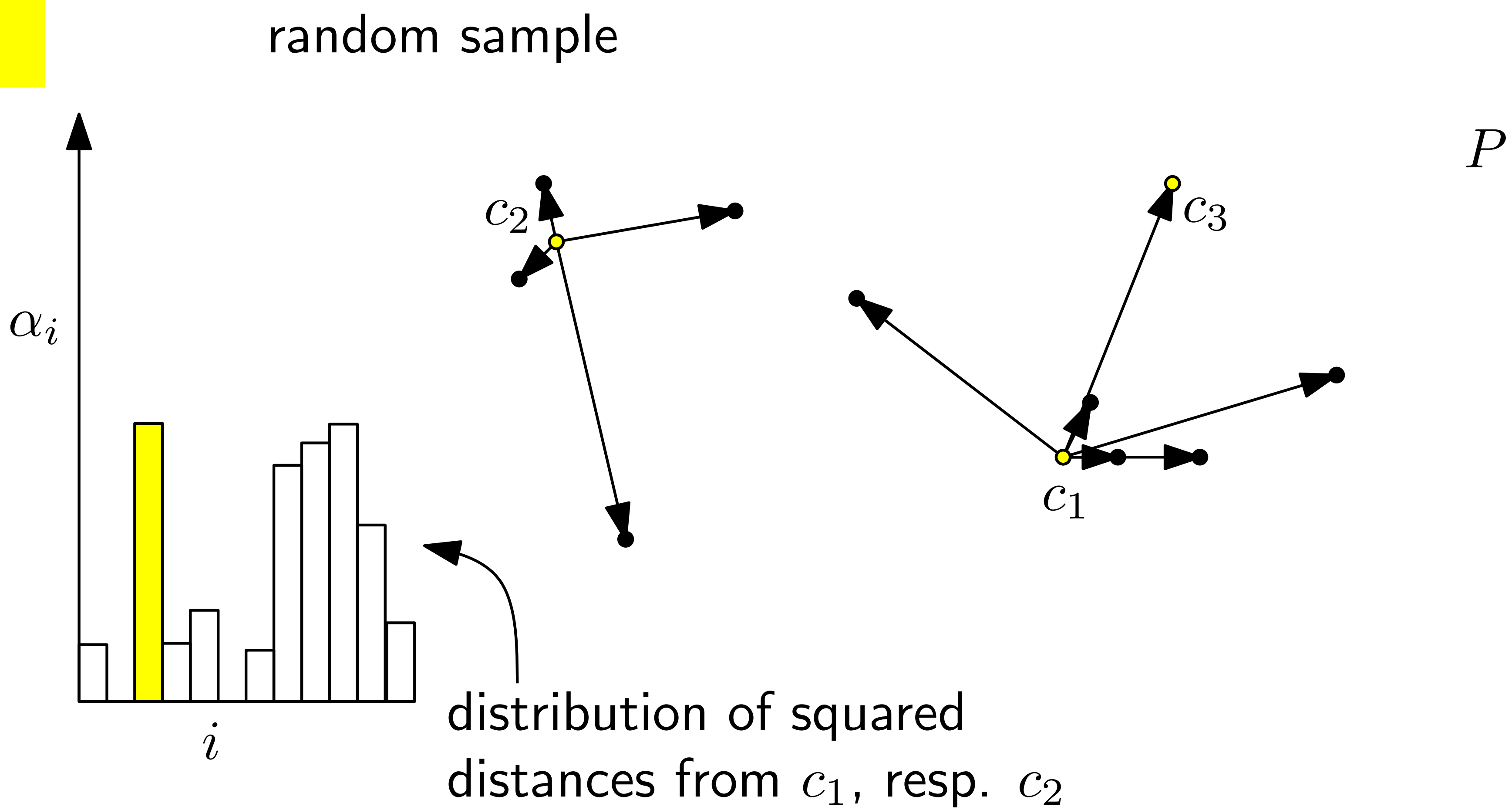
new



distribution of squared
distances from c_1, resp. c_2

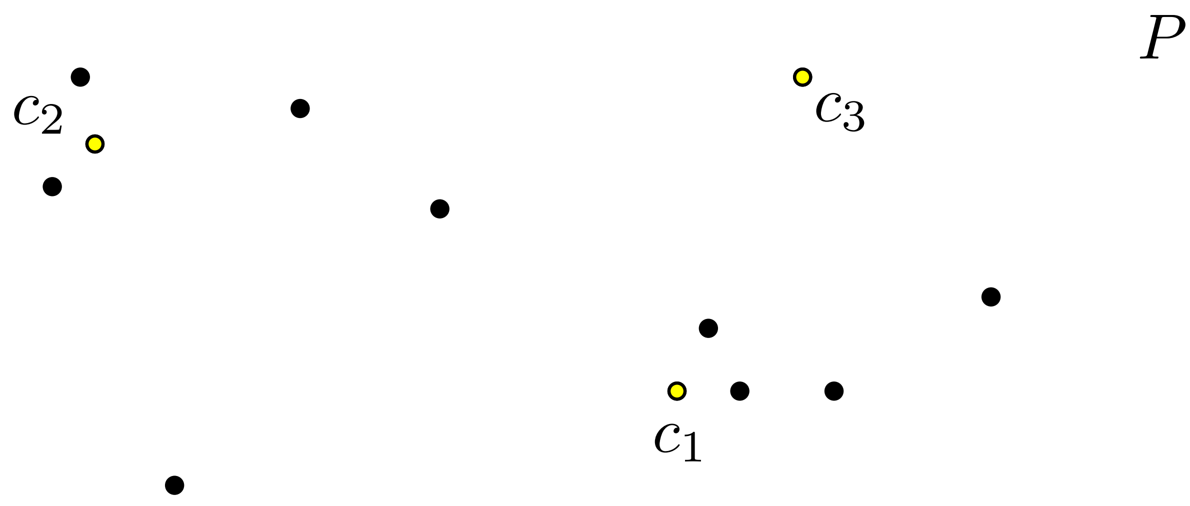
k-means++ Algorithm

new



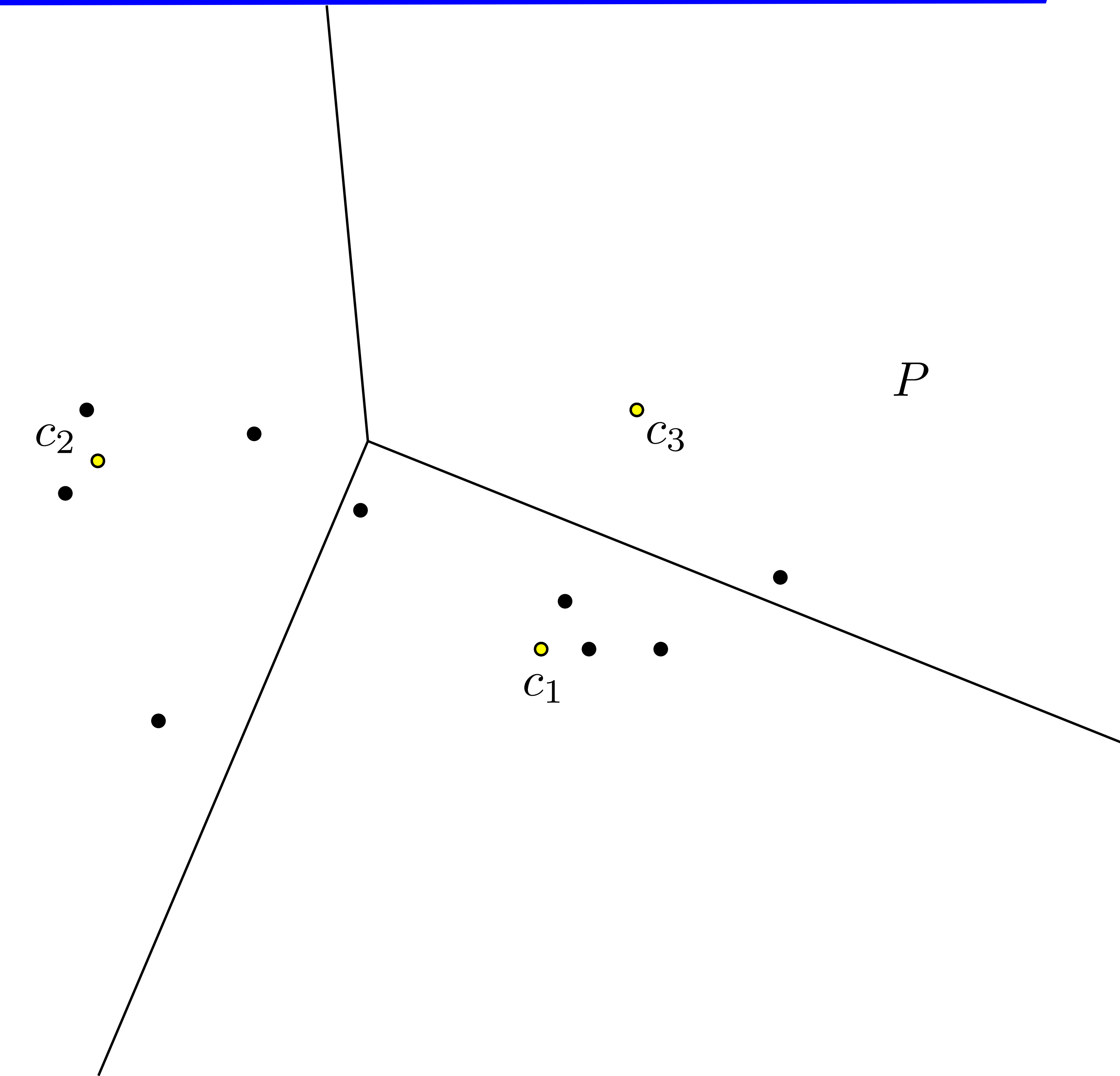
k-means++ Algorithm

as before



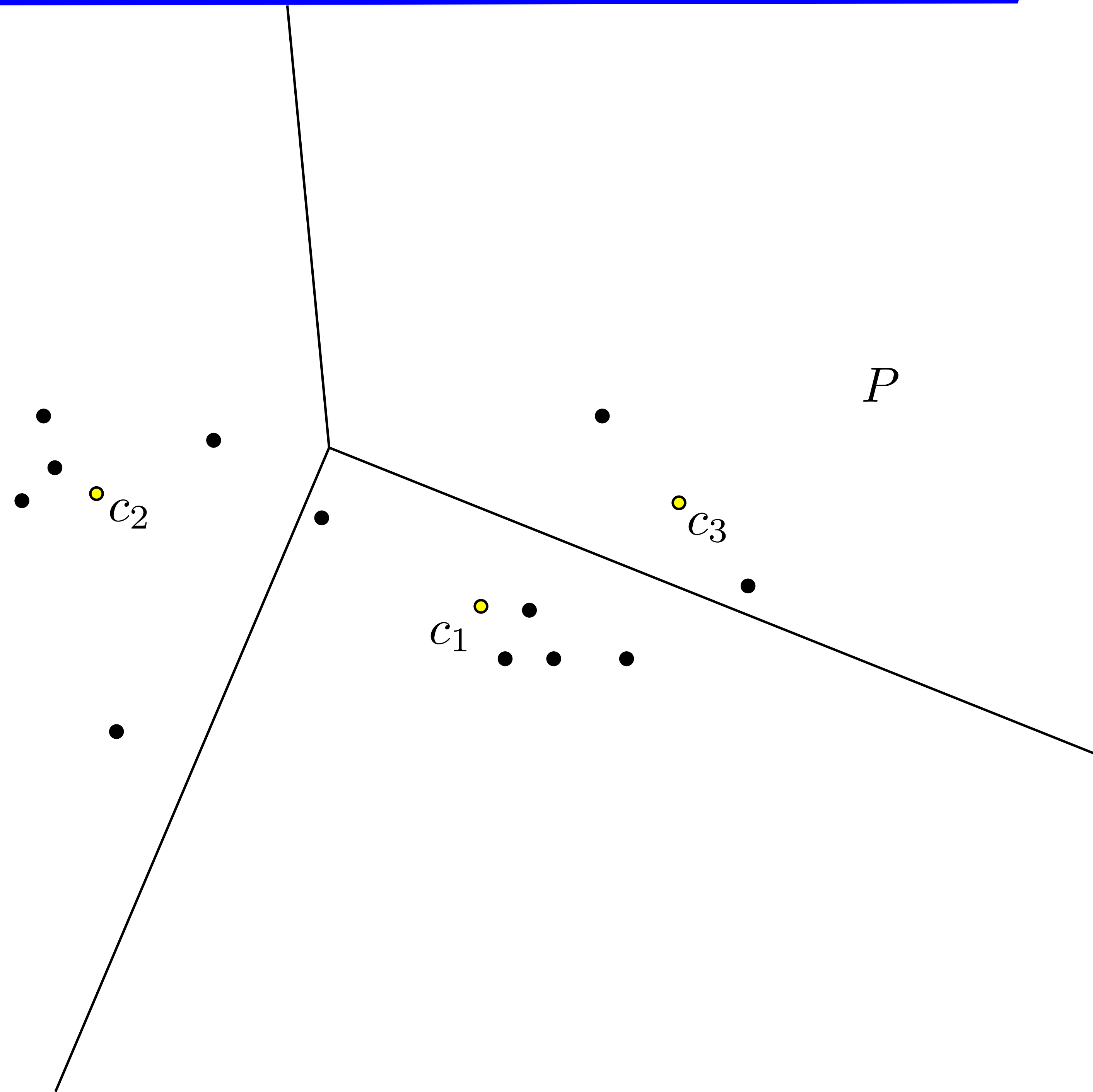
k-means++ Algorithm

as before



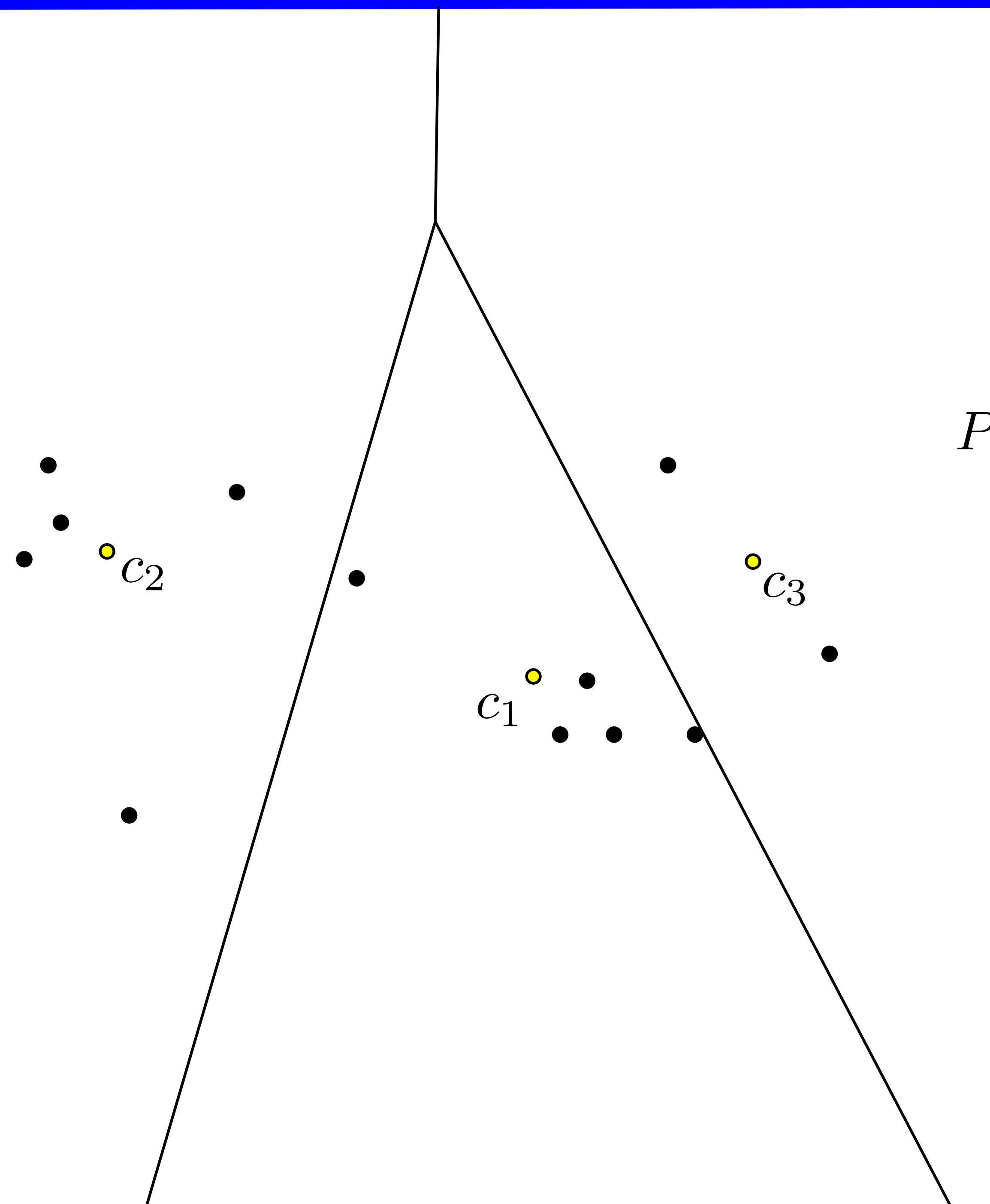
k-means++ Algorithm

as before



k-means++ Algorithm

as before



k-means++ Algorithm

For general k, the solution obtained by k-means++ will, in expectation, be at most a factor $O(\log k)$ worse than the optimal solution.

k-means++ Algorithm

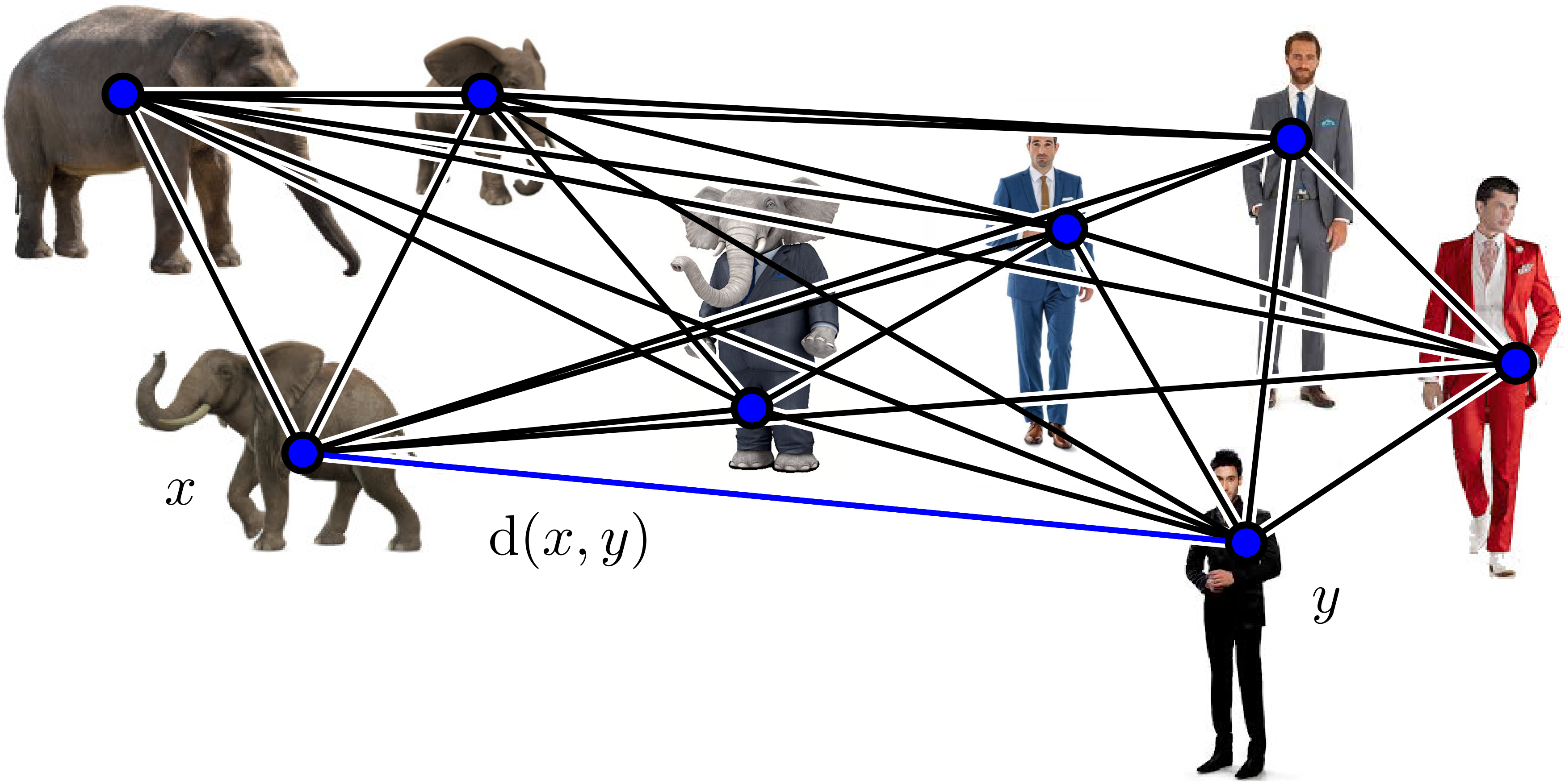
For general k, the solution obtained by k-means++ will, in expectation, be at most a factor $O(\log k)$ worse than the optimal solution.

Gonzales' algorithm and k-means++:

- **Gonzales**: choose the next center from P as the point that maximizes the current cost
- **k-means++**: choose the next center from P with probability relative to the contribution to the current cost

Clustering in Graphs

- Vertices of the graph represent the objects to be clustered
- Distance is measured by shortest path



Summary

- Clustering
- Facility Location
- Gonzales' algorithm
- Lloyd's algorithm (k-means)
- k-means++ algorithm
- Clustering in graphs

References

- Avrim Blum, John Hopcroft, Ravindran Khannan: *Foundations of Data Science*
- Sariel Har-Peled: *Geometric Approximation Algorithms*
- Arthur, D. and Vassilvitskii, S. (2007). "k-means++: the advantages of careful seeding" (PDF). Proc. 18th ACM-SIAM Symposium on Discrete Algorithms. pp. 1027-1035.