Introduction to Machine Learning
Part 2

Yingyu Liang

yliang@cs.wisc.edu

Computer Sciences Department

University of Wisconsin, Madison

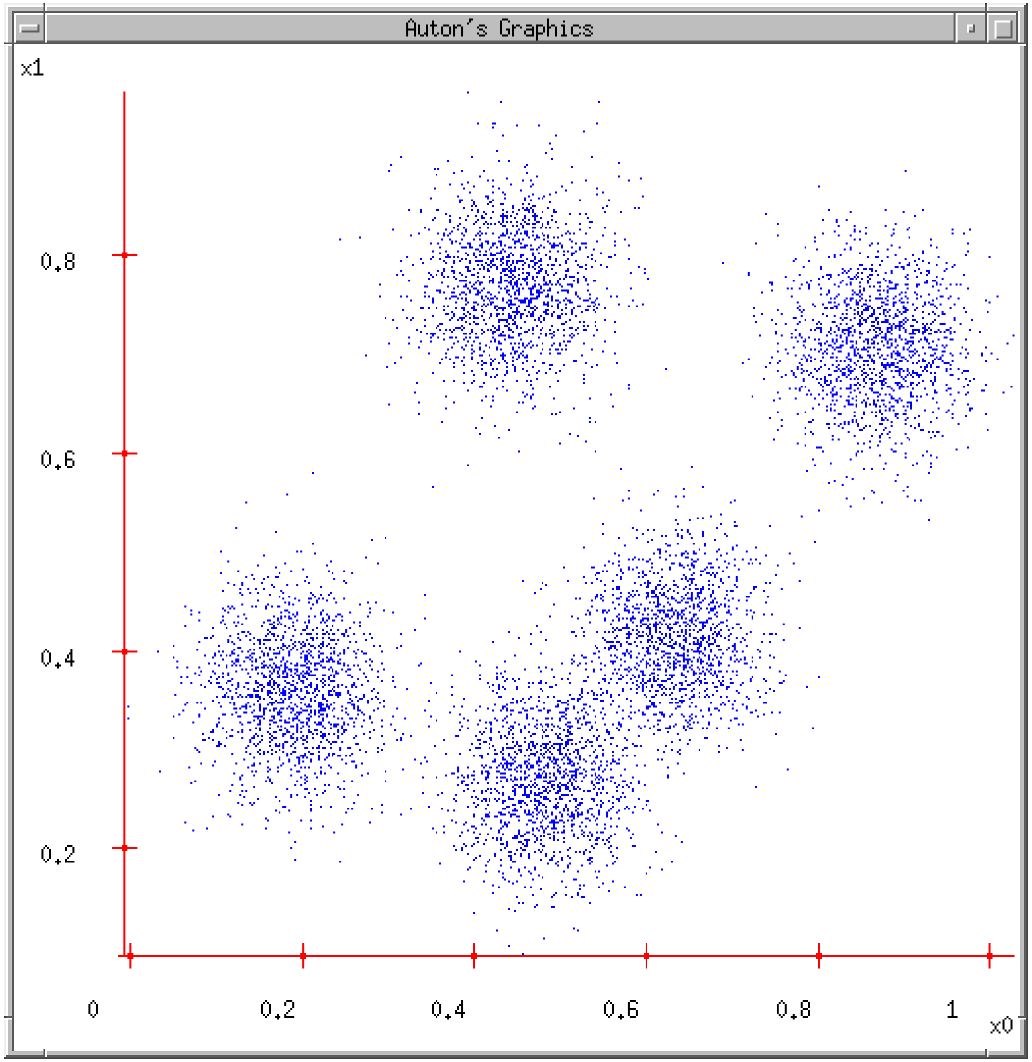
[Based on slides from Jerry Zhu]

K-means clustering

- Very popular clustering method
- Don't confuse it with the k-NN classifier
- Input:
  - A dataset x_1, …, x_n, each point is a numerical feature vector
  - Assume the number of clusters, k, is given

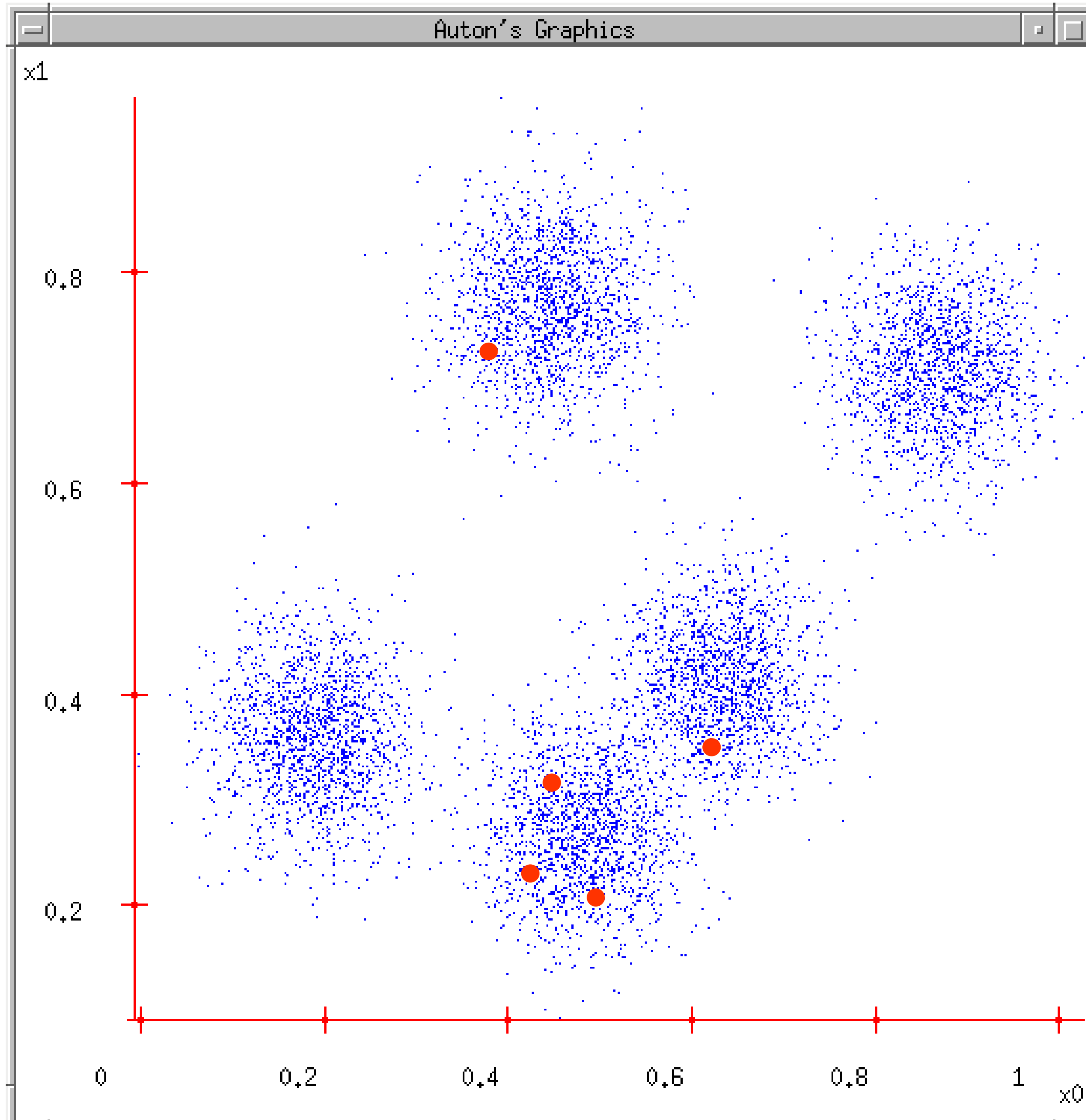
K-means clustering

- The dataset. Input k=5



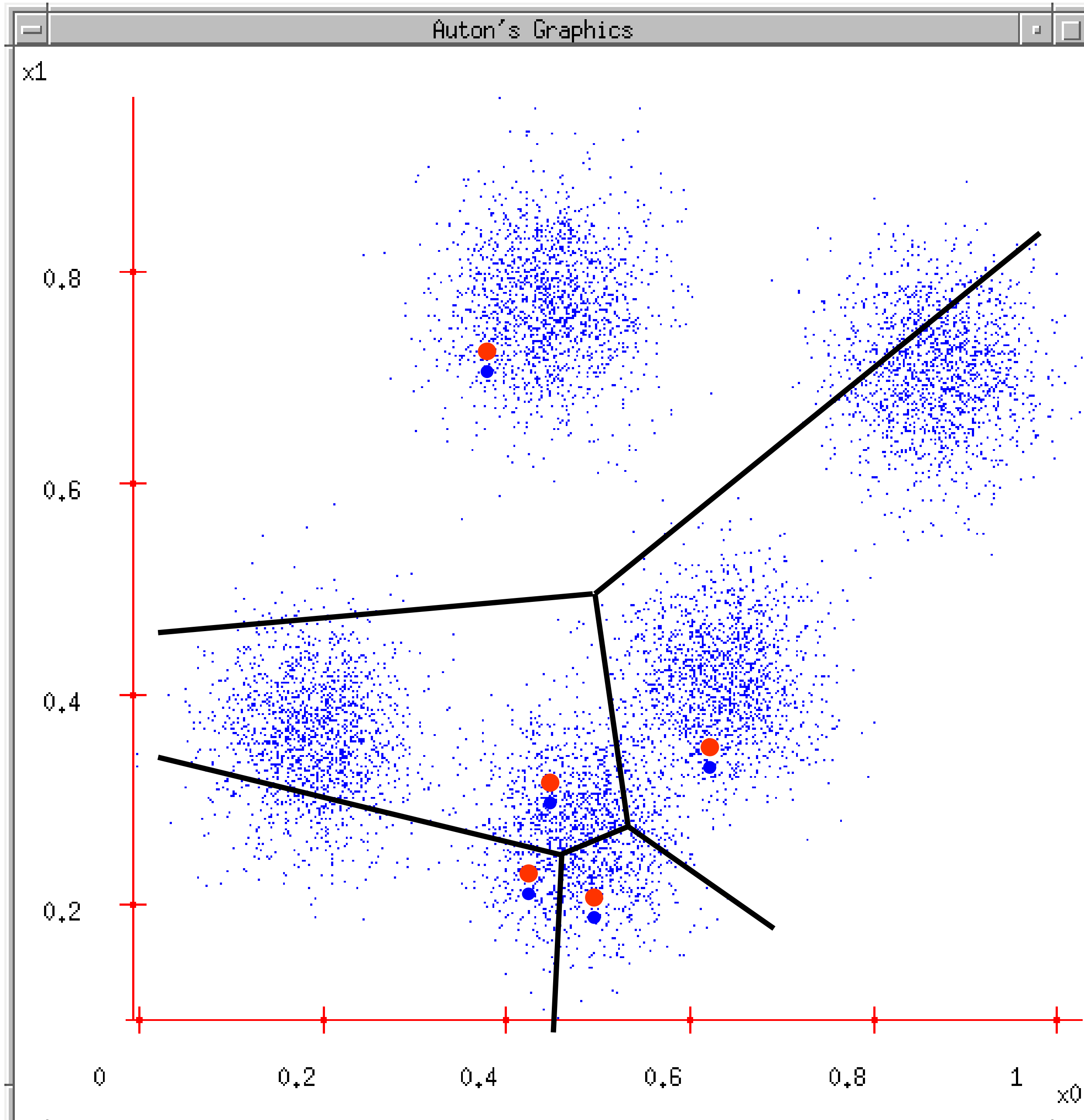
K-means clustering

- Randomly picking 5 positions as initial cluster centers (not necessarily a data point)



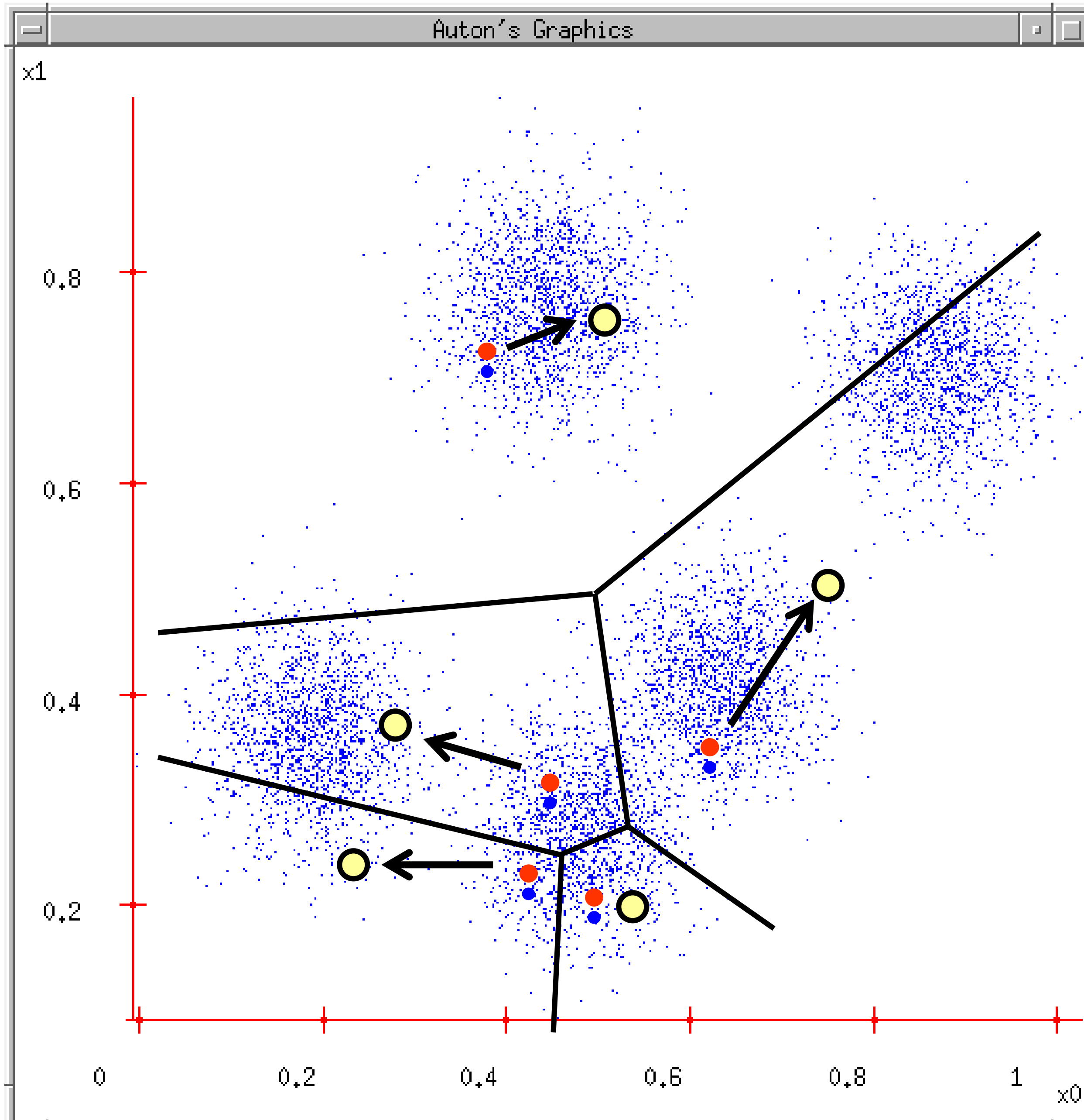
K-means clustering

- Each point finds which cluster center it is closest to (very much like 1NN). The point belongs to that cluster.



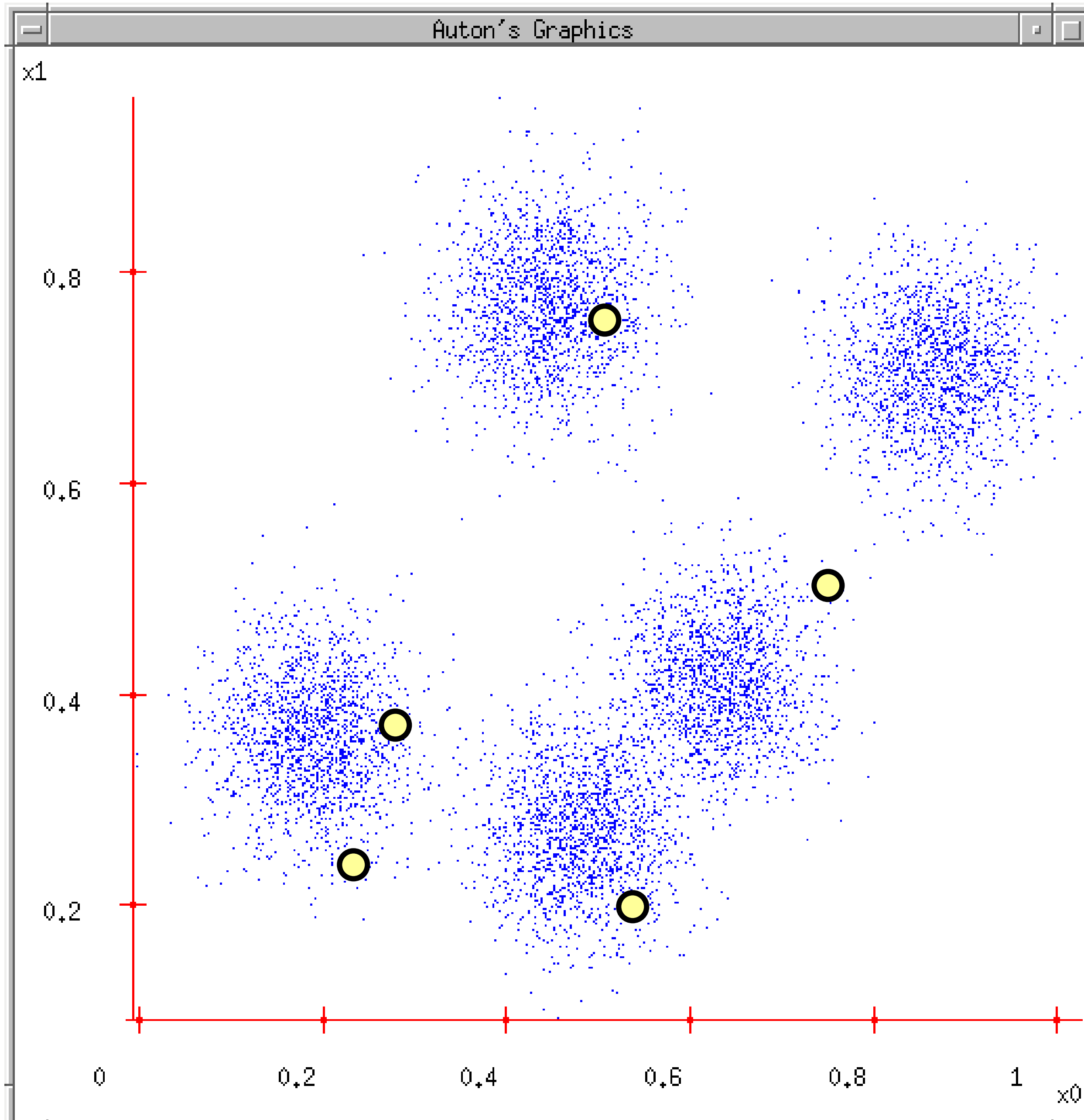
K-means clustering

- Each cluster computes its new centroid, based on which points belong to it

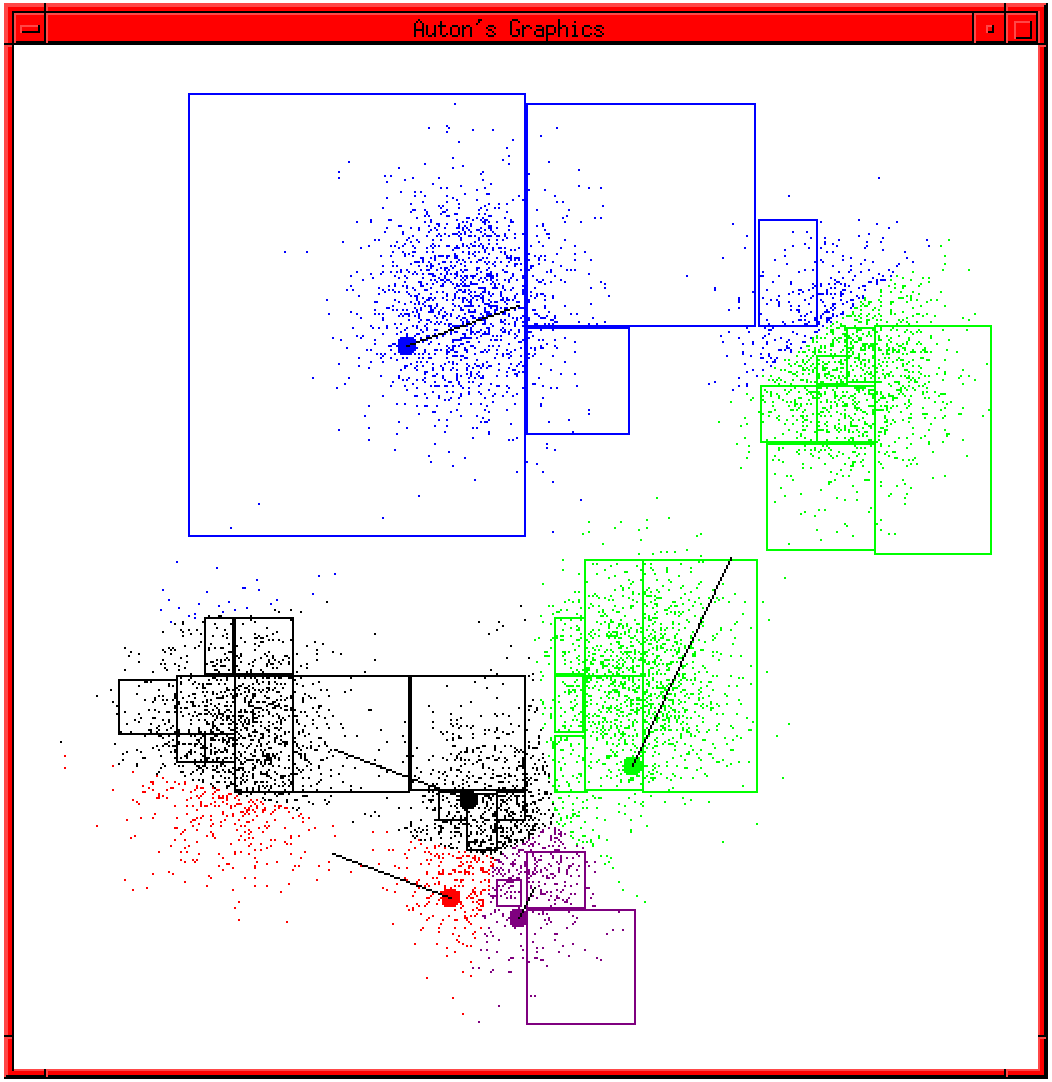


K-means clustering

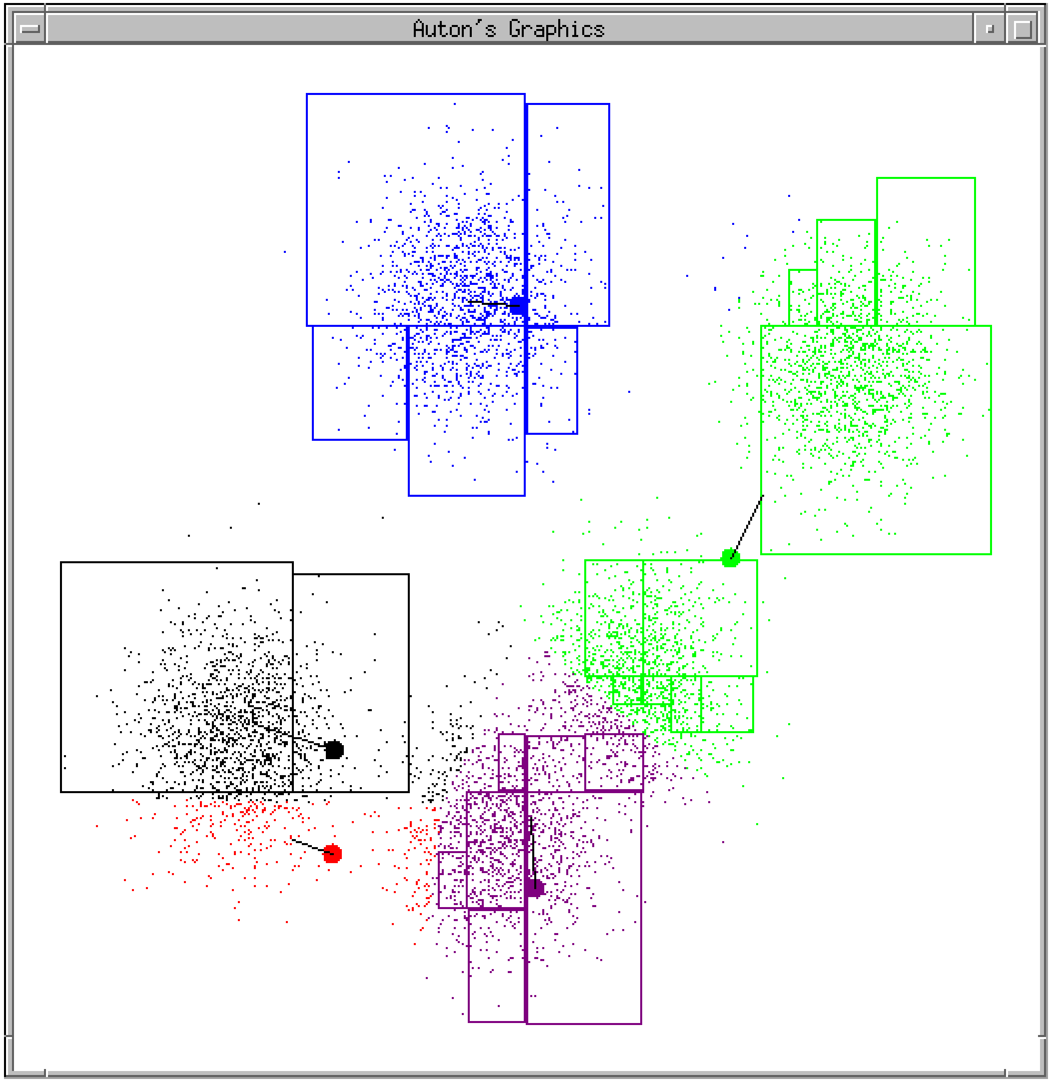
- Each cluster computes its new centroid, based on which points belong to it
- And repeat until convergence (cluster centers no longer move)…



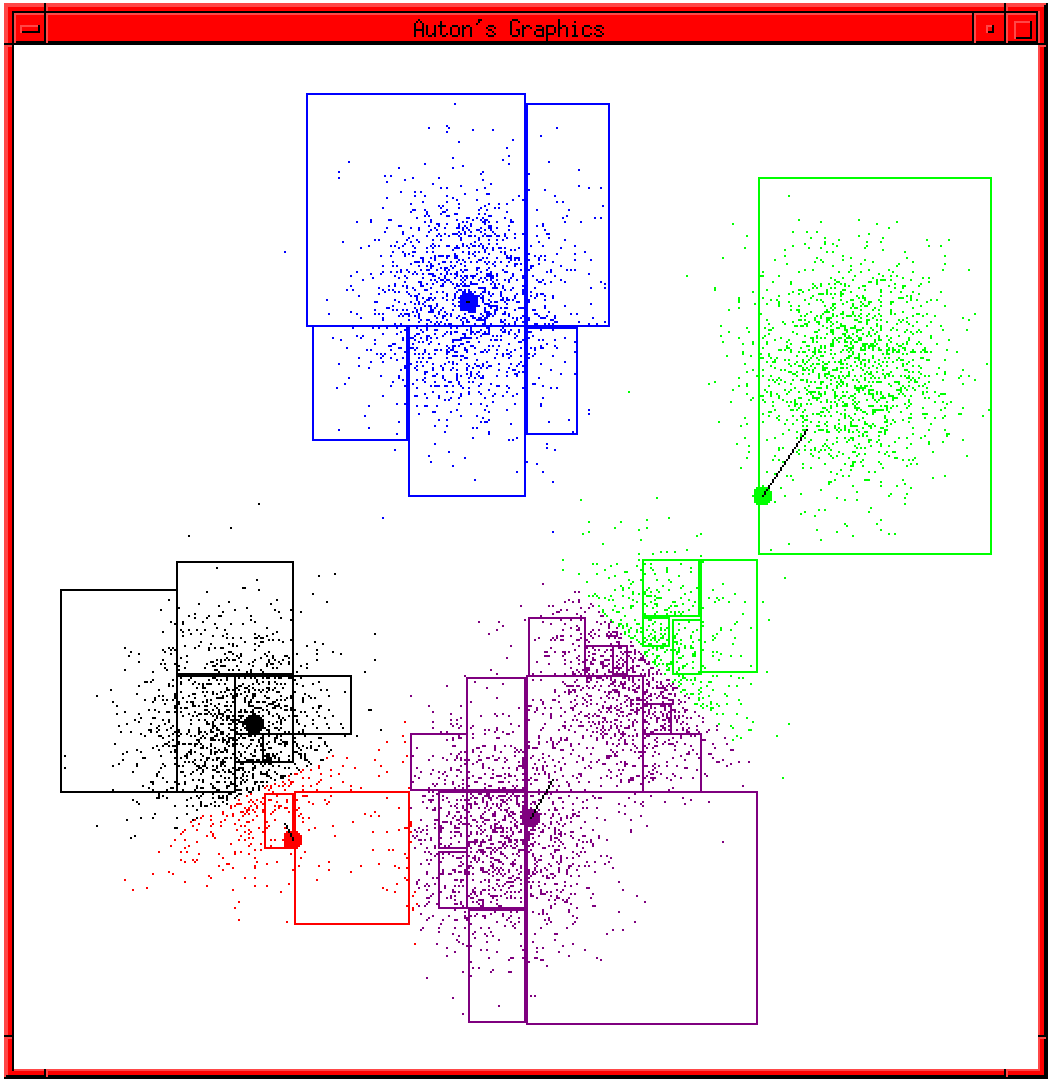
K-means: initial cluster centers



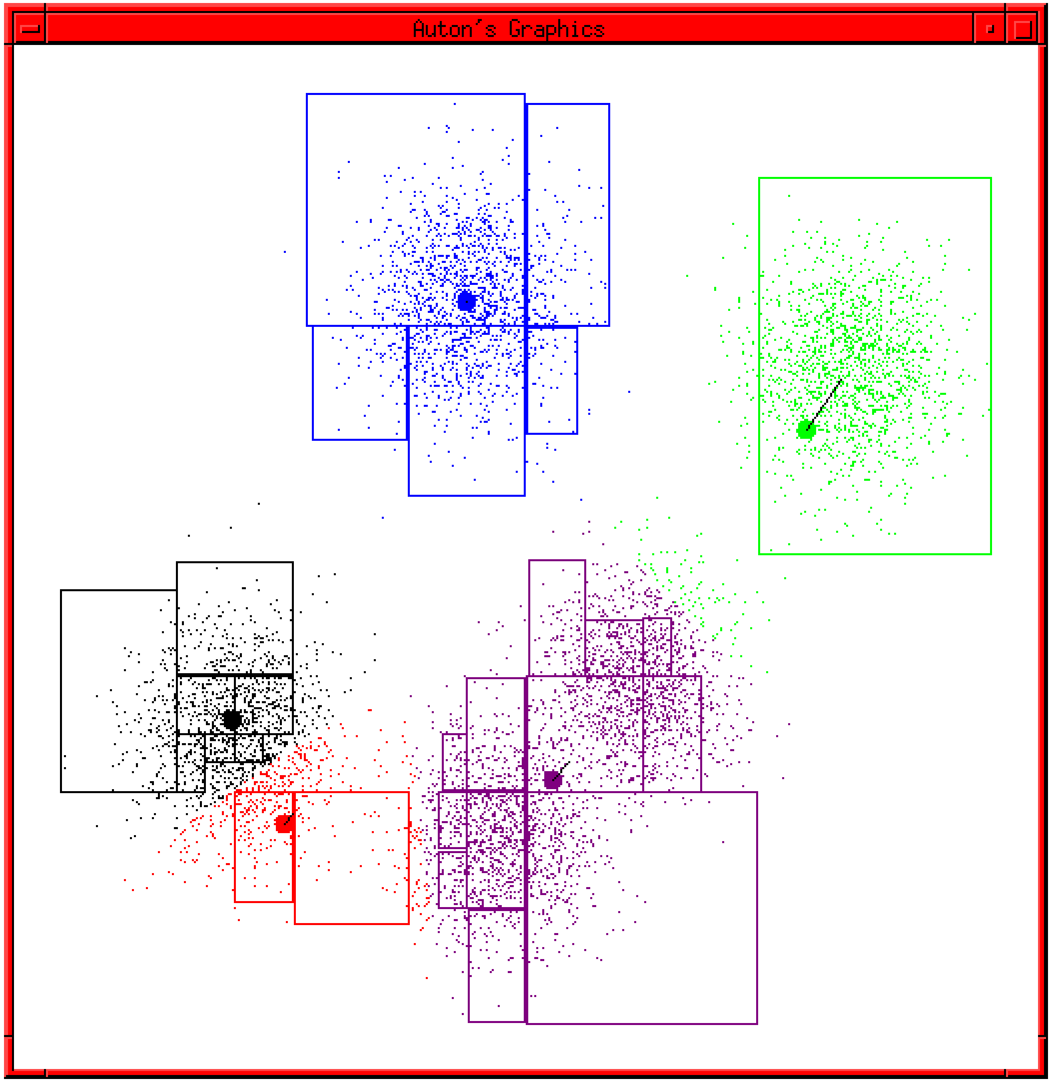
K-means in action



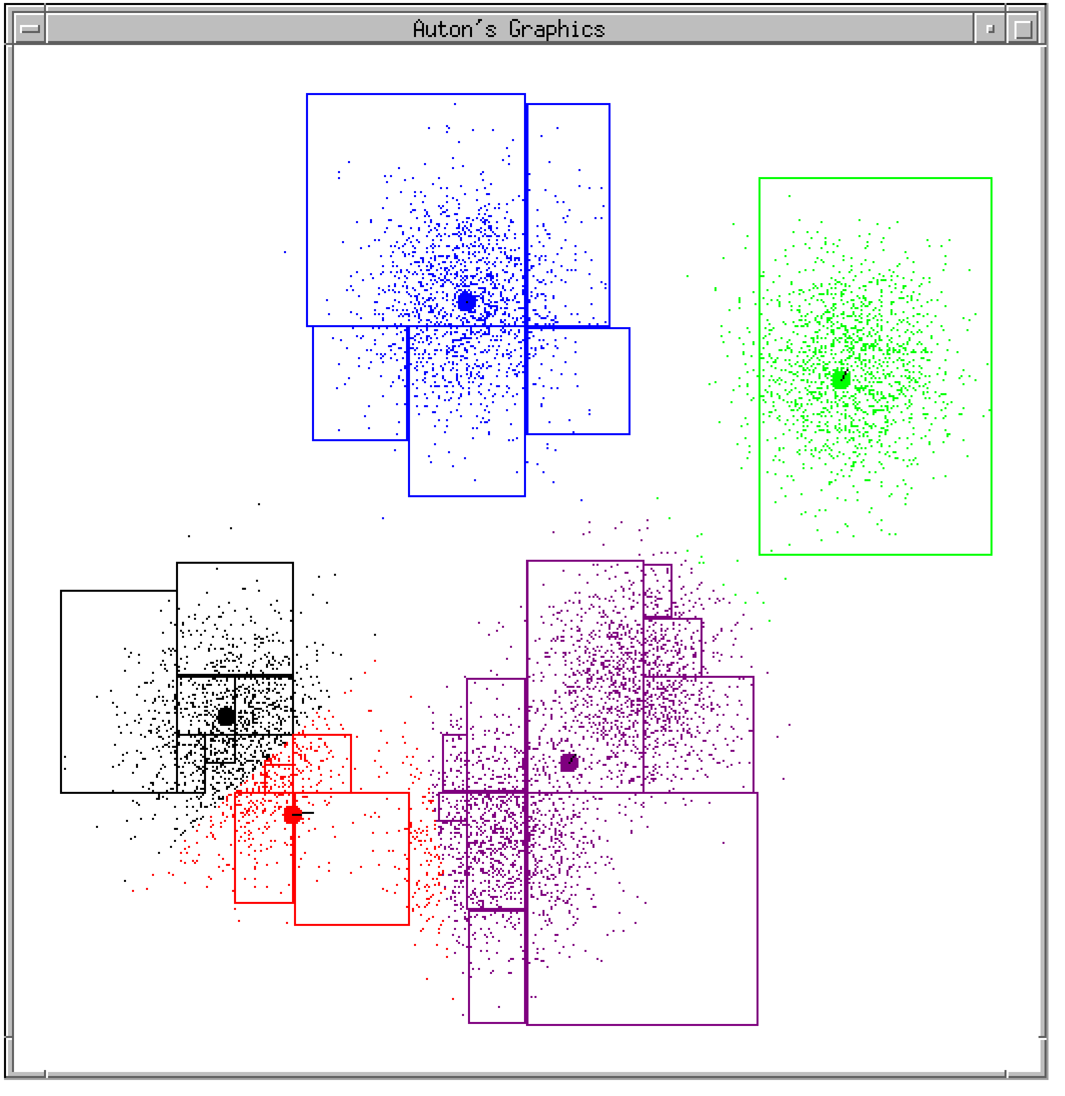
K-means in action



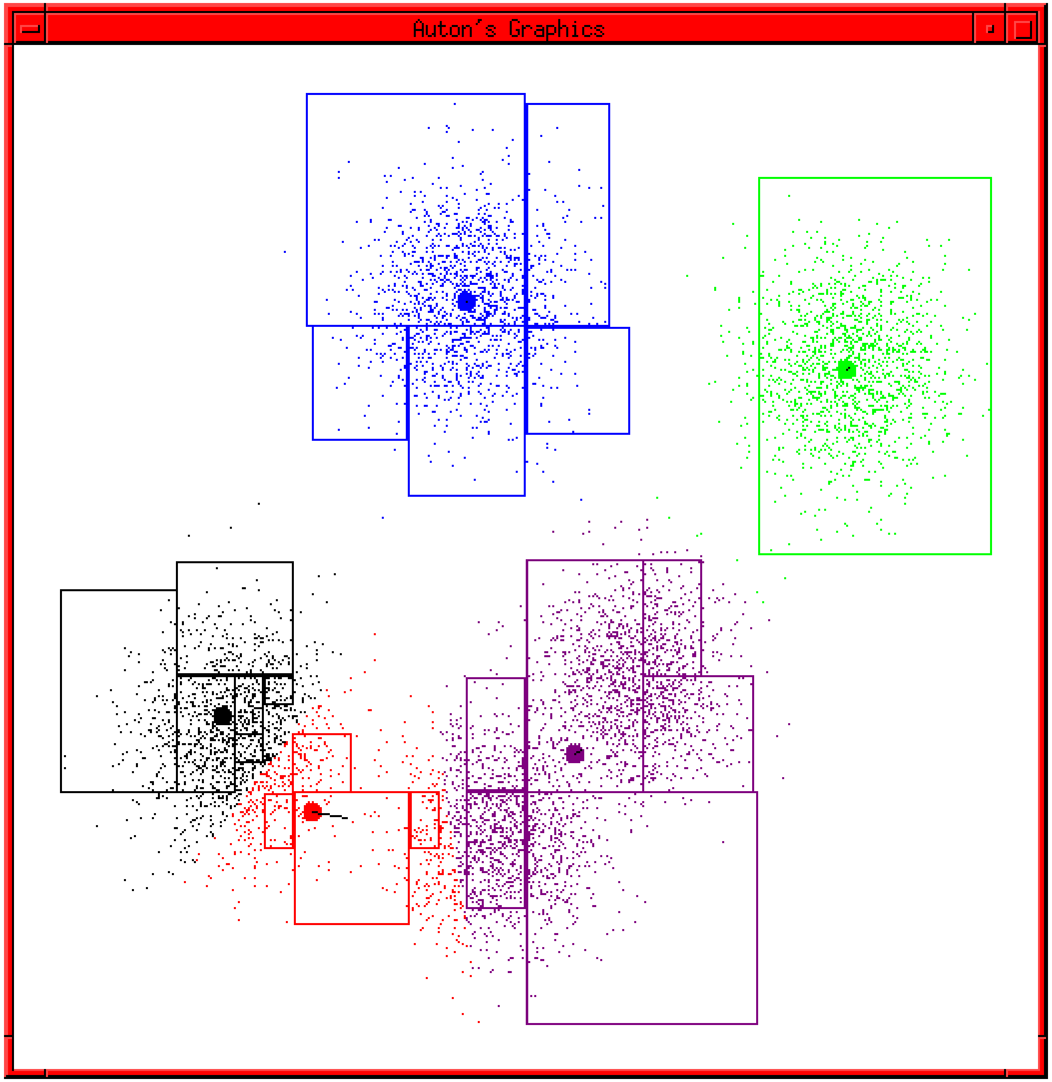
K-means in action



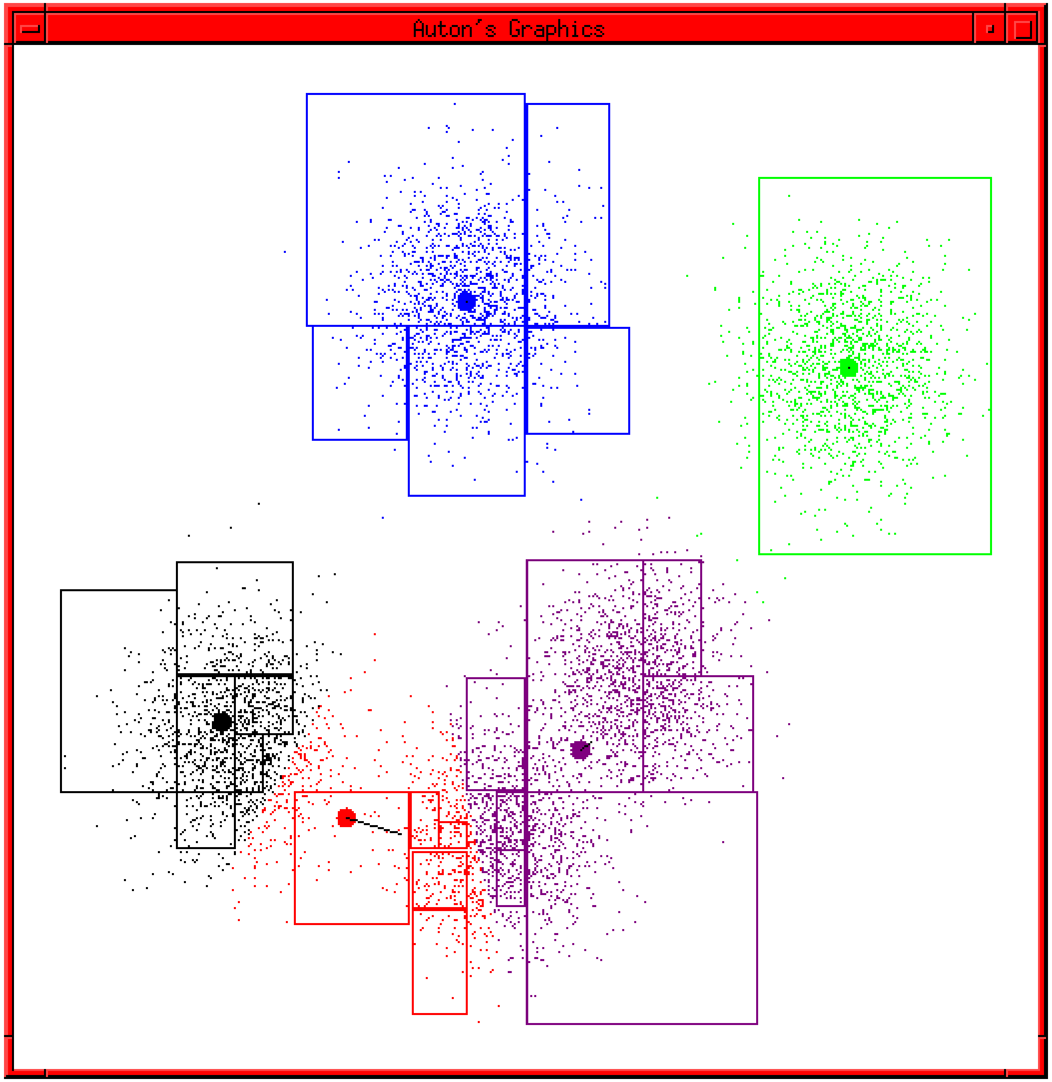
K-means in action



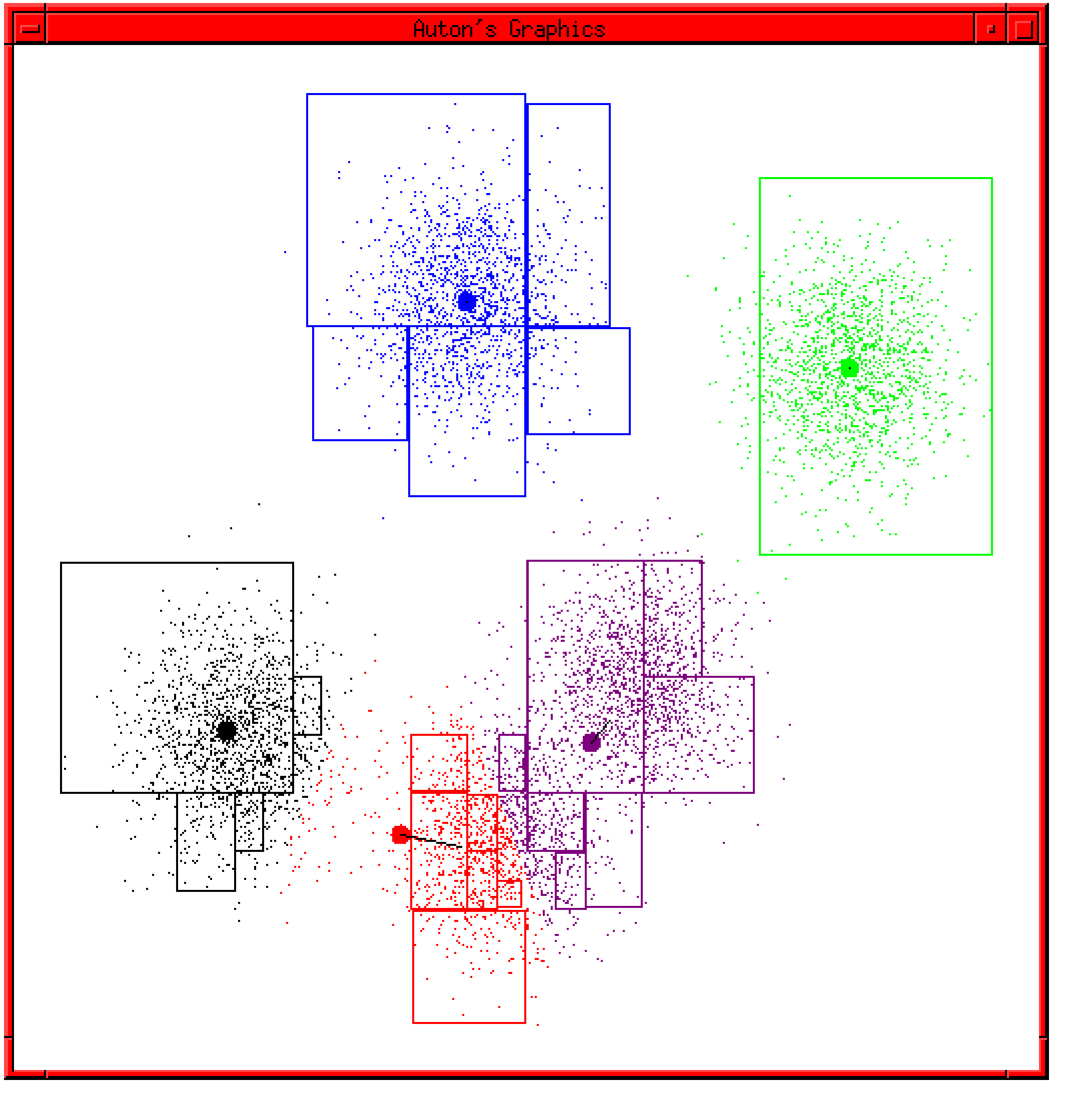
K-means in action



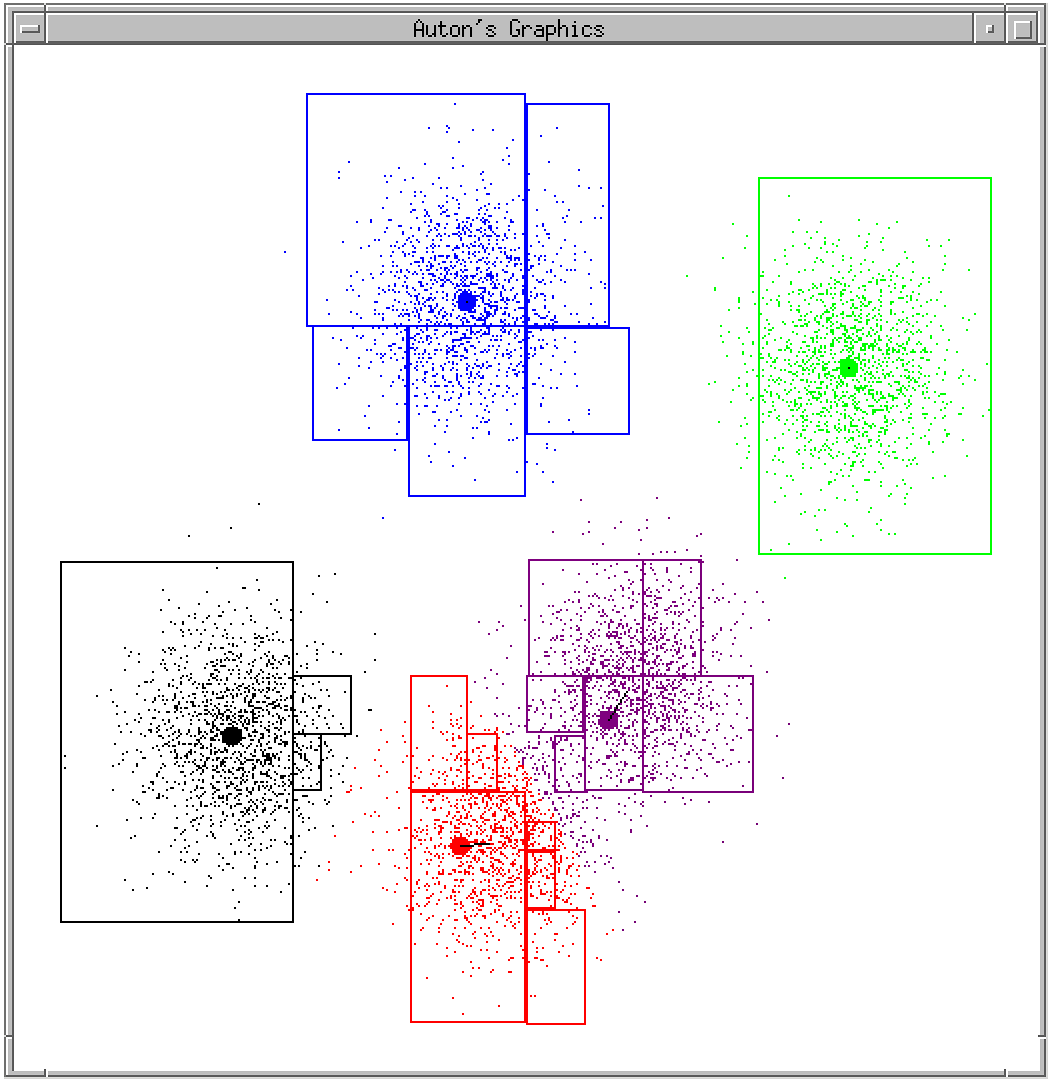
K-means in action



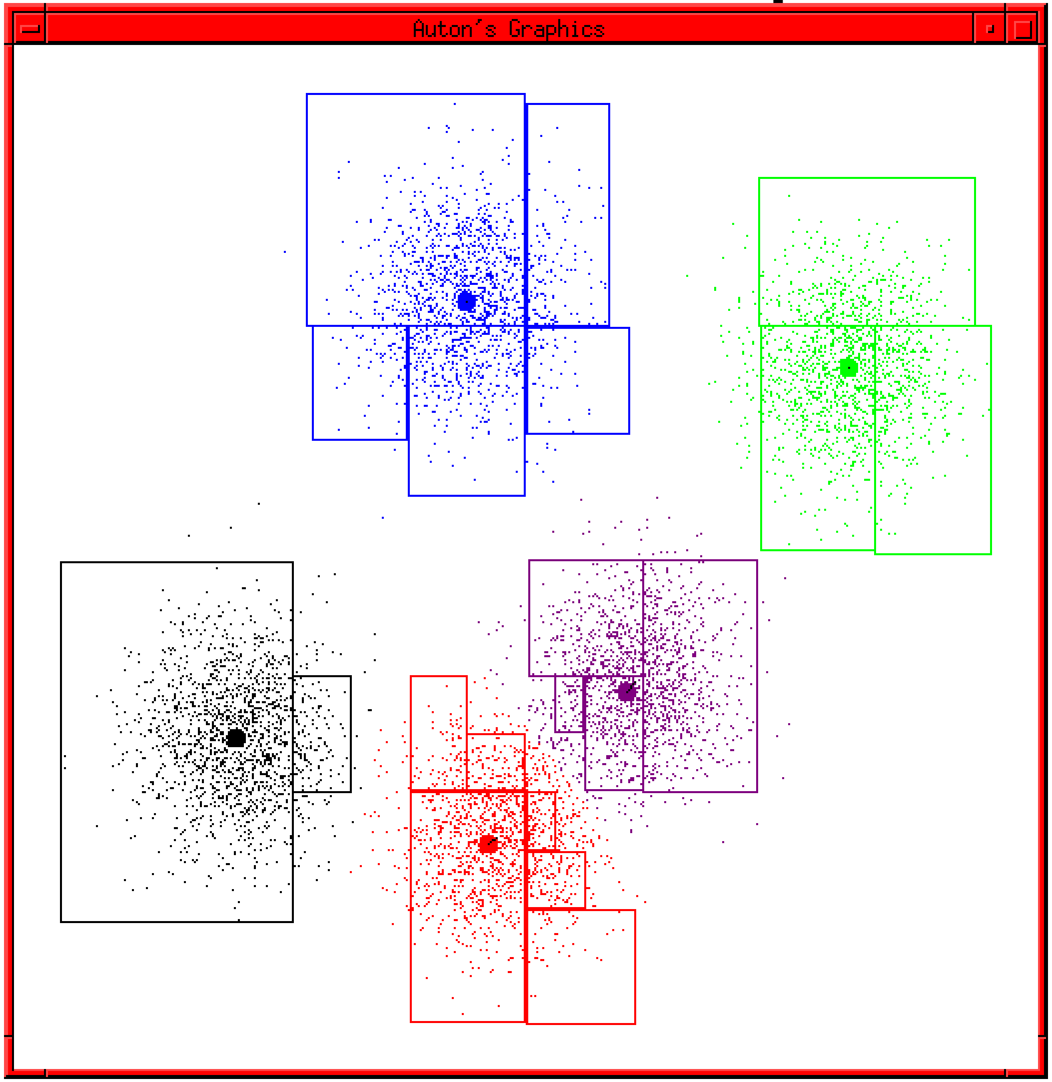
K-means in action



K-means in action



K-means stops



K-means algorithm

- Input: $x_1 \ldots x_n$, k
- **Step 1**: select k cluster centers $c_1 \ldots c_k$
- **Step 2**: for each point x, determine its cluster: find the closest center in Euclidean space
- **Step 3**: update all cluster centers as the centroids

$$c_i = \sum_{\{x \text{ in cluster } i\}} x \,/\, \text{SizeOf(cluster } i)$$

- Repeat step 2, 3 until cluster centers no longer change

Questions on k-means

- What is k-means trying to optimize?
- Will k-means stop (converge)?
- Will it find a global or local optimum?
- How to pick starting cluster centers?
- How many clusters should we use?

Distortion

- Suppose for a point x, you replace its coordinates by the cluster center $c_{(x)}$ it belongs to (lossy compression)
- How far are you off? Measure it with squared Euclidean distance: x(d) is the d-th feature dimension, y(x) is the cluster ID that x is in.

$$\Sigma_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$

- This is the distortion of a single point x. For the whole dataset, the distortion is

$$\Sigma_x \Sigma_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$

The minimization problem

$$\min \Sigma_x \Sigma_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$

$y(x_1)\ldots y(x_n)$
$c_1(1)\ldots c_1(D)$
…
$c_k(1)\ldots c_k(D)$

Step 1

- For fixed cluster centers, if all you can do is to assign x to some cluster, then assigning x to its closest cluster center y(x) minimizes distortion

$$\Sigma_{d=1 \dots D} [x(d) - c_{y(x)}(d)]^2$$

- Why? Try any other cluster $z \neq y(x)$

$$\Sigma_{d=1 \dots D} [x(d) - c_z(d)]^2$$

Step 2

- If the assignment of x to clusters are fixed, and all you can do is to change the location of cluster centers
- Then this is a continuous optimization problem!

$$\sum_x \sum_{d=1 \ldots D} [x(d) - c_{y(x)}(d)]^2$$

- Variables?

Step 2

- If the assignment of x to clusters are fixed, and all you can do is to change the location of cluster centers
- Then this is an optimization problem!
- Variables? $c_1(1)$, …, $c_1(D)$, …, $c_k(1)$, …, $c_k(D)$

$$\min \sum_x \sum_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$

$$= \min \sum_{z=1..k} \sum_{y(x)=z} \sum_{d=1\ldots D} [x(d) - c_z(d)]^2$$

- Unconstrained. What do we do?

Step 2

- If the assignment of x to clusters are fixed, and all you can do is to change the location of cluster centers
- Then this is an optimization problem!
- Variables? $c_1(1), \ldots, c_1(D), \ldots, c_k(1), \ldots, c_k(D)$

$$\min \sum_x \sum_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$

$$= \min \sum_{z=1..k} \sum_{y(x)=z} \sum_{d=1\ldots D} [x(d) - c_z(d)]^2$$

- Unconstrained.

$$\partial/\partial c_z(d) \sum_{z=1..k} \sum_{y(x)=z} \sum_{d=1\ldots D} [x(d) - c_z(d)]^2 = 0$$

Step 2

- The solution is

$$c_z(d) = \sum_{y(x)=z} x(d) \Big/ |n_z|$$

- The d-th dimension of cluster z is the average of the d-th dimension of points assigned to cluster z
- Or, update cluster z to be the centroid of its points. This is exact what we did in step 2.

Repeat (step1, step2)

- Both step1 and step2 minimizes the distortion
$$\sum_x \sum_{d=1\ldots D} [x(d) - c_{y(x)}(d)]^2$$
- Step1 changes x assignments y(x)
- Step2 changes c(d) the cluster centers
- However there is no guarantee the distortion is minimized over all… need to repeat
- This is hill climbing (coordinate descent)
- Will it stop?

Repeat (step1, step2)

- Both step1 and step2 r
- Step1 changes x assig
- Step2 changes c(d) th
- However there is no g
  repeat
- This is hill climbing (co
- Will it stop?

There are finite number of points

Finite ways of assigning points to clusters

In step1, an assignment that reduces distortion has to be a new assignment not used before

Step1 will terminate

So will step 2

So k-means terminates

What optimum does K-means find

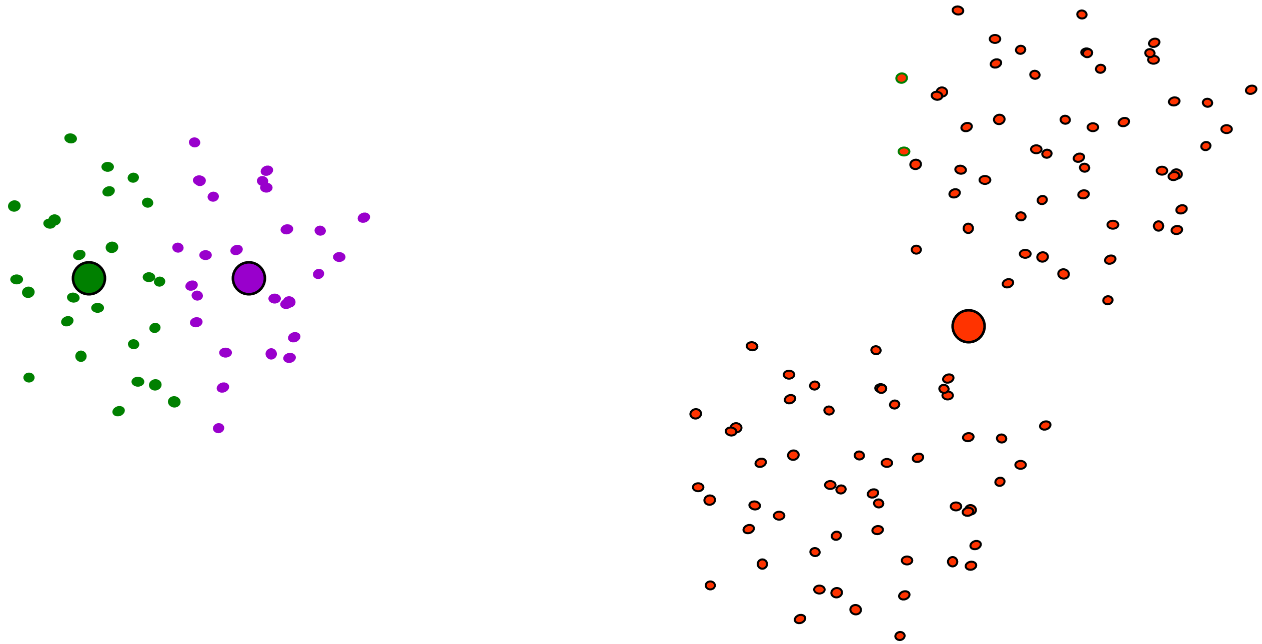
- Will k-means find the global minimum in distortion? Sadly no guarantee…
- Can you think of one example?

What optimum does K-means find

- Will k-means find the global minimum in distortion? Sadly no guarantee…
- Can you think of one example? (Hint: try k=3)

What optimum does K-means find

- Will k-means find the global minimum in distortion? Sadly no guarantee…
- Can you think of one example? (Hint: try k=3)



Picking starting cluster centers

- Which local optimum k-means goes to is determined solely by the starting cluster centers
  - Be careful how to pick the starting cluster centers. Many ideas. Here's one neat trick:
    1. Pick a random point x1 from dataset
    2. Find the point x2 farthest from x1 in the dataset
    3. Find x3 farthest from the closer of x1, x2
    4. … pick k points like this, use them as starting cluster centers for the k clusters
  - Run k-means multiple times with different starting cluster centers (hill climbing with random restarts)

Picking the number of clusters

- Difficult problem
- Domain knowledge?
- Otherwise, shall we find k which minimizes distortion?

Picking the number of clusters

- Difficult problem
- Domain knowledge?
- Otherwise, shall we find k which minimizes distortion? k = N, distortion = 0
- Need to regularize. A common approach is to minimize the Schwarz criterion

$$\text{distortion} + \lambda\ (\#\text{param})\ \log N$$

$$= \text{distortion} + \lambda\ \underbrace{D}_{\#\text{dimensions}}\ \underbrace{k}_{\#\text{clusters}}\ \log \underbrace{N}_{\#\text{points}}$$

Beyond k-means

- In k-means, each point belongs to one cluster
- What if one point can belong to more than one cluster?
- What if the degree of belonging depends on the distance to the centers?

- This will lead to the famous EM algorithm, or expectation-maximization
- K-means is a discrete version of EM algorithm with Gaussian mixture models with infinitely small covariances… (not covered in this class)