Universität Augsburg University

Quantization

Multimedia Computing, Universität Augsburg
Rainer.Lienhart@informatik.uni-augsburg.de
www.multimedia-computing.de

References

1. Brendan J. Frey and Delbert Dueck. Clustering by Passing Messages Between Data Points. Science 315, pp. 972–976, February 2007
2. Delbert Dueck. Affinity Propagation: Clustering Data by Passing Messages. University of Toronto Ph.D. thesis, June 2009.

Quantization

Quantization is the mapping of a set of values to a smaller set of values.

⇨ Loss of information

Examples:

- A/D Conversion
- Lossy Compression (JPEG, MP3)
- Clustering

  := discovering meaningful partitions of data based on a measure of similarity

- General goal: Minimize quantization error

Outline

1. K-Means clustering
2. K-Medoids clustering
3. Affinity propagation

Orderless description

1. 2017_Deep TEN - Texture Encoding Network_CVPR2017
2. Histogramm

„Examplars“ can be found by

1. Randomly choosing an initial subset of data points as exemplars and then iteratively refining it.
   ➔works only if initial choice is close to a good solution
   - K-Means Clustering
   - K-Medoids Clustering

2. Simultaneously considering all data points as potential exemplars through exchanging real-valued messages between data point until a high-quality set of exemplars and corresponding clusters gradually emerge.
   ➔ works – as we will see – better
   - Affinity Propagation

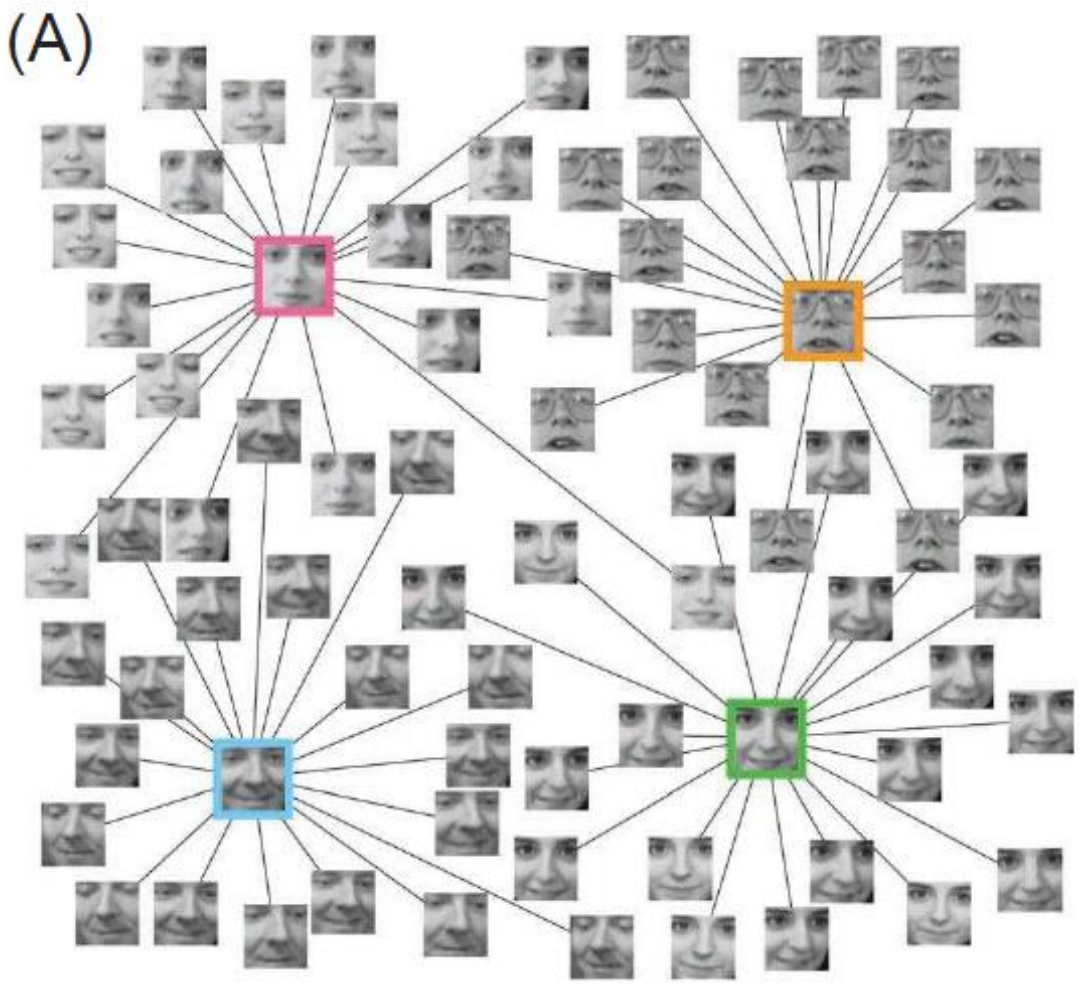
Definitions

- Clustering := discovering meaningful partitions of data based on a measure of similarity
- Exemplar-based clustering := Identify a subset of the N data points as exemplars and assign every other point to one of those exemplars.

The only input are

1. a set of real-valued pairwise similarities between data points $\{s(i,k)\}$ and
2. the number K of exemplars to find or a real-valued exemplar cost to balance against similarities.

Clustering Example



K-Means Clustering

S. Lloyd. Least squares quantization in pcm. IEEE transactions on information theory, 28(2):129–137, 1982. 3

K-Means Clustering - Idea (1)

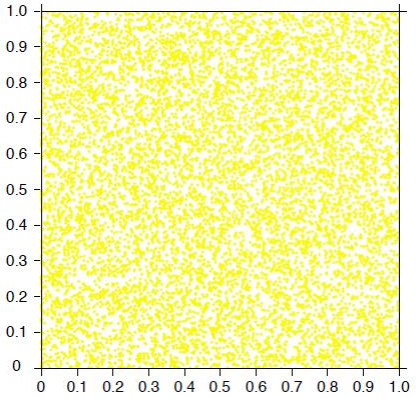
Given N vector data points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}, \mathbf{x}_i \in \mathbb{R}^D$ find K cluster centers $\{\boldsymbol{\mu}_j\}_{j=1}^{K}, \boldsymbol{\mu}_j \in \mathbb{R}^D$ such that the sum of the distance between the data points and their associated closest cluster center is minimized. Each data point is assigned to the closest cluster center.

Assignments are denoted by $Z = \{z_i\}_{i=1}^{N}, z_i \in \{1,2,\dots,K\}$.

$$(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K) = \underset{(\mathbf{c}_1,\mathbf{c}_2,\dots,\mathbf{c}_K),\mathbf{c}_j \in \mathbb{R}^D}{\operatorname{argmin}} \sum_{i=1}^{N} \min_{j \in \{1,2,\dots,K\}} \|\mathbf{x}_i - \mathbf{c}_j\|$$

K-Means Clustering - Idea (2)

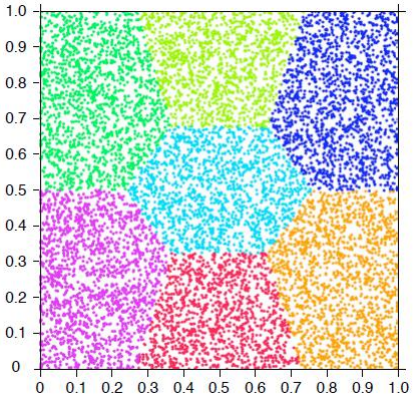
But: Problem is known to be NP hard!



One Solution: Replace global optimal solution by

(1) Starting with randomly chosen cluster centers

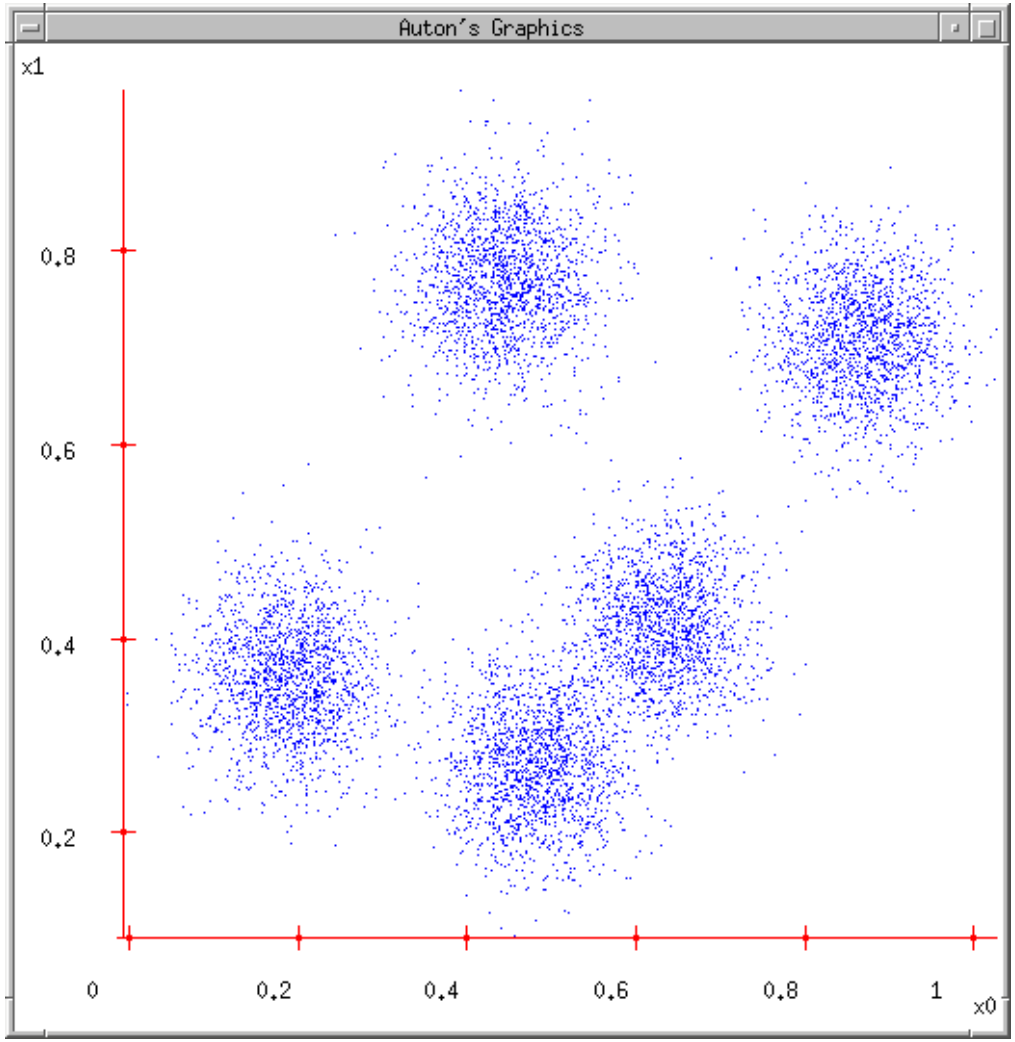
(2) Incrementally & monotonically improve cluster centers by gradient ascent



➔ Local optimum only

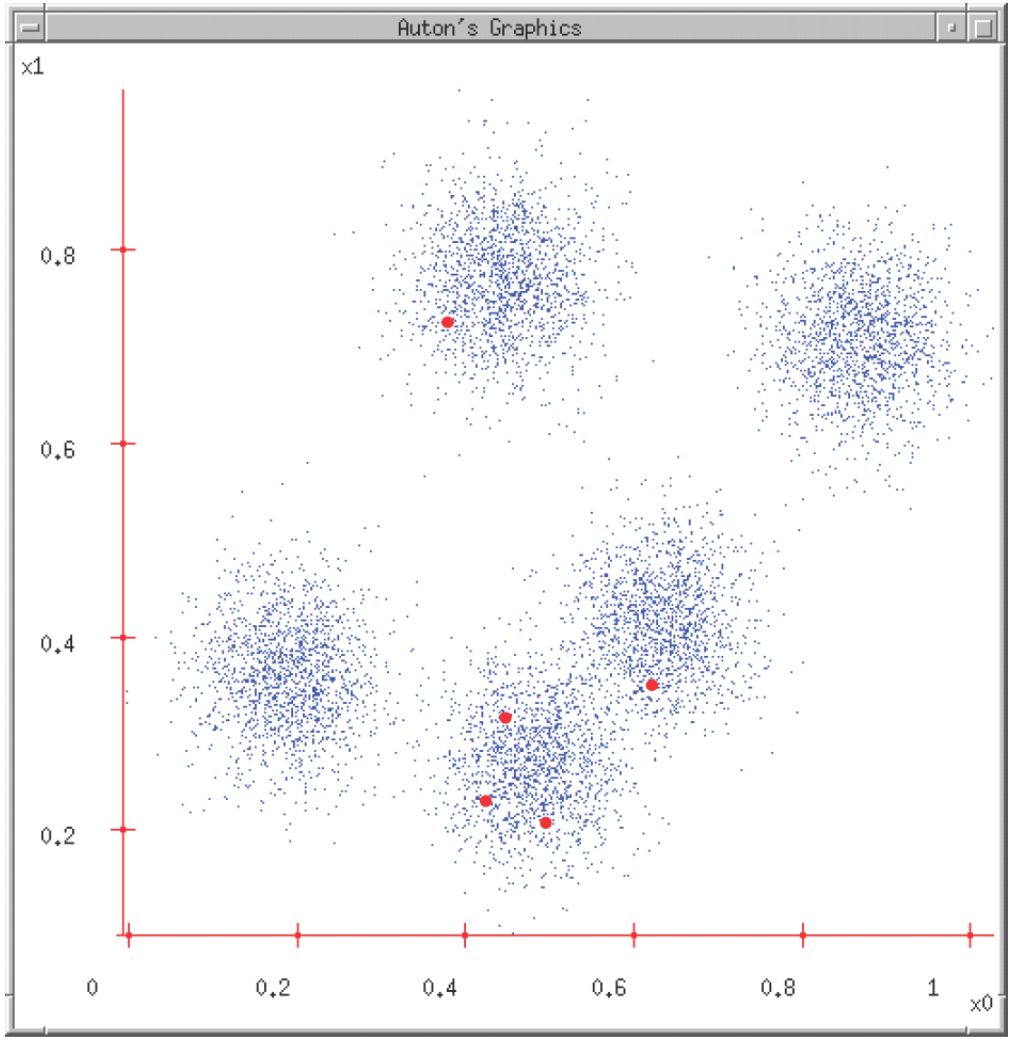
K - Means – Clustering (1)

1. Gebe gewünschte Anzahl der Repräsentanten (Cluster) vor *(e.g., k=5)*.



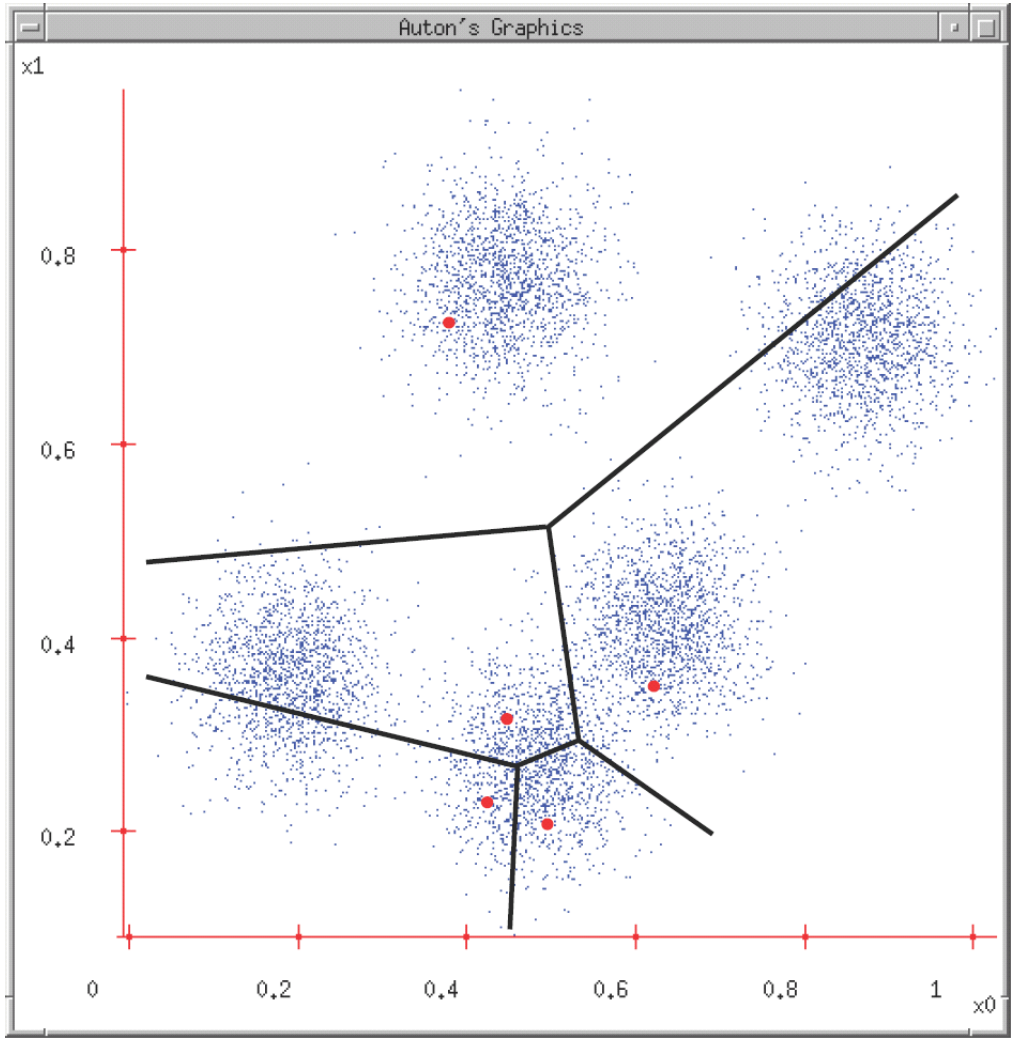
K - Means – Clustering (2)

1. Gebe gewünschte Anzahl der Repräsentanten (Cluster) vor *(e.g., k=5)*.
2. Wähle zufällig k Cluster-Mittelpunkte.



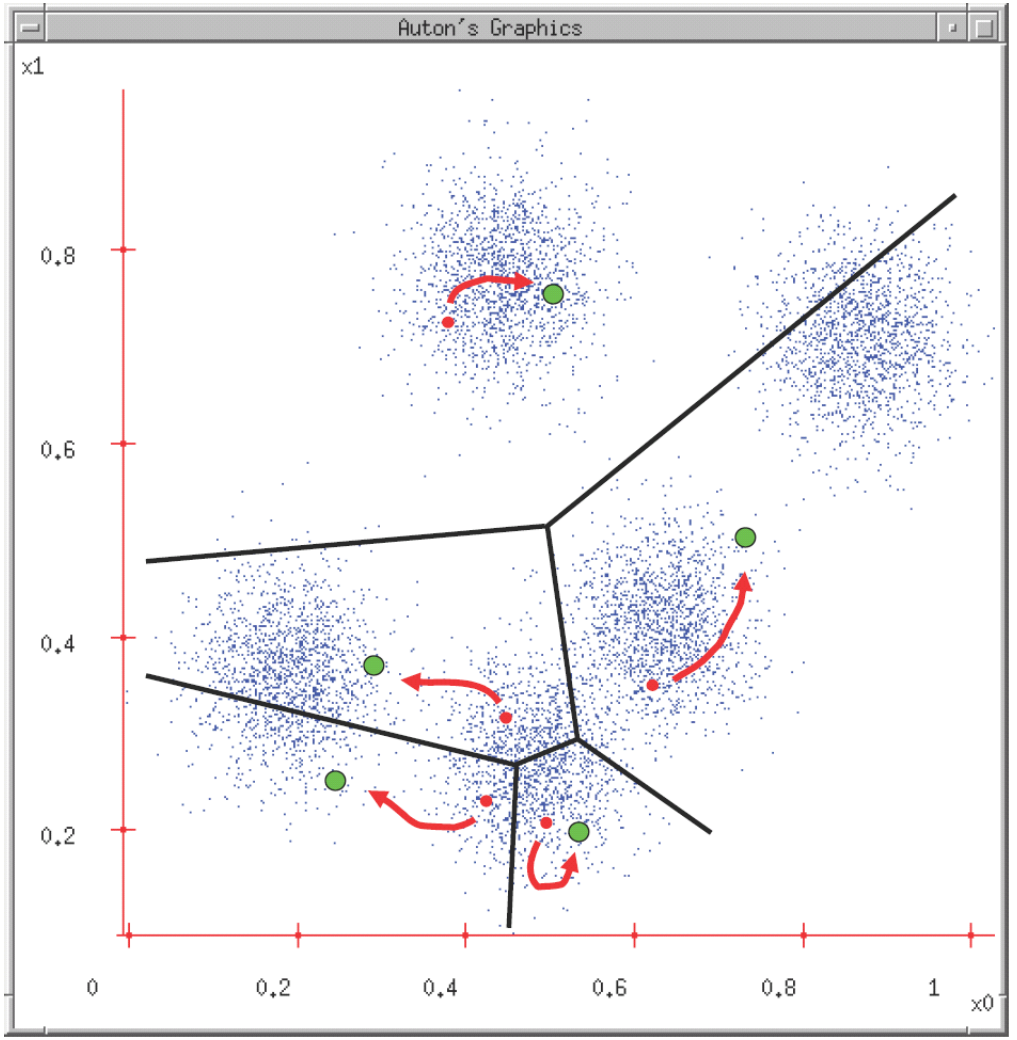
K - Means – Clustering (3)

1. Gebe gewünschte Anzahl der Repräsentanten (Cluster) vor *(e.g., k=5)*.
2. Wähle zufällig k Cluster-Mittelpunkte.
3. Ordne jeden Punkt dem nächsten Clusterpunkt zu.



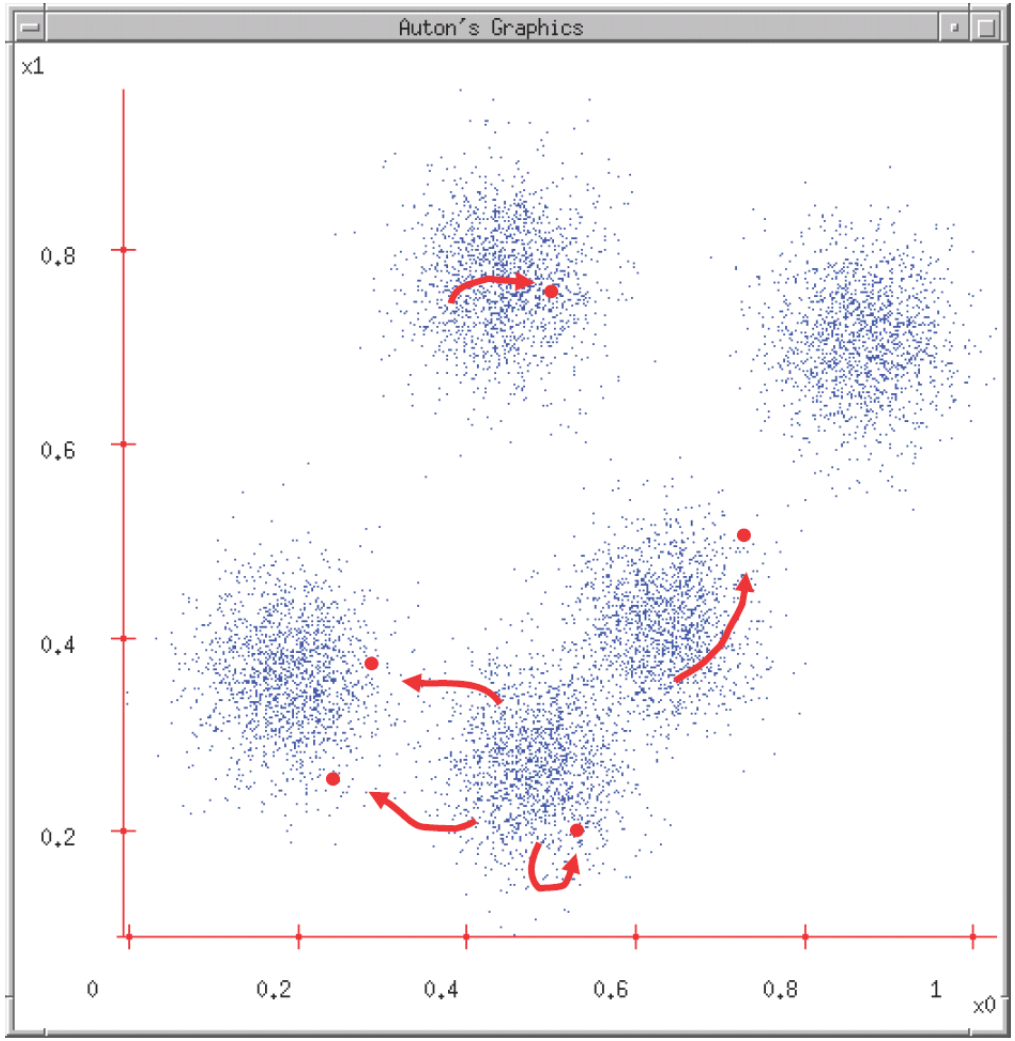
K - Means – Clustering (4)

1. Gebe gewünschte Anzahl der Repräsentanten (Cluster) vor *(e.g., k=5)*.
2. Wähle zufällig k Cluster-Mittelpunkte.
3. Ordne jeden Punkt dem räumlich nächsten Cluster-Mittelpunkt zu.
4. Berechne für jeden Cluster einen neuen Cluster-Mittelpunkt



K - Means – Clustering (4)

1. Gebe gewünschte Anzahl der Repräsentanten (Cluster) vor *(e.g., k=5)*.
2. Wähle zufällig k Cluster-Mittelpunkte.
3. Ordne jeden Punkt dem räumlich nächsten Cluster-Mittelpunkt zu.
4. Berechne für jeden Cluster einen neuen Cluster-Mittelpunkt
5. … und gehe dort hin
6. … und wiederhole ab 3.



K-Means Clustering Algorithm

INPUT: $\{\mathbf{x}_i\}_{i=1}^N, \mathbf{x}_i \in \mathbb{R}^D$ (data), K (number of clusters)

INITIALIZE: Set each μ_k to a random data point

REPEAT UNITL CONVERGENCE:

$$\forall i\colon\ z_i \leftarrow \arg\min_{k\in\{1,\dots,K\}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\| = \arg\max_{k\in\{1,\dots,K\}} \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \mathbf{I}_D)$$

$$\forall k\colon\ \boldsymbol{\mu}_k \leftarrow \text{mean}\{\mathbf{x}_i\}_{i:z_i=k} = \frac{\sum_{i=1}^N [z_i = k]\mathbf{x}_i}{\sum_{i=1}^N [z_i = k]}$$

OUPUT: $\{z_i\}_{i=1}^N$ (cluster assignment),

$\{\boldsymbol{\mu}_k\}_{k=1}^K, \boldsymbol{\mu}_k \in \mathbb{R}^D$ (cluster centers)

Latent (hidden) class variable

[true] =1
[false] = 0

K-Means Clustering Model (1)

The N vector data points $\{\mathbf{x}_i\}_{i=1}^{N}, \mathbf{x}_i \in \mathbb{R}^D$ are created by K multivariate unit-covariance spherical Gaussian processes $\mathcal{N}(\boldsymbol{\mu}_k, \mathbf{I}_D)$ of equal probability:

➔ Gaussian parameters: $\theta = (\boldsymbol{\mu}_1, \mathbf{I}_D, \boldsymbol{\mu}_2, \mathbf{I}_D, \ldots, \boldsymbol{\mu}_K, \mathbf{I}_D)$

➔ Probability of chosing *k*-th Gaussian: $P(z_i = k) = \frac{1}{K}$

➔ Probability of data vector $\mathbf{x}_i$ given parameters θ and hidden variable z_i:

$$P(\mathbf{x}_i | z_i = k, \theta) = \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \mathbf{I}_D) = \frac{1}{\sqrt{(2\pi)^D}} e^{-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T (\mathbf{x}_i - \boldsymbol{\mu}_k)}$$

Using the EM-Algorithm the *K*-means clustering algorithm is the solution.

K-Means Clustering Model (2)

Likelihood of data vector $\boldsymbol{x}_i$ given current parameters:

$$P(\boldsymbol{x}_i|\theta) = \sum_{k=1}^{K} P(\boldsymbol{x}_i, z_i = k|\theta)$$
$$= \sum_{k=1}^{K} P(\boldsymbol{x}_i|z_i = k, \theta) \cdot P(z_i|\theta)$$
$$= \sum_{k=1}^{K} \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{I}_D) \cdot \frac{1}{K}$$

Data Likelihood given current parameters and current cluster assignments:

$$P(\boldsymbol{X}|Z, \theta) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{I}_D)$$

Using the EM-Algorithm the *K*-means clustering algorithm is the solution.

E-Step: Class assignment for data point $\boldsymbol{x}_i$ given current parameters θ:

$$\forall i: \; z_i \leftarrow \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} P(z_i = k|\boldsymbol{x}_i, \theta)$$

M-Step: Maximizing parameters θ given class assignments:

$$\theta \leftarrow \underset{\acute{\theta}}{\operatorname{argmax}} \{P(\boldsymbol{X}|Z, \acute{\theta})\}$$
$$= \underset{\acute{\theta}}{\operatorname{argmax}} \{log[P(\boldsymbol{X}|Z, \acute{\theta})]\}$$
$$= \underset{\acute{\theta}}{\operatorname{argmax}} \left\{ \sum_{i=1}^{N} log[\mathcal{N}(\boldsymbol{x}_i; \acute{\boldsymbol{\mu}}_{z_i}, \boldsymbol{I}_D)] \right\}$$

Derivation

- See chalkboard

EM Algorithm for Mixtures of Gaussians (GM-Model=GMM)

- *K*-means makes all-or-nothing assignments of data points to clusters (hard decisions)
- Learn covariances Σ_k, instead of setting $\forall k\colon \Sigma_k = I$
- Learn mixture weights π_k , instead of setting $\forall k\colon \pi_k = \frac{1}{K}$

EM ALGORITHM FOR A MIXTURE OF GAUSSIANS

INPUT: $\{\mathbf{x}_i\}_{i=1}^N$ (data), K (number of clusters)

INITIALIZE: set $\{\boldsymbol{\mu}_k\}$ to random data points, $\forall k\colon \boldsymbol{\pi_k} \leftarrow \frac{1}{K}$ and $\boldsymbol{\Sigma}_k \leftarrow \mathrm{var}(\{\mathbf{x}_i\})$

REPEAT UNTIL CONVERGENCE:

$$\forall i,k\colon q_{ik} \leftarrow \frac{\pi_k \cdot \mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{k'=1}^K \pi_{k'} \cdot \mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_{k'},\boldsymbol{\Sigma}_{k'})}$$

$$\forall k\colon \pi_k \leftarrow \frac{\sum_{i=1}^N q_{ik}}{N},\ \boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N q_{ik}\mathbf{x}_i}{\sum_{i=1}^N q_{ik}},\ \boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N q_{ik}(\mathbf{x}_i-\boldsymbol{\mu}_k)(\mathbf{x}_i-\boldsymbol{\mu}_k)^\top}{\sum_{i=1}^N q_{ik}} \qquad (2.6)$$

OUTPUT: $\{z_i \leftarrow \underset{k\in\{1,\ldots,K\}}{\mathrm{argmax}}\ q_{ik}\}_{i=1}^N$ (assignments), $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ (Gaussians)

K-Medoids Clustering

K-Medoids Clustering – Idea (1)

Given N vector data points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}, \mathbf{x}_i \in \mathbb{R}^D$ find K exemplars $\{\boldsymbol{m}_i\}_{i=1}^{K}, \boldsymbol{m}_i \in \{\mathbf{x}_i\}_{i=1}^{N}$ such that the sum of the distance between the data points and their associated closest exemplars is minimized. Each data point is assigned to the closest cluster center.

Assignments are denoted by $\{z_i\}_{i=1}^{N}, z_i \in \{1,2,\dots,K\}$.

$$(\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_K) = \underset{(\boldsymbol{m}_1,\boldsymbol{m}_2,\dots,\boldsymbol{m}_K),\boldsymbol{m}_i \in \mathbf{X}}{\arg\min} \sum_{i=1}^{N} \min_{j \in \{1,2,\dots,K\}} \left\| \mathbf{x}_i - \mathbf{m}_j \right\|$$

before it was $\mathbb{R}^D$

K-Medoids Clustering – Idea (2)

But: Problem is known to be NP hard!

One Solution: Replace global optimal solution by

(1) Starting with randomly chosen exemplars

(2) Incrementally & monotonically improve exemplars

➔ Local optimum only

K-Medoids Clustering Algorithm

INPUT: $\{\mathbf{x}_i\}_{i=1}^N, \mathbf{x}_i \in \mathbb{R}^D$ (data), K (number of clusters)

INITIALIZE: Set each $\mathrm{m}_k \in \{1, \ldots, N\}$ to the index of a random data point

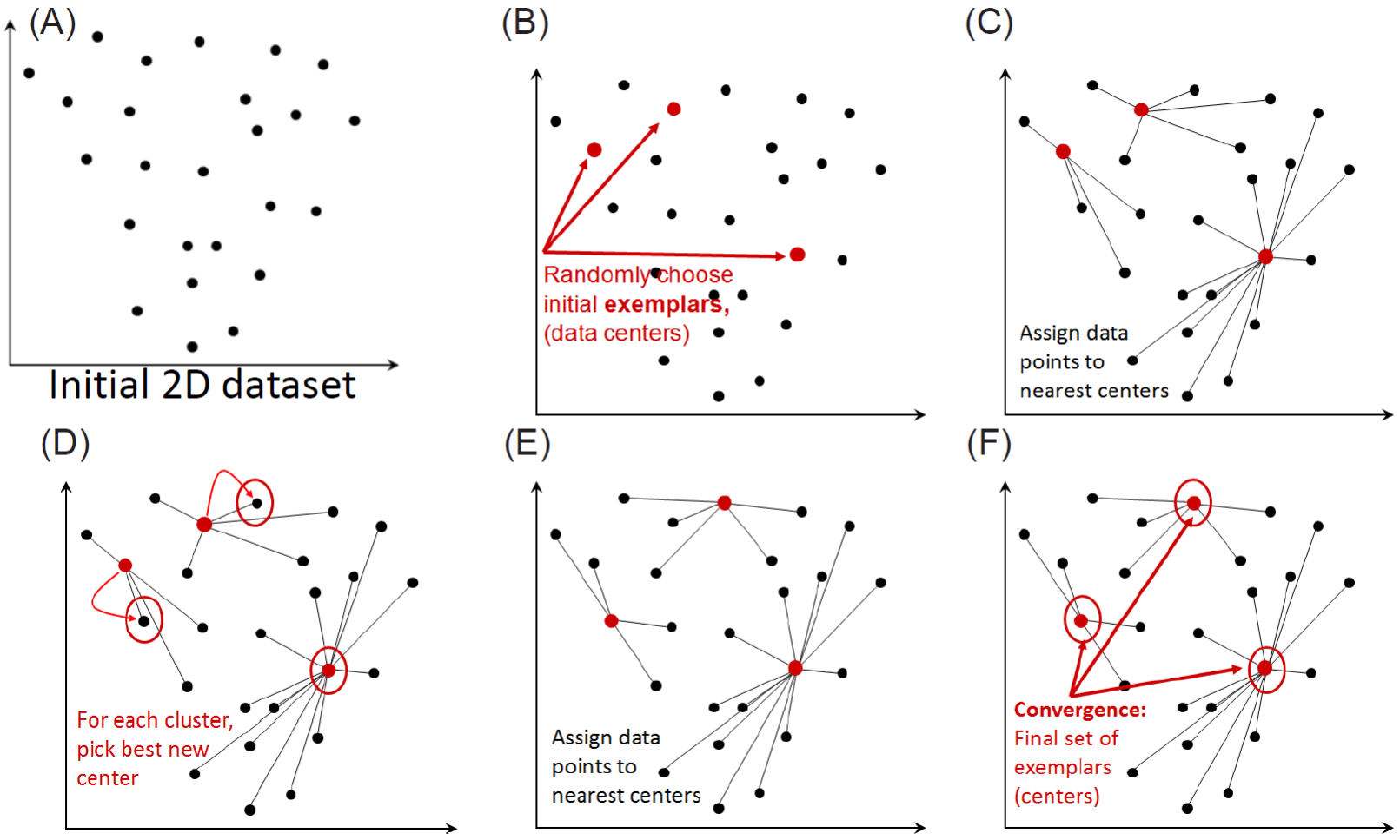
REPEAT UNITL CONVERGENCE:

$$\forall i: \; z_i \leftarrow \arg\min_{k \in \{1,\ldots,K\}} \left\| \mathbf{x}_i - \mathbf{x}_{m_k} \right\|$$

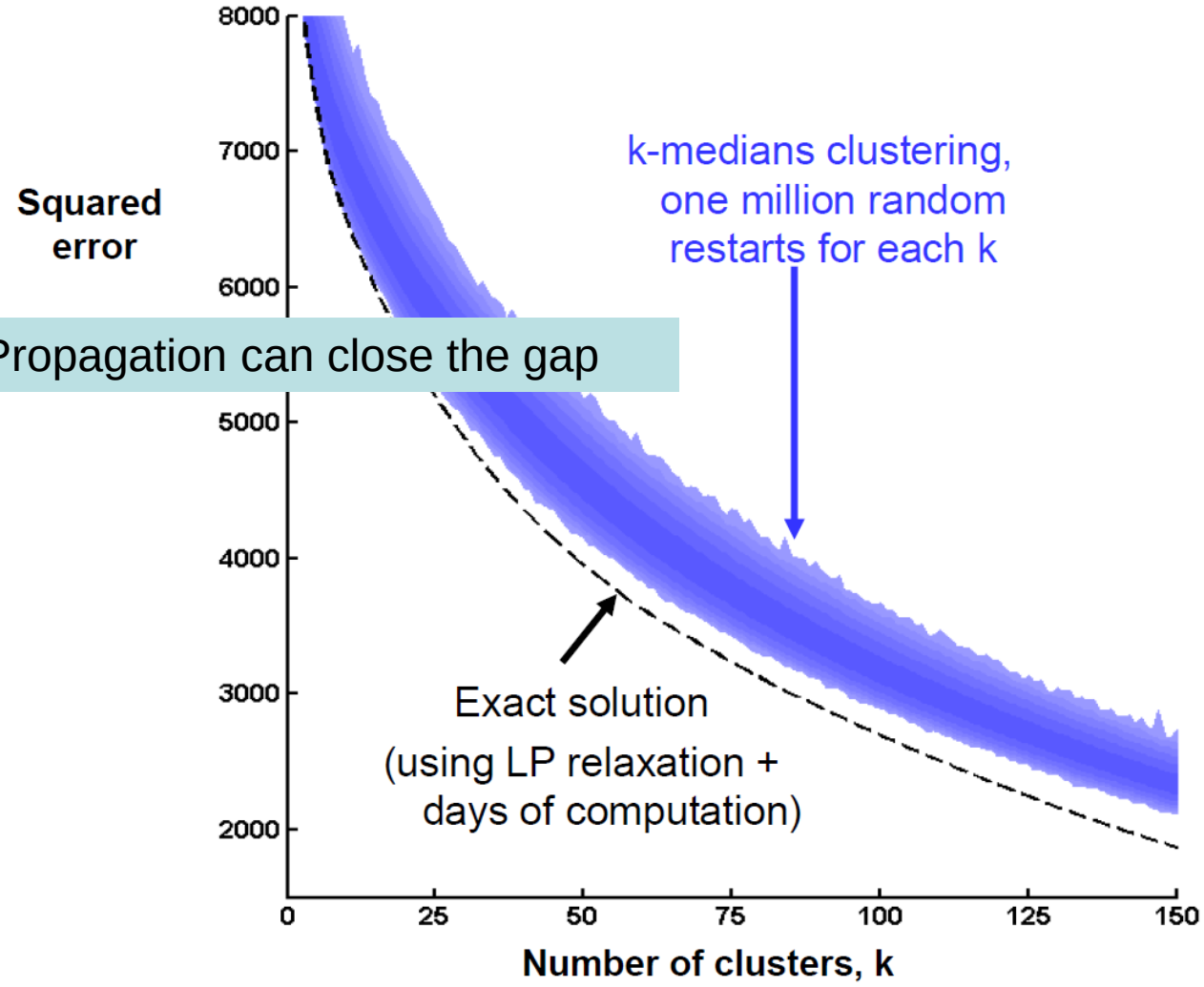
$$\forall k: \; m_k \leftarrow \arg\min_{n:z_n=k} \sum_{l:z_l=k} \|\mathbf{x}_l - \mathbf{x}_n\|$$

OUPUT: $\{z_i\}_{i=1}^N$ (cluster assignments),
$\{m_k\}_{k=1}^K, m_k \in \{1, \ldots, N\}$ (index set of exemplars)

K-Medoids



Squared error achieved by 1 million runs of k-medians clustering on 400 Olivetti face images



Affinity Propagation can close the gap

Universität Augsburg University

Affinity Propagation

Slide takes from Frey

Toronto University

Reference

- Brendan J. Frey and Delbert Dueck. Clustering by Passing Messages Between Data Points. Science 315, pp. 972–976, February 2007

- Code, data software:
  http://www.psi.toronto.edu/affinitypropagation

Input to AP

- A set of pair-wise **similarities** $\{s(i,k)\}$:

  $s(i,k)$ is a real number indicating how well-suited data point k is as an exemplar for point i

  – E.g: $s(i,k) = -\|x_i - x_k\|^2, i \neq k$

  Need not be metric

- For each data point k, a real number $s(k,k)$:

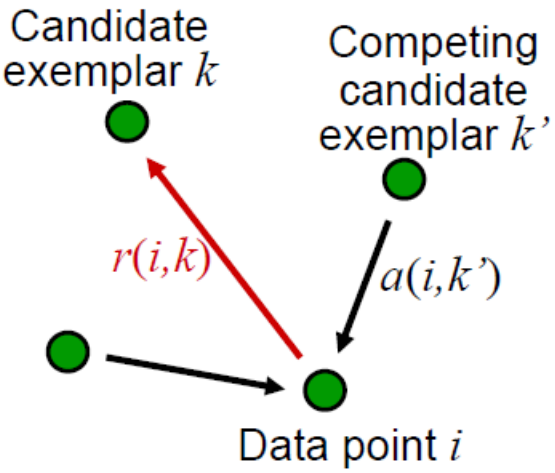
  $s(k,k)$ is a real number indicating the a priori **preference** that it be chosen as an exemplar

  – E.g.: $s(k,k) = median\{s(i,j)\}$

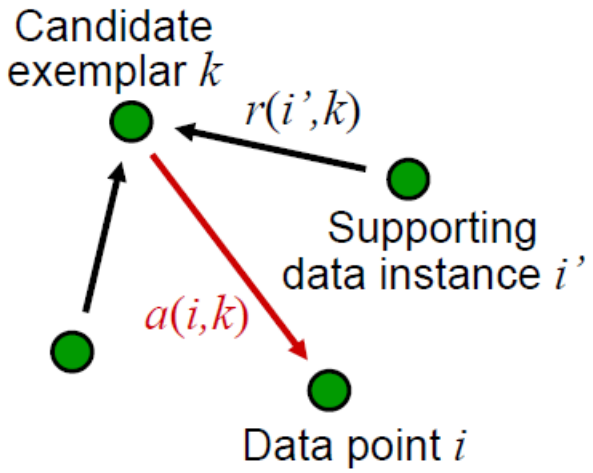
Core Idea (1)

All data points are simultaneously considered as exemplars, but **exchange deterministic messages until** a good set of **exemplars** gradually **emerges**

Sending responsibilities

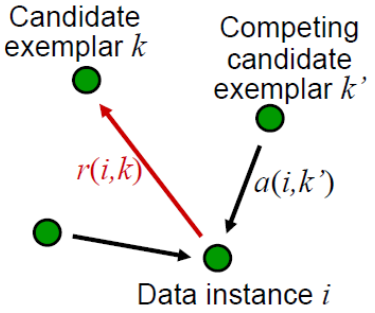


Sending availabilities



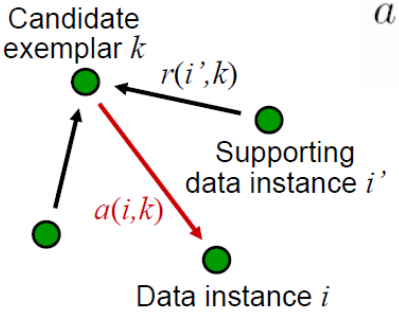
Core Idea (2)

Sending responsibilities



$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\}$$

Sending availabilities



$$a(i,k) \leftarrow \min\Big\{0\ ,\ r(k,k) + \sum_{i' \neq i,k} \max\{0, r(i',k)\}\Big\}$$

$$a(k,k) \leftarrow \sum_{i' \neq k} \max\{0, r(i',k)\}$$

Making decisions:

$$\text{argmax}_k\{a(i,k) + r(i,k)\}$$

AFFINITY PROPAGATION

INPUT: $\{s(i,j)\}_{i,j\in\{1,\ldots,N\}}$ (data similarities and preferences)

INITIALIZE: set 'availabilities' to zero *i.e.* $\forall i,k\colon a(i,k)=0$

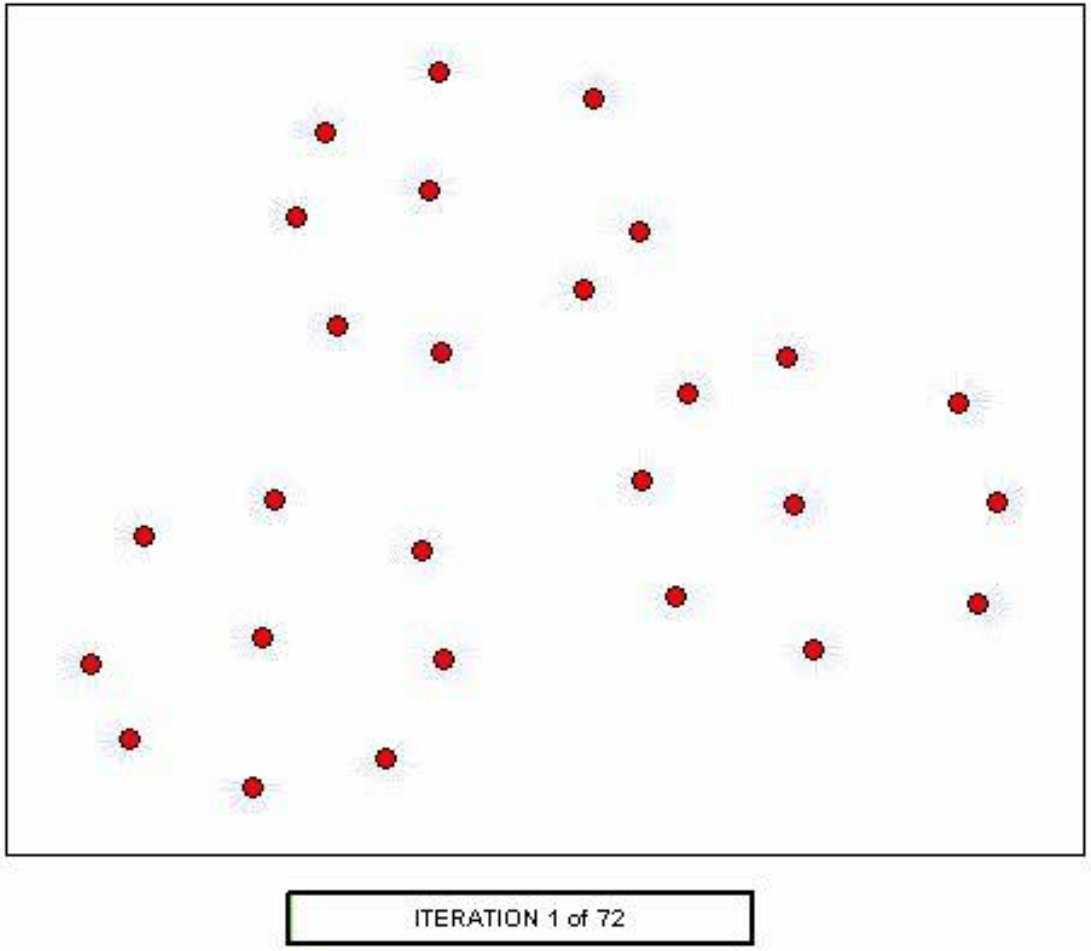
REPEAT: responsibility and availability updates until convergence

$$\forall i,k\colon\ r(i,k)=s(i,k)-\max_{k':k'\neq k}\left[s(i,k')+a(i,k')\right]$$

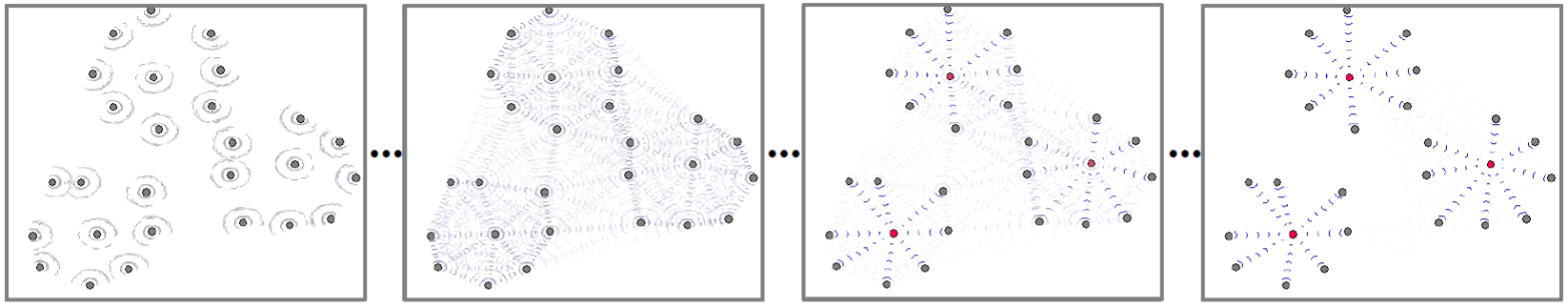
$$\forall i,k\colon\ a(i,k)=\begin{cases}\sum_{i':i'\neq i}\max[0,\ r(i',k)],\ \text{for } k=i\\ \min\left[0,\ r(k,k)+\sum_{i':i'\notin\{i,k\}}\max[0,\ r(i',k)]\right],\ \text{for } k\neq i\end{cases}\qquad(3.15)$$

OUTPUT: cluster assignments $\hat{\mathbf{c}}=(\hat{c}_1,\ldots,\hat{c}_N)$, $\hat{c}_i=\text{argmax}_k\left[a(i,k)+r(i,k)\right]$

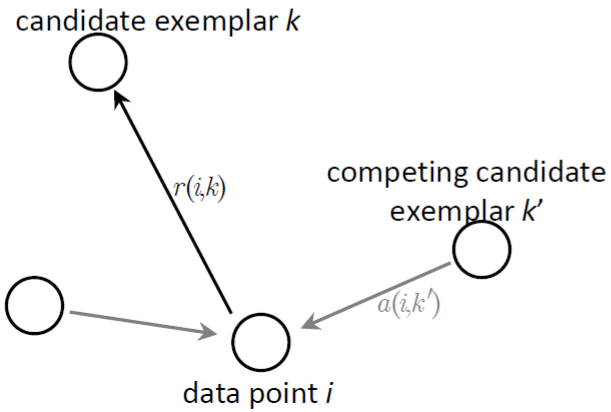
Video



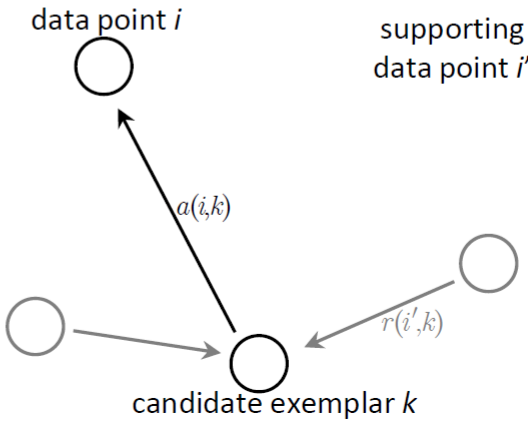
(A)



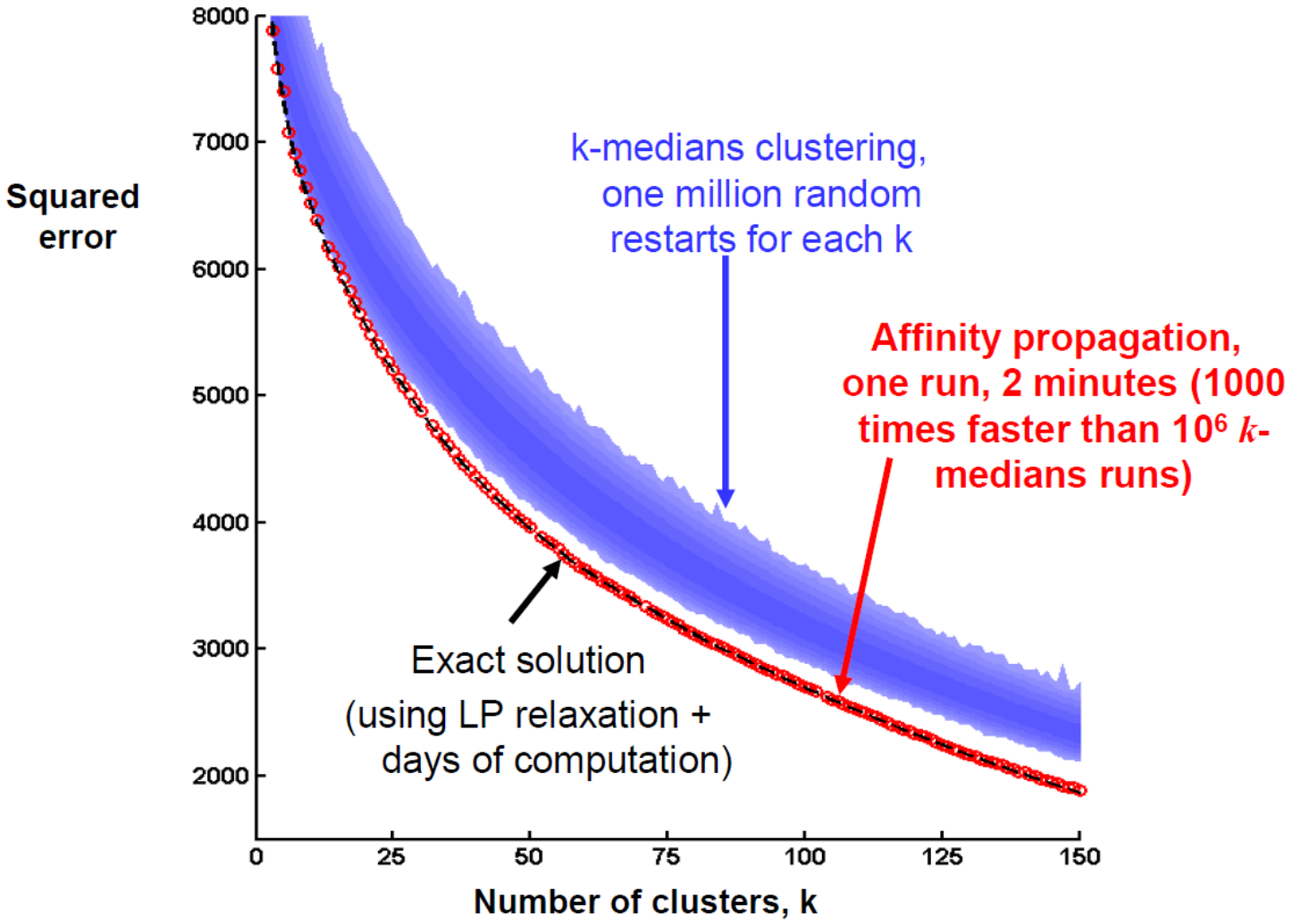
(B)



(C)



Squared error achieved by affinity propagation on 400 Olivetti face images



Affinity Propagation can close the gap

Example: Unsupervised Categorization

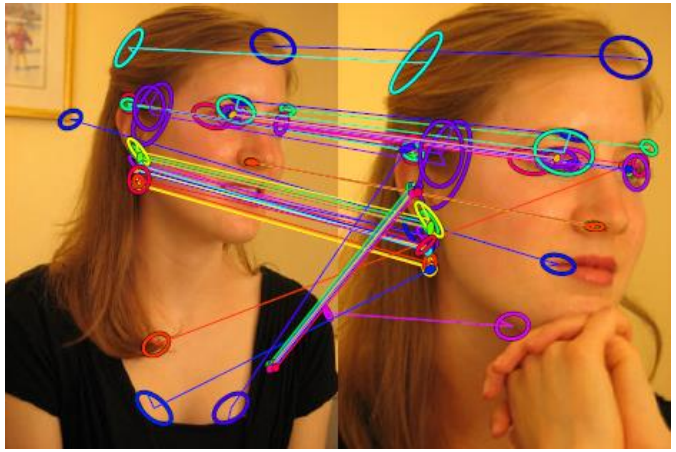
D. Dueck, B. J. Frey. Nonmetric affinity propagation for unsupervised image categorization. Proceedings IEEE International Conference on Computer Vision, January 2007. IEEE Computer Society Press, Los Alamitos, CA, 8 pages.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.3411&rep=rep1&type=pdf

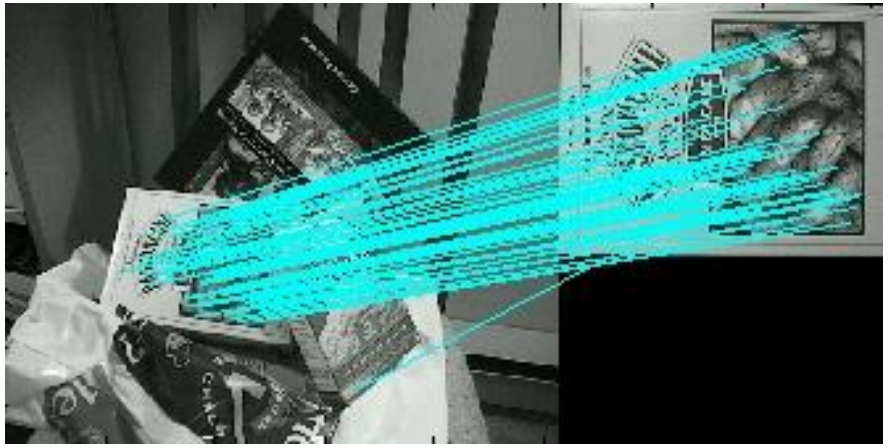
Given: Caltech 101

- 8677 images of objects from 101 categories

Features for estimating similarity between images:

- Sparse SIFT (according to Lowe)
- Compute similarity between all possible image pairings





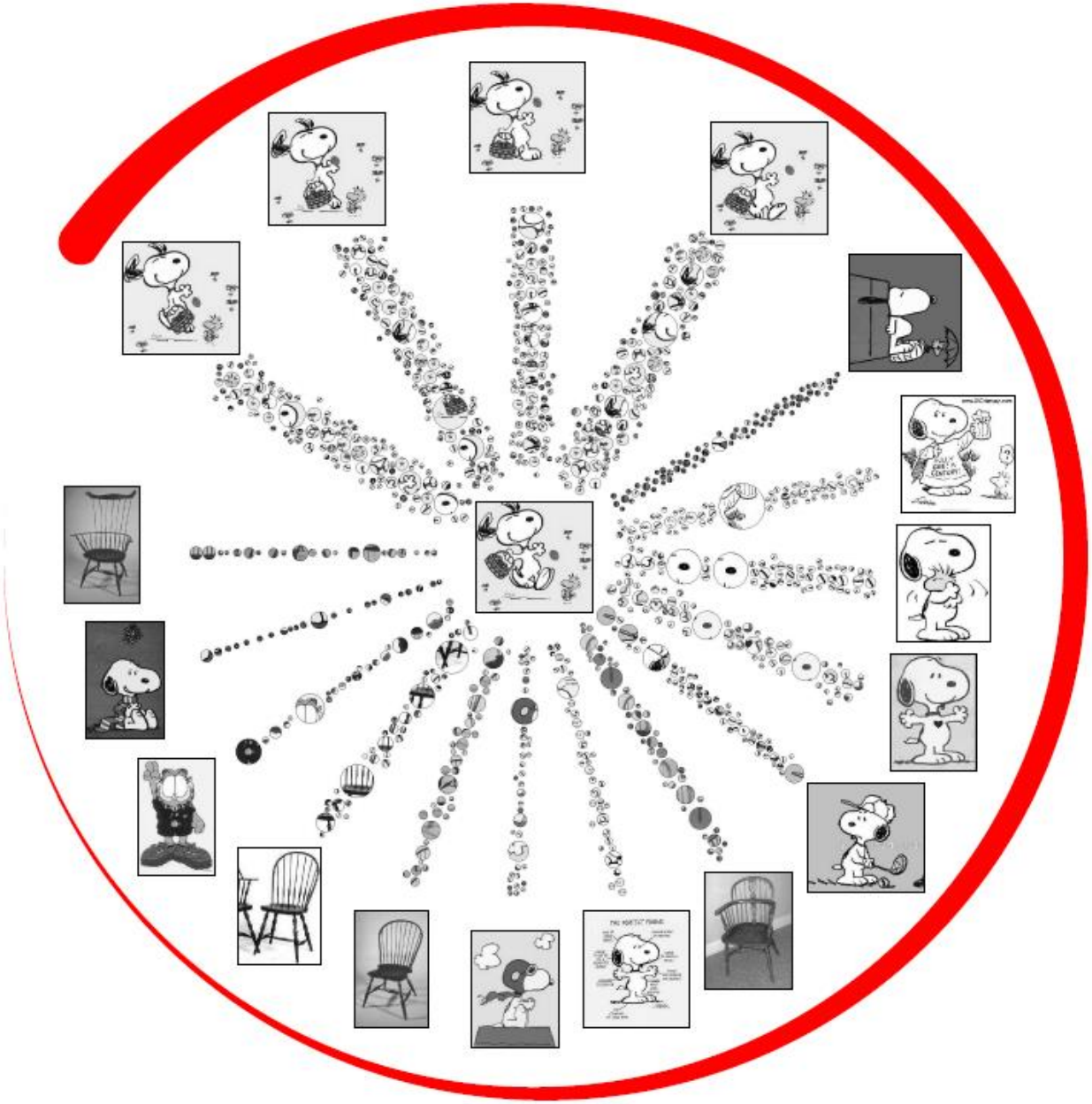


Figure 5. A sample category of images learned by affinity propagation. The central "Snoopy" image is the exemplar; the 17 other category members are shown with a stream of SIFT features matching the exemplar leading up to them. The thickness of the red "swoop" around the perimeter indicates the relevance of each image to this category as measured by the rank of its normalized similarity. Notice that the shared SIFT features for the three (weakly) misclassified chair images contain pieces that look similar to Snoopy's basket.

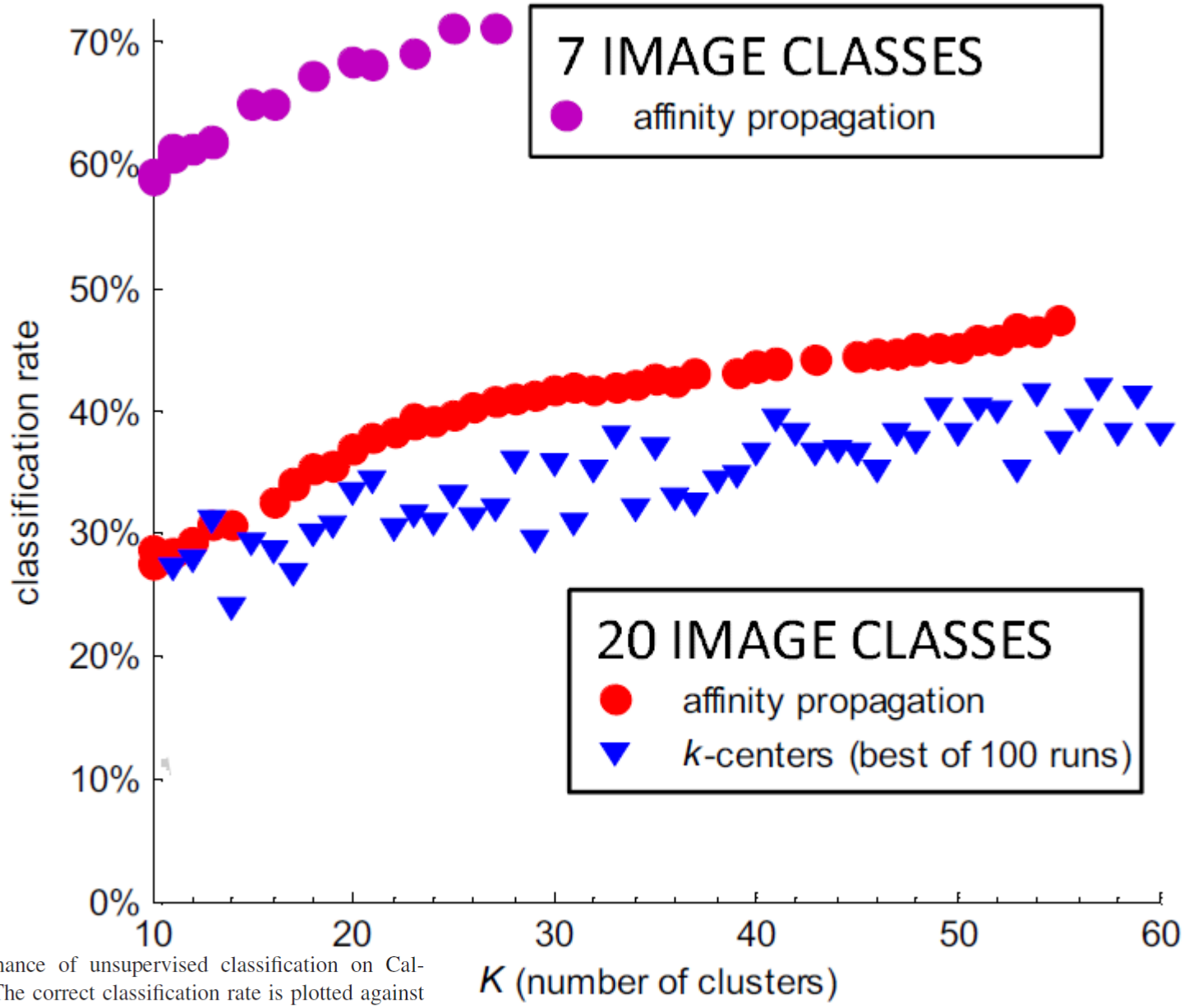


Figure 6. Performance of unsupervised classification on Caltech101 images. The correct classification rate is plotted against the number of categories learned (clusters). Affinity propagation achieves consistently better classification rates than the best of 100 k-centers runs for a dataset of 20 image classes. Affinity propagation's classification rate is also shown for a different subset of the data containing 7 image classes.