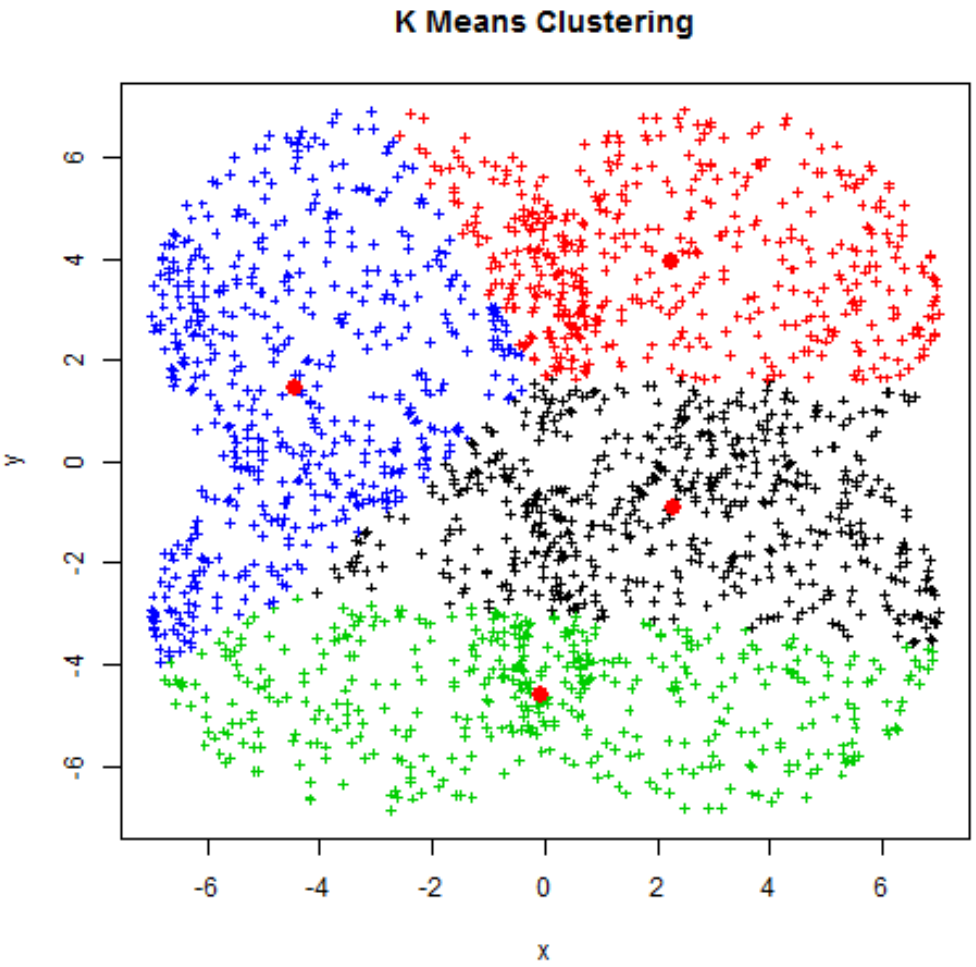
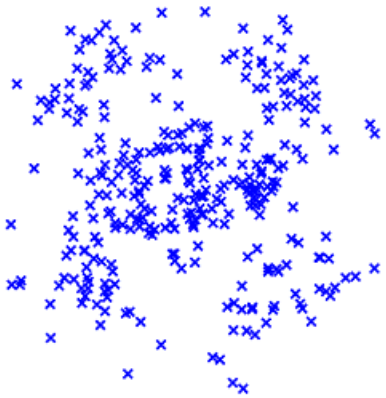
K-means Clustering Algorithm



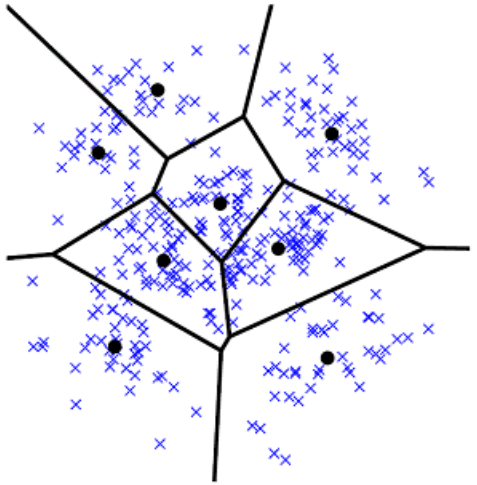
출처: http://rossfarrelly.blogspot.com/2012/12/k-means-clustering.html

1. 군집화(Clustering) 개요

- 아래의 좌측 그림처럼 분산된 데이터 집합은 수많은 특징 벡터들로 이루어져 있음
- 이들 데이터 집합을 그룹들 혹은 클러스터들로 나누어 각 클러스터의 중심의 대표 벡터로 할당하려고 함
- 이러한 대표 벡터들의 개수 K는 미리 정해져 있어야 한다. 즉, K는 결정적이지 않음



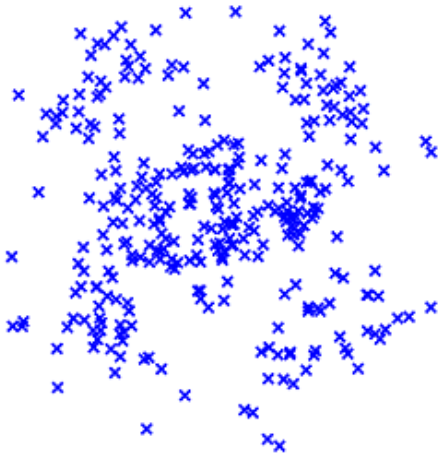
(a)



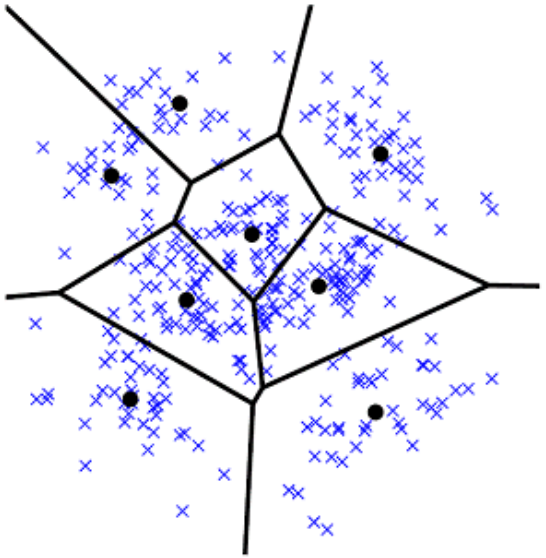
(b)

1. 군집화 개요

- 아래 우측의 그림은 2차원 유클리디안 공간상에서 K=8 의 영역으로 공간을 분할한 그림
- 이 중심 벡터들의 위치는 검은 점으로 보여주고 있는데, 클러스터에 할당되는 특징 벡터들은 '클러스터링' 되었다고 말함.



(a)



(b)

2. 군집화의 세가지 단계

1) 표본 간의 유사(또는 차이)정도를 측정 방법 정의
2) 군집화를 위한 결정 함수 정의
3) 결정 함수를 최대화(또는 최소화) 시키는 알고리즘 정의

- 특징 벡터 집합 x가 새로운 벡터 집합 y로 군집화되는 과정은 특징 벡터들 간에 정의된 거리 척도(distance measure) 또는 측량(metric)과 깊은 관련이 있다.
- **유클리디안 자승 거리** $d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_2 = \sqrt{(\mathbf{x}-\mathbf{y})^{\mathrm{T}}(\mathbf{x}-\mathbf{y})}$
- **중심 평균** $c(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} x_n \quad \text{where} \quad \mathbf{x} = \{x_n \mid n = 1,\ldots,N\}$
- **N개의 벡터에서 전체 왜곡** $D = \sum_{n=1}^{N} d(\mathbf{x}_n, \mathbf{y}_{i(n)}) \quad \text{where} \quad i(n) = k, \ if \ \mathbf{x}_n \in X_k$

3. K-means(Hard C-means(HCM)) 알고리즘

- 다음과 같은 평균자승오차(Mean Squared Error) 함수 를 반복 수행으로 최소화시키는 가장 단순한 군집화(clustring) 과정

$$J_{MSE} = \sum_{i=1}^{K} \sum_{x \approx \omega_i} \left| x - \mu_i \right|^2 \qquad \text{where} \quad \mu_i = \frac{1}{N} \sum_{x \approx \omega_i} x$$

① 클러스터의 개수 K 결정

② 임의의 클러스터 중심을 할당하여 클러스터 초기화

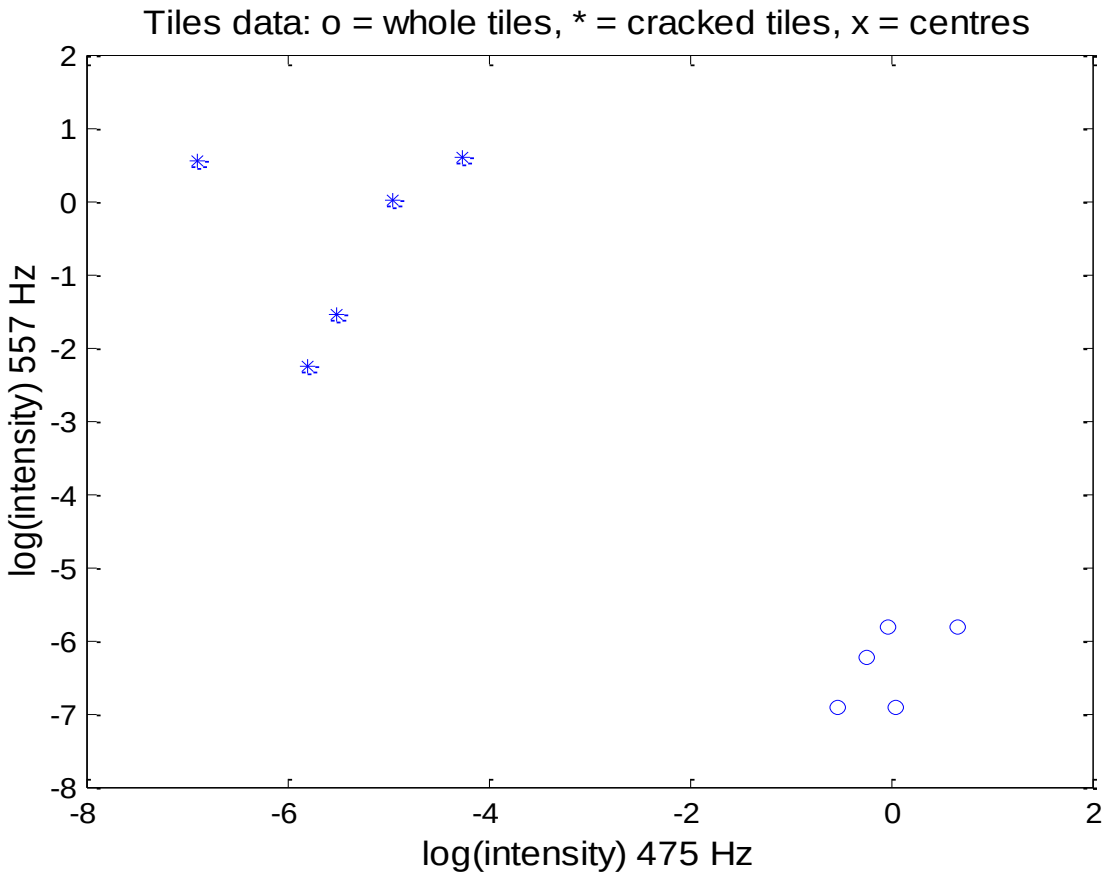
③ 각각의 클러스터에 대하여 표본 평균을 새로 구함

④ 각각의 표본을 가장 근접한 평균을 갖는 클러스터로 다시 할당

⑤ 모든 표본에 변화가 없으면 알고리즘 중지 또는 3번 단계로 이동하여 계속 수행

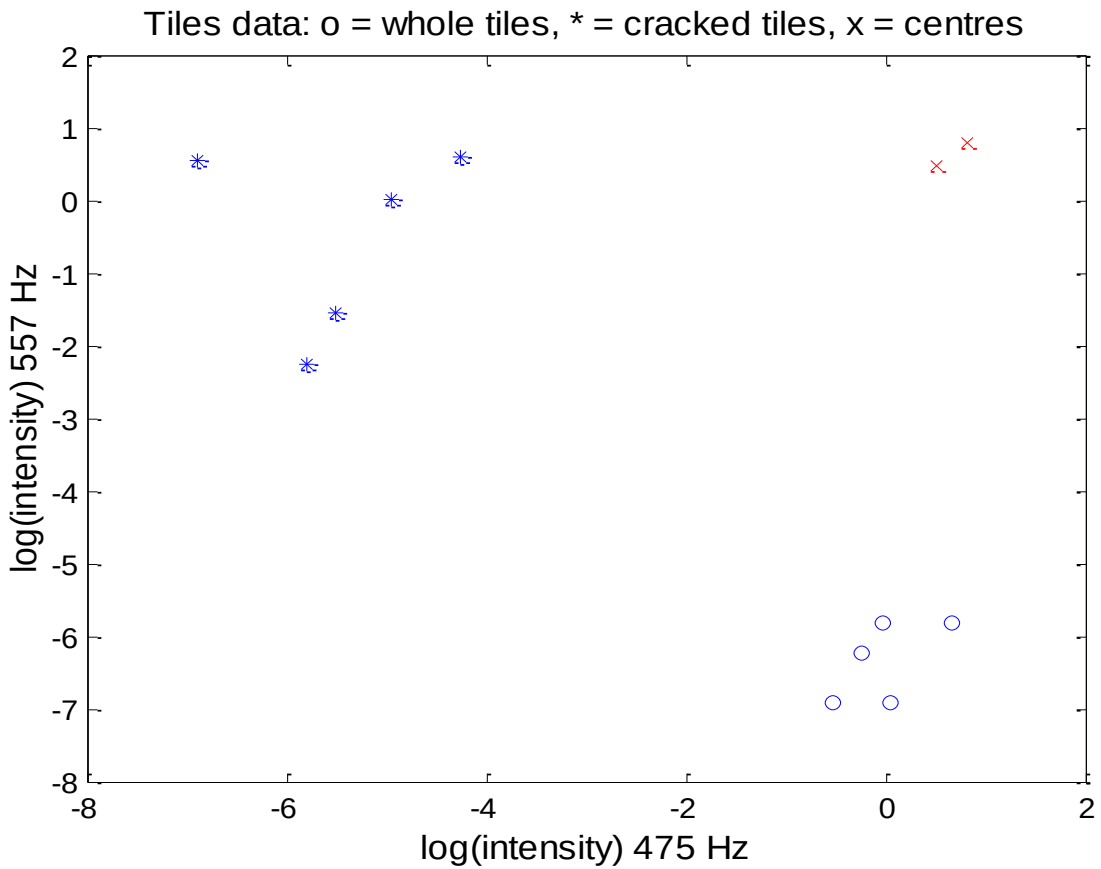
3. K-means 알고리즘

- Plot of tiles by frequencies(logarithms). The whole tiles (o) seem well separated from the cracked tiles (*). The objective is to find the two clusters.



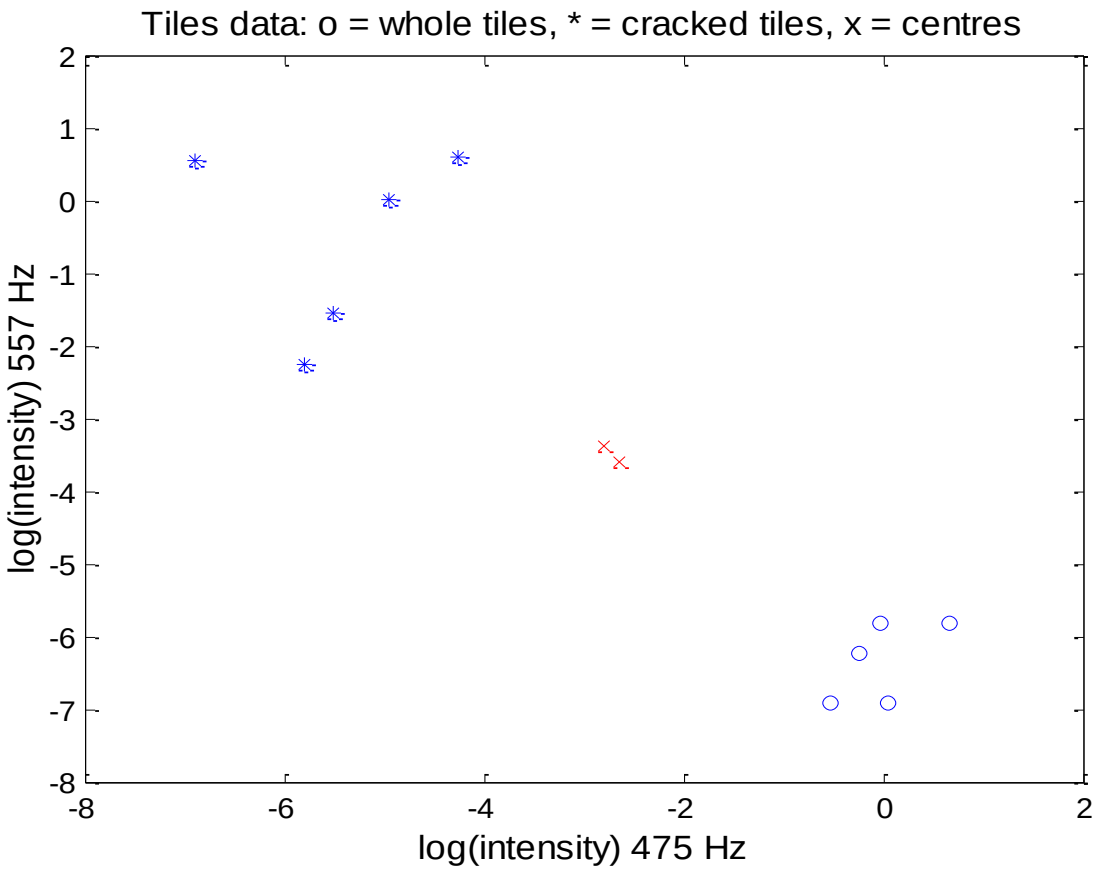
3. K-means 알고리즘

- Place two cluster centers (x) at random.
- Assign each data point (* , o) to the nearest cluster center (x)



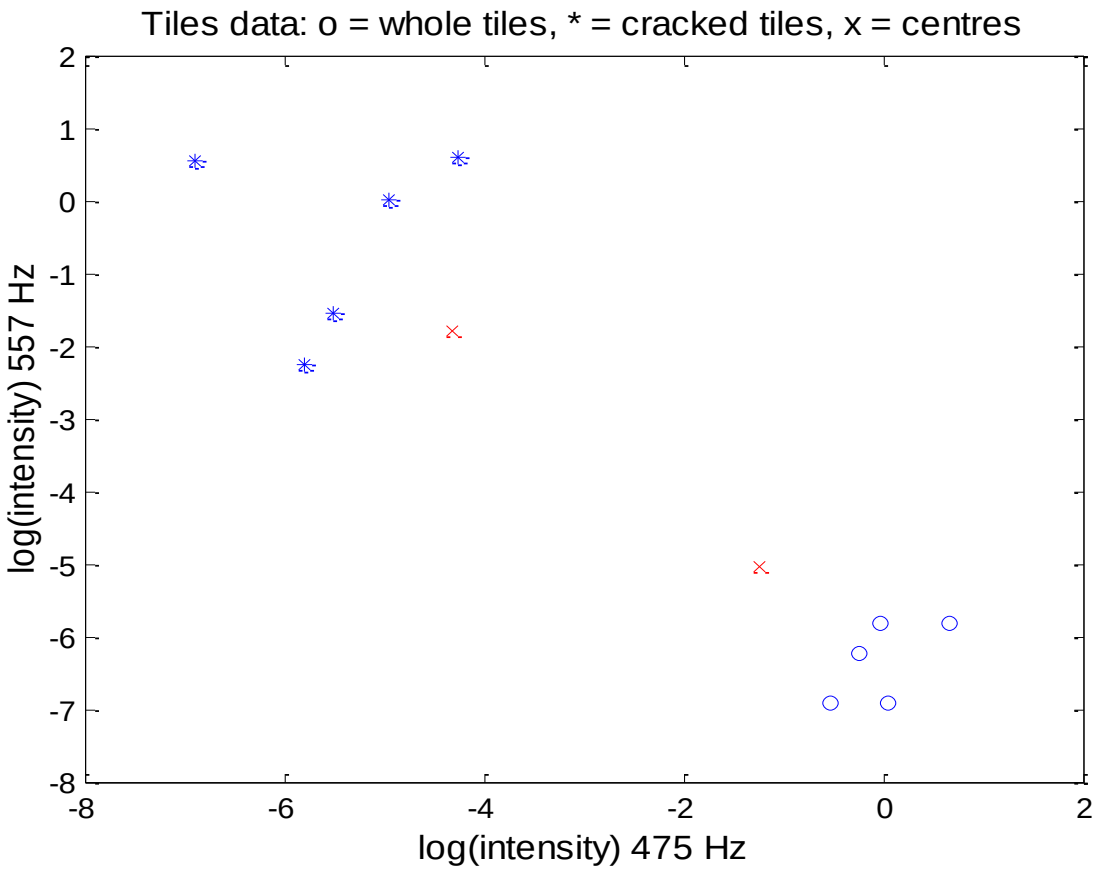
3. K-means 알고리즘

- Compute the new center of each class
- Move the crosses (x)



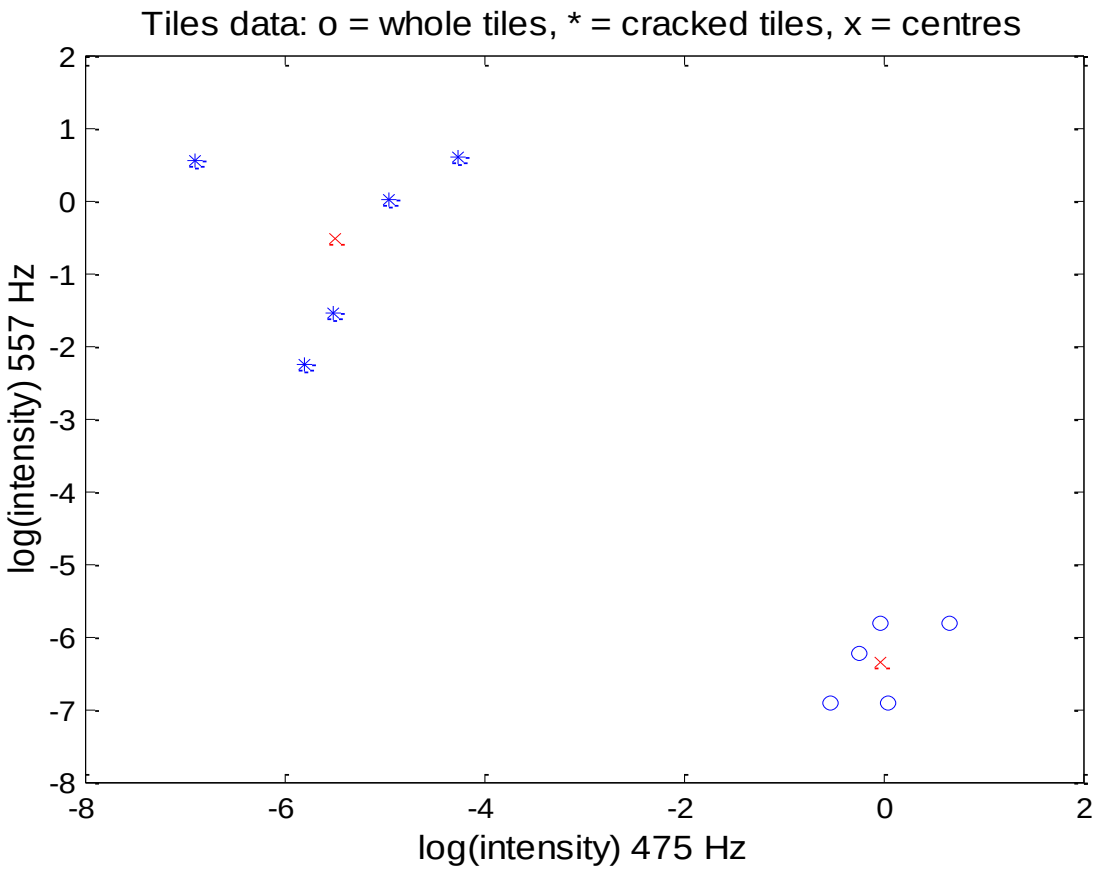
3. K-means 알고리즘

- Iteration 2



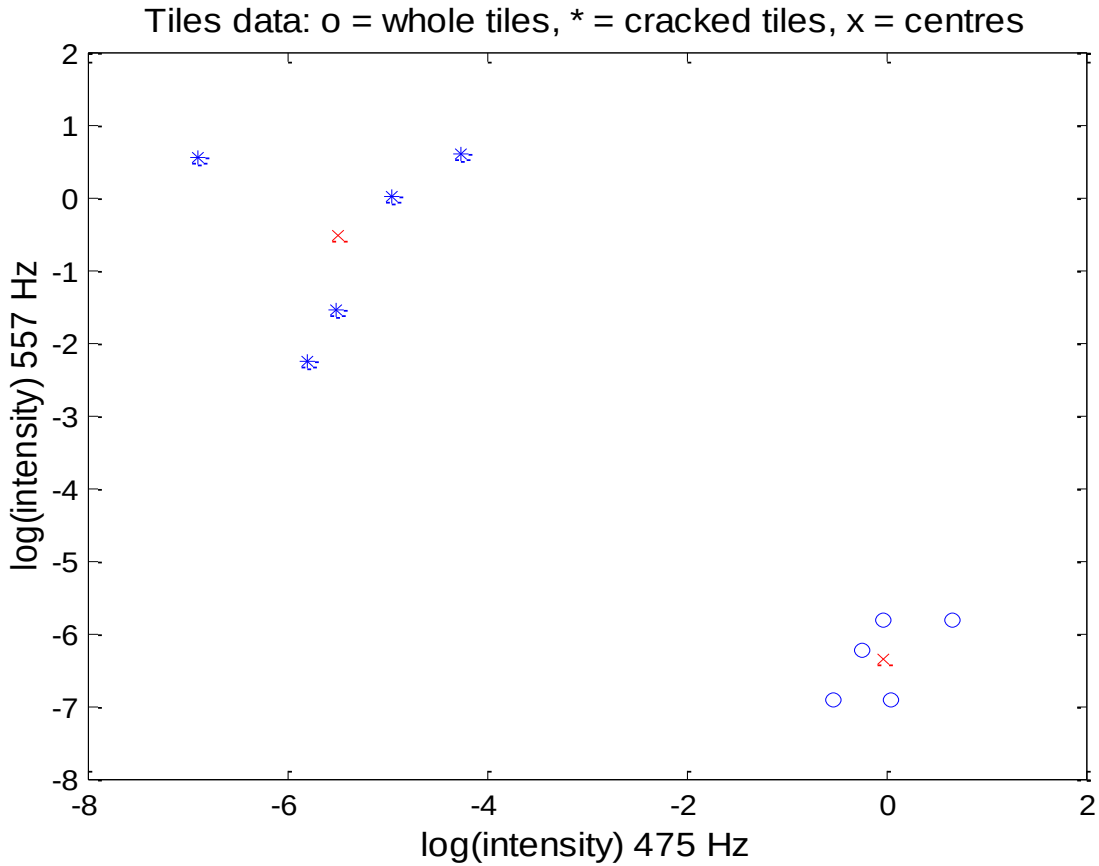
3. K-means 알고리즘

- Iteration 3



3. K-means 알고리즘

- Iteration 4 (then stop, because no visible change)
- Each data point belongs to the cluster defined by the nearest center



4. K-means 알고리즘 계산 절차

① **중심 초기화** : 데이터 집합 {x1,..., xN} 으로부터 임의의 K개의 벡터를 선택하여 K개의 초기 중심 집합 {y1, ..., yK} 을 만든다.

② **클러스터링 단계** : 만약 데이터 xn 이 yi 에 가장 가깝다면 클러스터 Xi 에 속하도록 라벨링 한다. 결국 데이터 집합을 K개의 클러스터들 {X1, ..., XK}로 나누어진다.

$$X_i = \left\{ x_n \,\middle|\, d(\mathbf{x}_n, \mathbf{y}_i) \le d(\mathbf{x}_n, \mathbf{y}_j),\ j = 1, \cdots, K \right\}$$

③ **중심 갱신 단계** : 클러스터링 단계에서 구한 새로운 클러스터들에서 각각의 중심을 갱신한다.

$$y_i = c(X_i) = \frac{1}{M} \sum_{m=1}^{M} \{X_i\},\ \ i = 1, \cdots, K$$

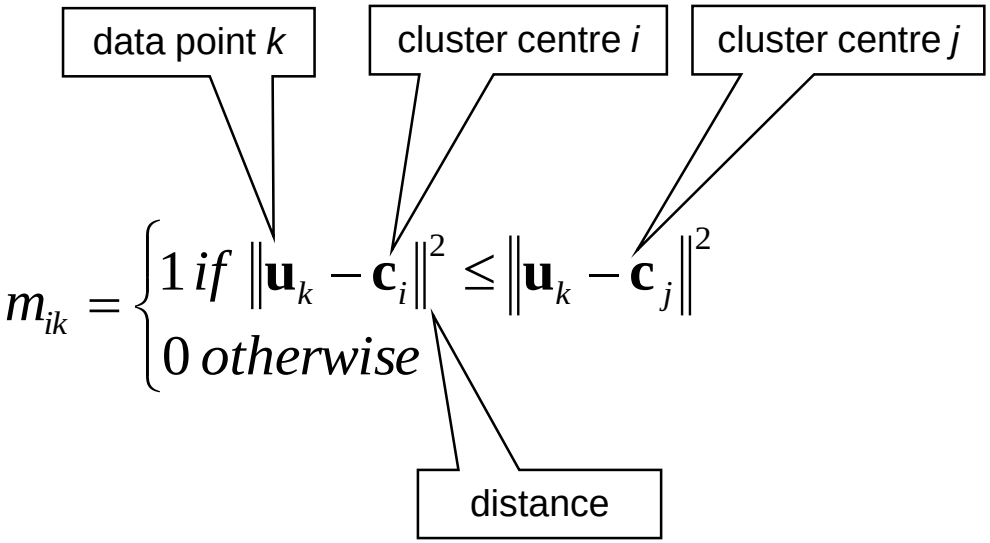
① **왜곡 검증 단계** : 데이터와 가장 가까운 클러스터 중심들과 거리의 합으로 총 왜곡(distortion)을 구한다.

$$D = \sum_{n=1}^{N} d(\mathbf{x}_n, \mathbf{y}_{i(n)}) \quad \text{where} \quad i(n) = k,\ if\ \ \mathbf{x}_n \in X_k$$

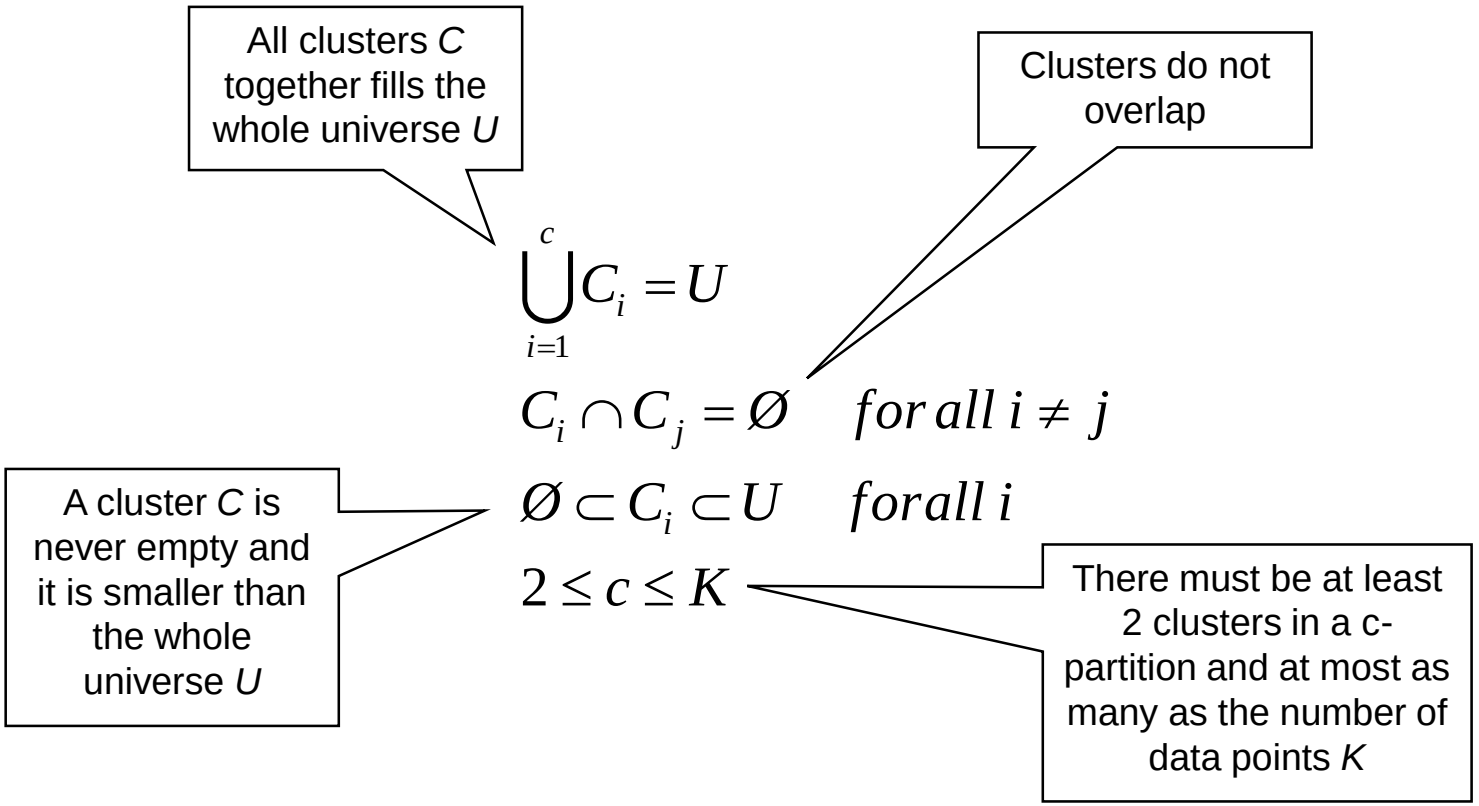
② 총 왜곡이 적절하게 변하지 않거나 설정된 반복 횟수에 도달할 때까지 단계 2~단계 4를 반복한다.

$$\Delta D = \frac{D_{prev} - D_{curr}}{D_{prev}} < 10^{-4}$$

5. Membership matrix **M**



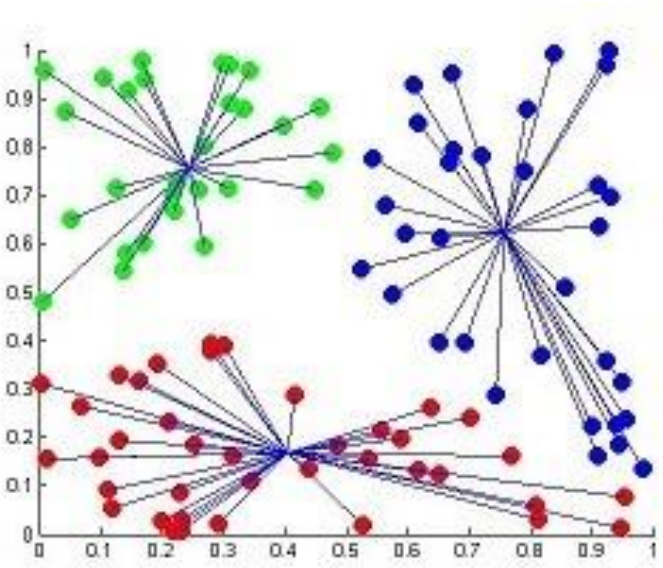
6. c–partition



7. Objective function

Minimize the total sum of all distances

$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \left(\sum_{k, \mathbf{u}_k \in C_i} \left\| \mathbf{u}_k - \mathbf{c}_i \right\|^2 \right)$$



8. K-means 초기화 기법

◆ 무작위 분할 (Random Partition)

- 무작위 분할 알고리즘은 가장 많이 쓰이는 초기화 기법
- 각 데이터들을 임의의 클러스터에 배당한 후, 각 클러스터에 배당된 점들의 평균 값을 초기로 설정
- 무작위 분할 기법의 경우 다른 기법들과는 달리 데이터 순서에 대해 독립적
- 무작위 분할의 경우 초기 클러스터가 각 데이터들에 대해 고르게 분포되기 때문에 각 초기 클러스터의 무게중심들이 데이터 집합의 중심에 가깝게 위치하는 경향을 띔
- 이러한 특성 때문에 **K-조화 평균**이나 **퍼지 K-평균**에서는 무작위 분할이 선호

8. K-means 초기화 기법

◆ Forgy 알고리즘

- 1965년 Forgy에 의해 고안된 알고리즘
- 현재 주로 쓰이는 초기화 기법 중 하나
- 데이터 집합으로부터 임의의 k개의 데이터를 선택하여 각 클러스터의 중심값으로 설정
- 무작위 분할 기법과 마찬가지로 Forgy 알고리즘은 데이터 순서에 대해 독립적
- 초기 클러스터가 임의의 k개의 점들에 의해 설정되기 때문에 각 클러스터의 무게중심이 중심으로부터 퍼져있는 경향을 띔
- 이러한 특성 때문에 **EM 알고리즘**이나 **표준 K-평균 알고리즘**에서는 Forgy 알고리즘이 선호

8. K-means 초기화 기법

MacQueen 알고리즘

- 1967년 MacQueen에 의해 고안된 알고리즘
- Forgy 알고리즘과 마찬가지로 데이터 집합으로 부터 임의의 k개의 데이터를 선택하여 각 클러스터의 중심값으로 설정
- 이후 선택되지 않은 각 데이터들에 대해, 해당 점으로부터 가장 가까운 클러스터를 찾아 데이터를 배당
- 모든 데이터들이 클러스터에 배당되고 나면 각 클러스터의 무게중심을 다시 계산하여 중심값으로 다시 설정
- MacQueen 알고리즘의 경우 최종 수렴에 가까운 클러스터를 찾는 것은 비교적 빠르나, 최종 수렴에 해당하는 클러스터를 찾는 것은 매우 느림

8. K-means 초기화 기법

◆ Kaufman 알고리즘

- 1990년 Kaufman과 Rousseeuw에 의해 고안된 알고리즘
- 전체 데이터 집합 중 가장 중심에 위치한 데이터를 첫번째 중심값으로 설정
- 이후 선택되지 않은 각 데이터들에 대해, 가장 가까운 무게중심 보다 선택되지 않은 데이터 집합에 더 근접하게 위치한 데이터를 또 다른 중심값으로 설정하는 것을 총 k개의 중심값이 설정될 때 까지 반복
- 무작위 분할과 마찬가지로, Kaufman 알고리즘은 초기 클러스터링과 데이터 순서에 대해 비교적 독립적이기 때문에, 해당 요소들에 의존적인 다른 알고리즘들 보다 월등한 성능을 보임

9. K-means 알고리즘 실습

IRIS 데이터세트를 이용한 실습

- https://en.wikipedia.org/wiki/Iris_flower_data_set
- https://rpubs.com/wjholst/322258
- 3개의 클래스로 각 50개씩, 전체 150개 데이터로 이루어짐
- iris2.dat 파일 참조
- 가장 기본적인 K-means 알고리즘을 C++로 구현
- 초기값 설정 알고리즘은 아래의 두 가지 방법 사용
  - Forgy 기법
  - Random Partition 기법