Lecture 20
Mid-level Vision: Unsupervised Clustering

ECEN5283 Computer Vision

Dr. Guoliang Fan

Oklahoma State University

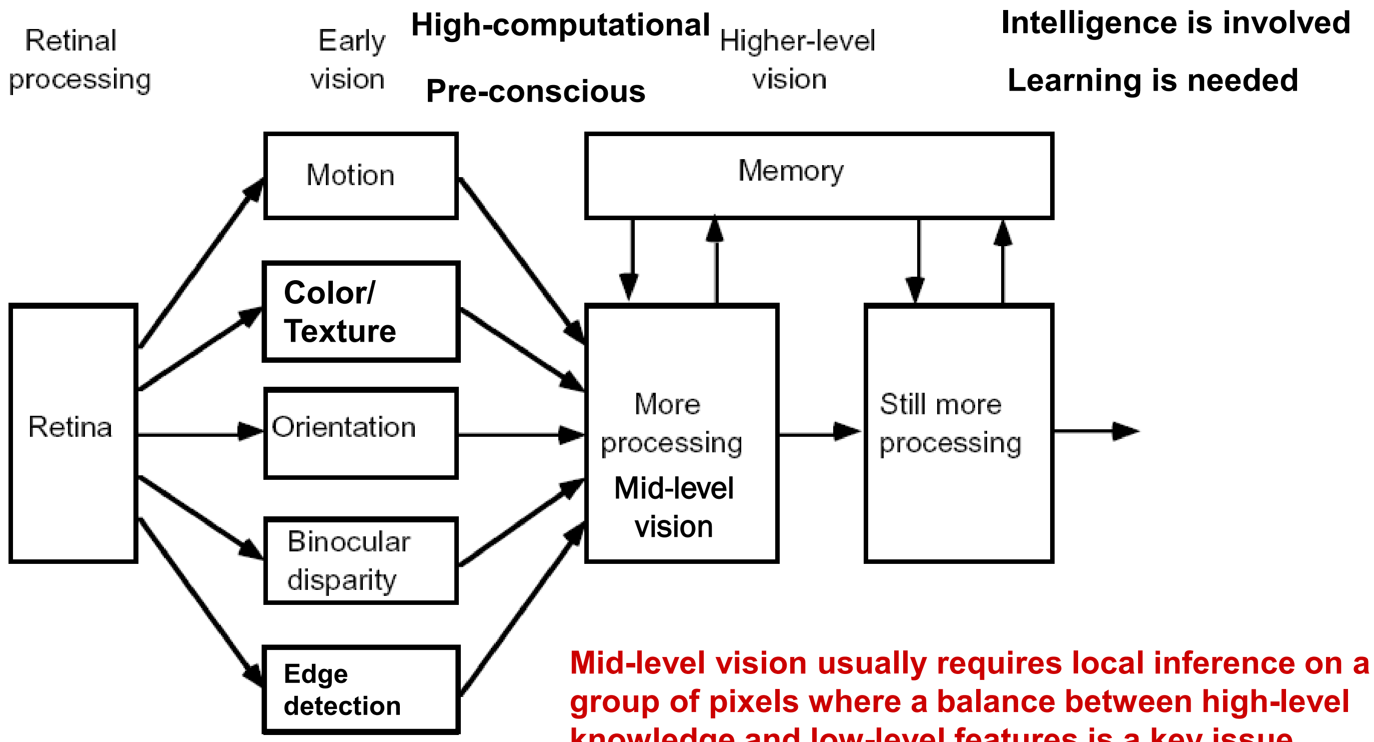
Goals

To introduce mid-level vision.

To study a few issues related to unsupervised clustering.

To introduce the K-means clustering method.

Parallel pathways in early vision



Mid-level vision usually requires local inference on a group of pixels where a balance between high-level knowledge and low-level features is a key issue.

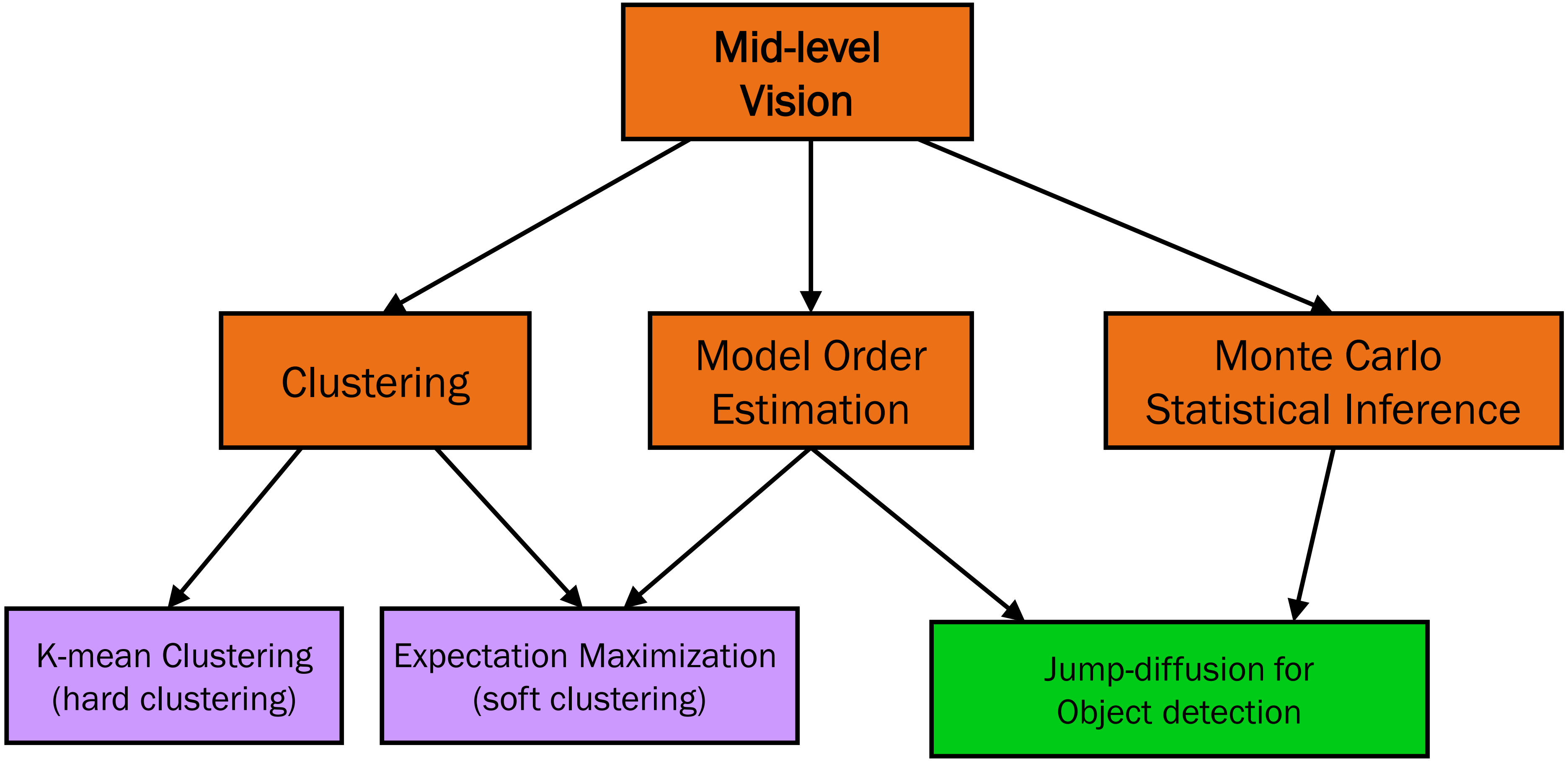
Fig.1.1
A generic diagram for visual processing. In this approach, early vision consists of a set of parallel pathways, each analyzing some particular aspect of the visual stimulus.

What is mid-level vision?

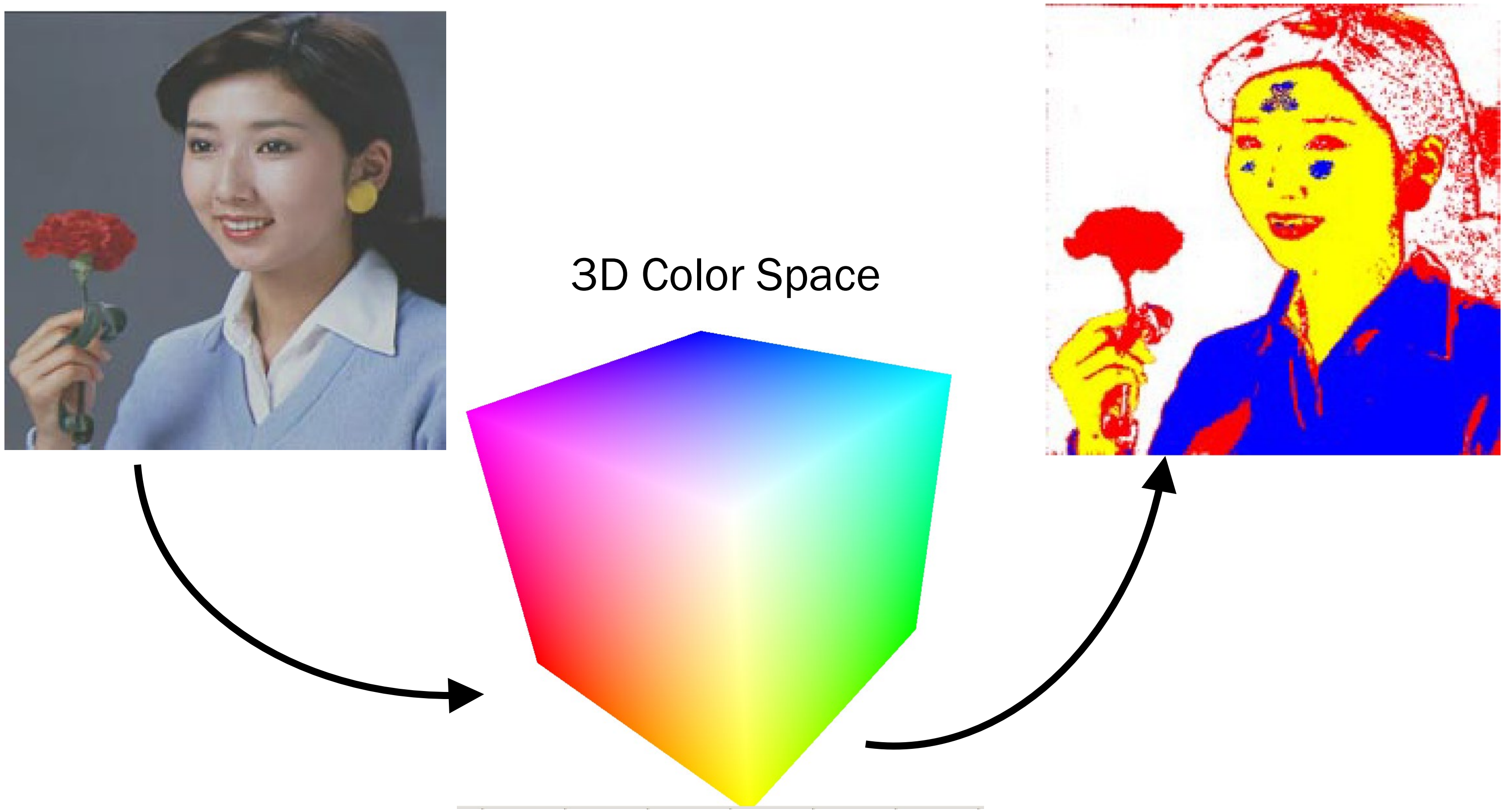
Mid-level vision is the second stage of visual perception that involves inferring the structure of the world from the measurements of low-level vision.

	Low-level vision	Mid-level vision
Purpose	To extract a set of visual primitives at the pixel-level for further visual processing.	To bridge low-level vision and high-level vision with inference of the local structure by involving multiple pixels.
Computational model	Pixel-level linear filtering (convolution)	Region-level statistical Inference (clustering and grouping)
High-level knowledge involved	It is a pre-conscious process.	It is a process that requires some intelligence.
Bottom-up and Top-down flow	It is mainly a data-driven bottom-up process.	Both data-driven bottom-up and knowledge-driven top-down are involved.

Overview of Mid-level Vision



Clustering in the Feature Space



Problem Formulation of Unsupervised Clustering

Given a set of un-labelled data samples $\mathbf{D} = \{y_1, y_2, \dots, y_n\}$ in a d-dimensional space, we partition the set D into a few of disjoint subsets

$$\mathbf{D} = \cup_{j=1}^{K} D_j \quad D_i \cap D_j = \varphi, i \neq j$$

So that points in each subset are coherent according to certain criterion denoted by $f(.)$. We denote a partition by

$$\Pi = (D_1 \quad D_2 \quad \dots \quad D_K)$$

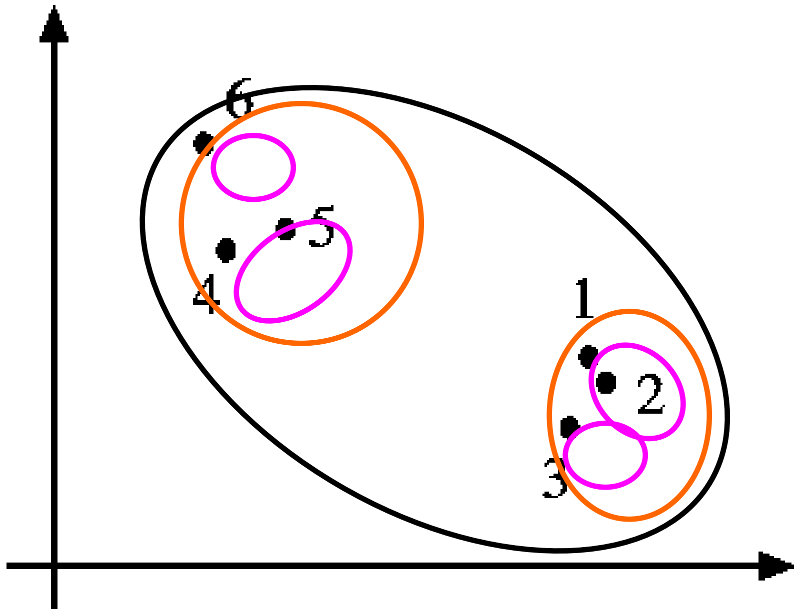
Thus the problem is formulated as

$$\Pi^* = \arg_{\Pi} \max f\,(\Pi).$$

Two Problems to Avoid for Clustering

There are two problems to avoid during clustering:

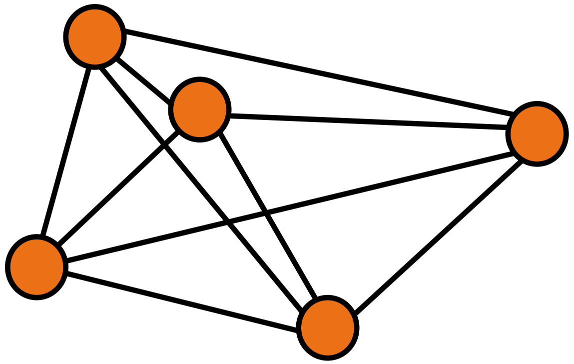
- Under-fitting
- Over-fitting



Two Issues to Consider for Clustering

There are two major issues in thinking of clustering:

- What is a good distance metric?
  - Between two data samples
  - Between a cluster and a sample
  - Between two clusters
- How many clusters are there?



There are two fashions of clustering

- Agglomerative: From many to a few
- Divisive: From one to a few

K-means Clustering

Goal: Given the number of classes *k*, we want to optimize an objective function.

Objective:

- We want to segment data points in the feature space, then **y** represent the feature vector, and **c** is the center of a cluster.
- We assume that elements are close to center of their cluster, yielding the objective function (intra-class divergence)

$$\Phi(\text{clusters,data}) = \sum_{i \in \text{clusters}} \left\{ \sum_{j \in i^{th}\ \text{cluster}} \left| \mathbf{y}_j - \mathbf{c}_i \right|^2 \right\}$$

Two activities:

- Assume the cluster centers are known, allocate each data point to the closest cluster center.
- Assume the allocation is known, compute a new set of cluster centers (means).

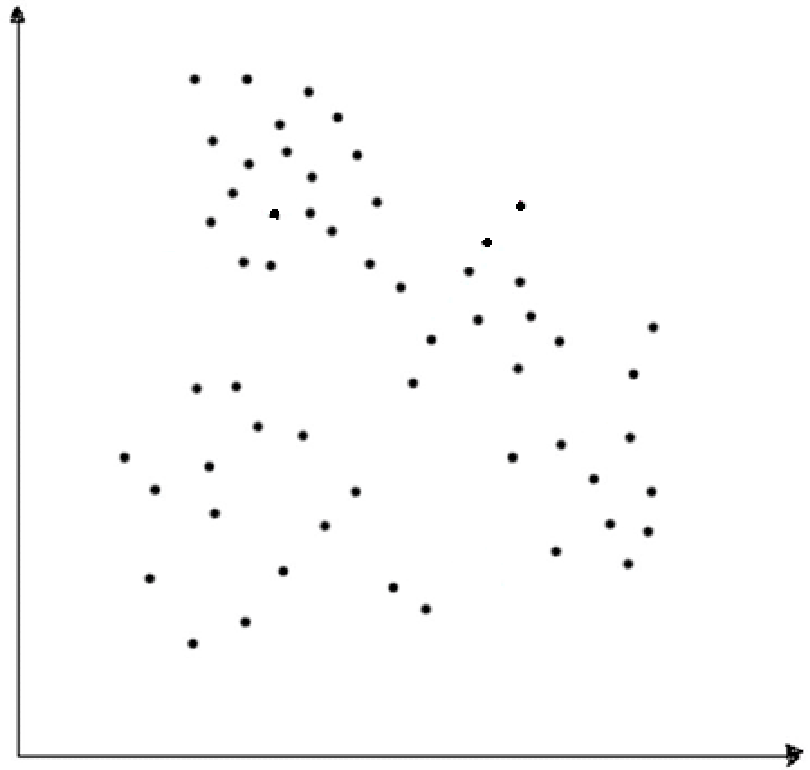
K-means Algorithm

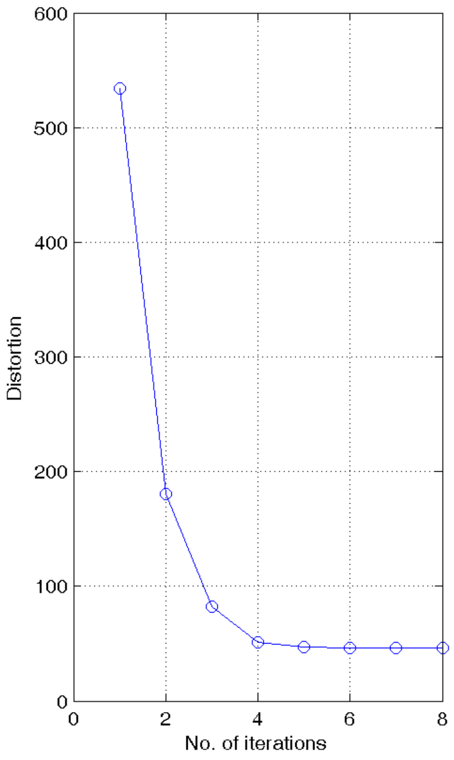
Form K-means clusters from a set of n-dimensional vectors.

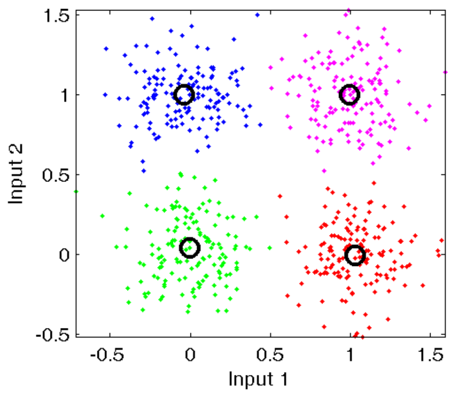
- Set *Nc* =1 (iteration number).
- Choose randomly a set of K means, $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_k\}$.
- For each vector y, compute $D(\mathbf{y}, \mathbf{c}_k)$ for each $k = 1,2, \ldots, K$, and assign y to the cluster with the nearest distance.
- Increment *Nc* by 1 and update the means based on new class labels to get a new set of centers $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_k\}$
- Repeat until no change to $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_k\}$ or the objective function has insignificant change, or *Nc=Nmax*

This process eventually converges to a local minimum of the objective function.

K-means Clustering Examples







K-means Clustering for Color Quantization

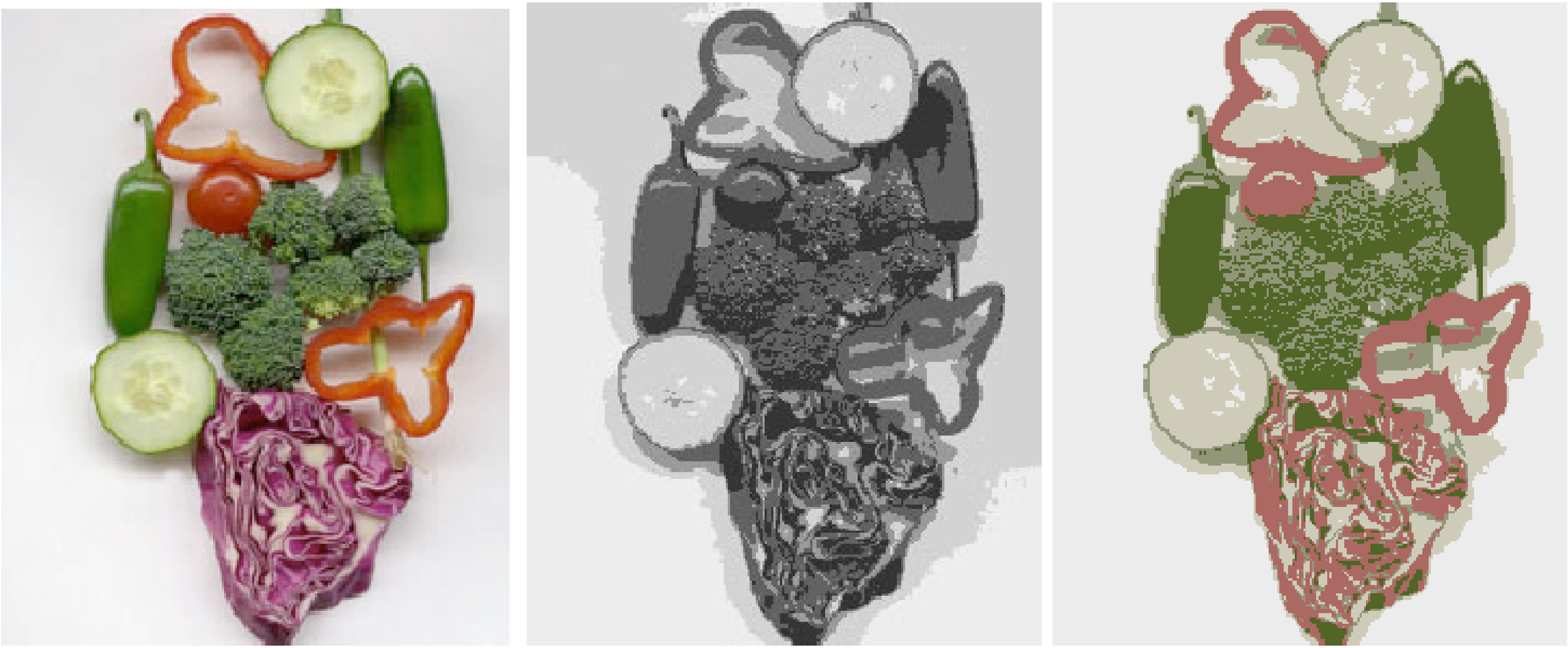


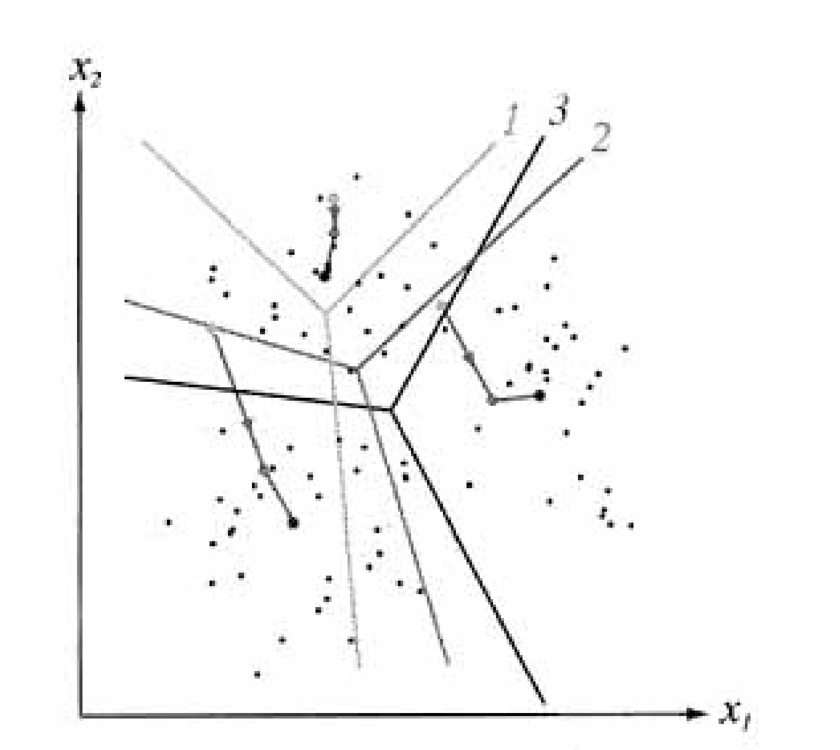
Figure 16.13. On the left, an image of mixed vegetables, which is segmented using k-means to produce the images at center and on the right. We have replaced each pixel with the mean value of its cluster; the result is somewhat like an adaptive requantization, as one would expect. In the center, a segmentation obtained using only the intensity information. At the right, a segmentation obtained using colour information. Each segmentation assumes five clusters.

Why K-means Converge?

Whenever an assignment is changed, the sum squared distances of data-points from their assigned cluster centers is reduced.

Whenever a cluster center is moved the sum squared distances of the data-points from their currently assigned cluster centers is reduced.

If the assignments do not change in the assignment step, we have converged.



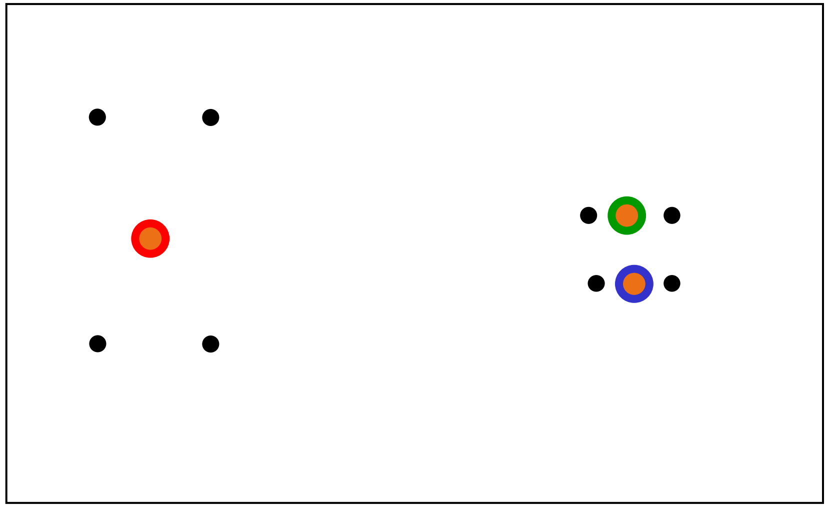
Why K-means can be stuck at a local minima?

There is nothing to prevent k-means getting stuck at local minima.

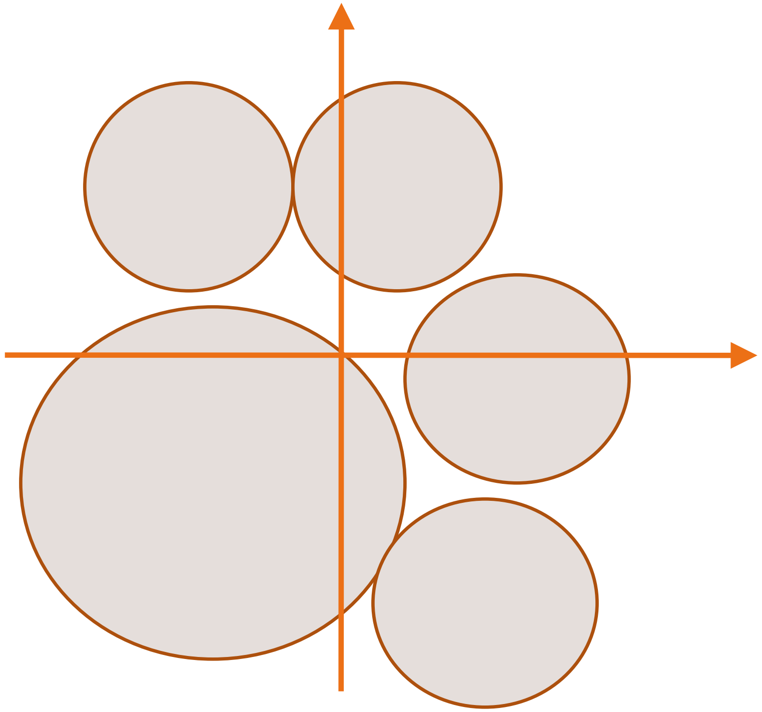
We could try many random starting points

We could try non-local split-and-merge moves: simultaneously merge two nearby clusters and split a big cluster into two.

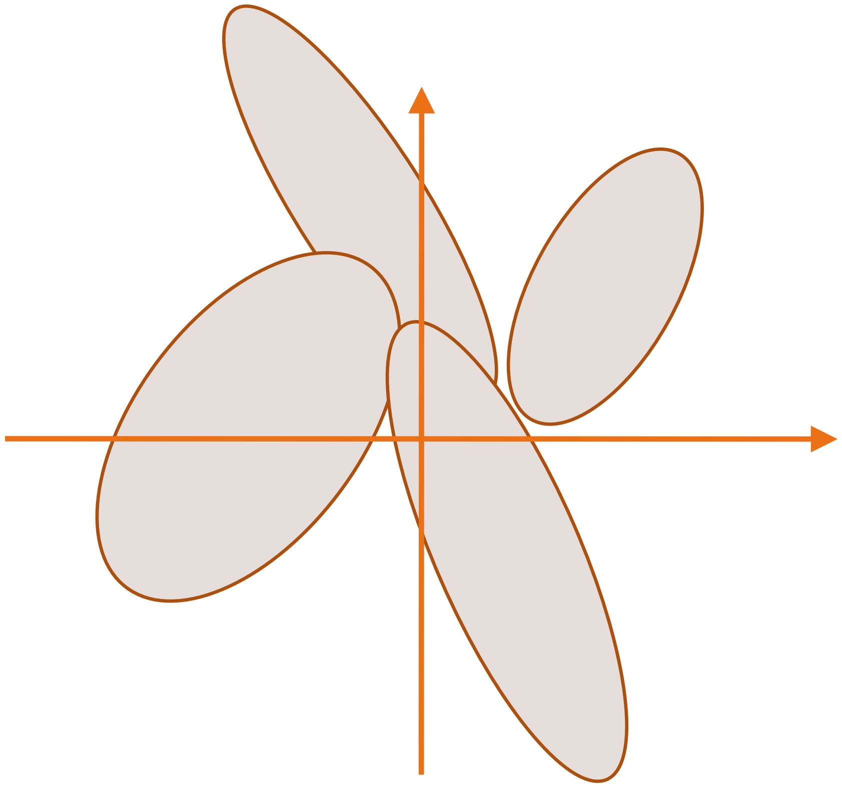
A bad local optimum



Underlying Assumption of K-means



All feature distributions are isotropic and equally probable



K-mean does not work well for the case of non-isotropic feature distributions