Introduction to Mobile Robotics

Clustering

Wolfram Burgard
Cyrill Stachniss
Giorgio Grisetti
Maren Bennewitz
Christian Plagemann

Clustering (1)

- Common technique for statistical data analysis (machine learning, data mining, pattern recognition, …)
- Classification of a data set into subsets (clusters)
- Ideally, data in each subset have a similar characteristics (proximity according to distance function)

Clustering (2)

- Needed: distance (similarity / dissimilarity) function, e.g., Euclidian distance
- Clustering quality
  - Inter-clusters distance maximized
  - Intra-clusters distance minimized
- The quality depends on
  - Clustering algorithm
  - Distance function
  - The application (data)

Types of Clustering

- Hierarchical Clustering
  - Agglomerative Clustering (buttom up)
  - Divisive Clustering (top-down)
- Partitional Clustering
  - K-Means Clustering (hard & soft)
  - Gaussian Mixture Models (EM-based)

K-Means Clustering

- Partitions the data into *k* clusters (k is to be specified by the user)
- Find *k* reference vectors $\boldsymbol{m}_j$, $j = 1,\ldots,k$ which best explain the data $\boldsymbol{X}$
- Assign data vectors to nearest (most similar) reference $\boldsymbol{m}_i$

$$\left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\|$$

r-dimensional data vector in a real-valued space

reference vector (center of cluster = mean)

Reconstruction Error
(K-Means as Compression Alg.)

- The total reconstruction error is defined as

$$E\left(\{\mathbf{m}_i\}_{i=1}^{k} \middle| \mathcal{X}\right) = \sum_t \sum_i b_i^t \left\| \mathbf{x}^t - \mathbf{m}_i \right\|^2$$

with $$b_i^t = \begin{cases} 1 & \text{if } \left\| \mathbf{x}^t - \mathbf{m}_i \right\| = \min_j \left\| \mathbf{x}^t - \mathbf{m}_j \right\| \\ 0 & \text{otherwise} \end{cases}$$

- Find reference vectors which minimize the error
- Taking its derivative with respect to m_i and setting it to 0 leads to

$$\mathbf{m}_i = \frac{\sum_t b_i^t \mathbf{x}^t}{\sum_t b_i^t}$$

K-Means Algorithm

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge

Recompute the cluster centers $\boldsymbol{m}_i$ using current cluster membership

Assign each $\boldsymbol{x}^t$ to the closest cluster

K-Means Example

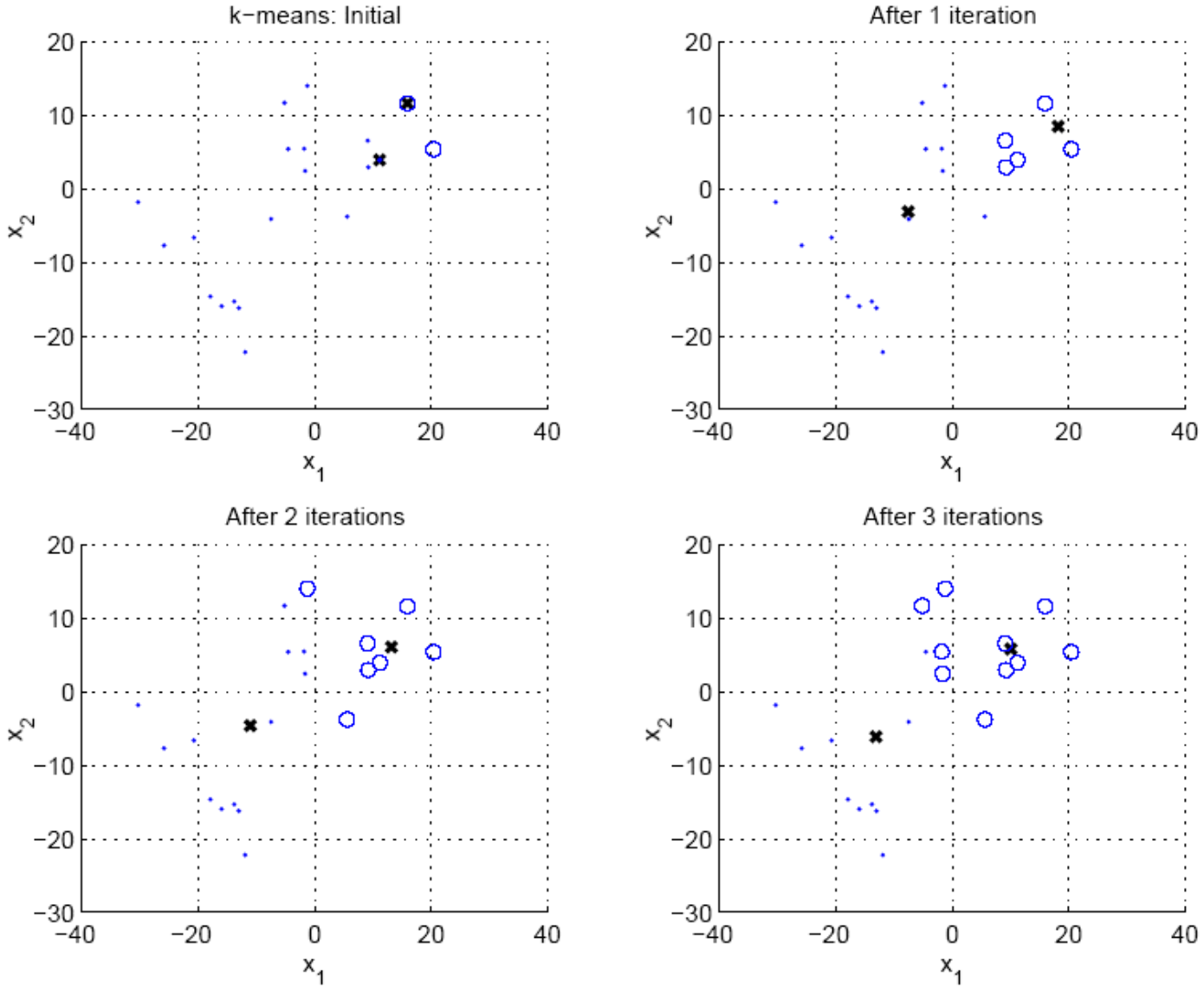


Image source: Alpaydin, Introduction to Machine Learning

Strength of K-Means

- Easy to understand and to implement
- Efficient O(nkt)
  n = #iterations, k = #clusters, t = #data points
- Converges to a local optimum (global optimum is hard to find)
- Most popular clustering algorithm

Weaknesses of K-Means

- User needs to specify #clusters (*k*)
- Sensitive to initialization (strategy: use different seeds)
- Sensitive to outliers since all data points contribute equally to the mean (strategy: try to eliminate outliers)

Soft Assignments

- So far, each data point was assigned to exactly one cluster
- A variant called soft k-means allows for making fuzzy assignments
- Data points are assigned to clusters with certain probabilities

Soft K-Means Clustering

- Each data point is given a soft assignment to all means

$$c_{tk} = \frac{\exp(-\beta \, ||x^t - m_k||^2)}{\sum_i \exp(-\beta \, ||x^t - m_i||^2)}, \; \sum_k c_{tk} = 1$$

- β is a "stiffness" parameter and plays a crucial role
- Means are updated

$$m_k = \frac{\sum_t c_{tk} x^t}{\sum_t c_{tk}}$$

- Repeat assignment and update step until assignments do not change anymore

Soft K-Means Clustering

- Points between clusters get assigned to both of them
- Points near the cluster boundaries play a partial role in several clusters
- Additional parameter β
- Clusters with varying shapes can be treated in a probabilistic framework (mixtures of Gaussians)

Similarity of Soft K-Means and Expectation Maximization (EM)

- Goal of EM: Find component parameters that maximize the likelihood of dataset
- Two sets of random variables:
  - observed data set *d*
  - hidden variables *c* (assignment of data points to clusters)
- Since the complete data likelihood cannot determined, work with its expectation

Expected Data Likelihood

- Observed data $d = \{d_1, \ldots, d_I\}$
- Correspondence variables (hidden)

$$c = \{c_1, \ldots, c_I\}$$

- Joint likelihood of *d* and *c* given model θ

$$P(d, c \mid \theta) = \prod_{i=1}^{I} P(d_i, c_i \mid \theta)$$
$$\ln P(d, c \mid \theta) = \sum_{i=1}^{I} \ln P(d_i, c_i \mid \theta)$$

- Since the values of *c* are hidden, optimize the expected log likelihood

$$E_c[\ln P(d, c \mid \theta) \mid \theta, d] = E_c[\sum_{i=1}^{I} \ln P(d_i, c_i \mid \theta) \mid \theta, d]$$

Expectation Maximization

- Optimizing the expected log likelihood is usually not easy
- EM iteratively maximizes log likelihood functions
- EM generates a sequence of models $\theta^{[1]}, \theta^{[2]}, \ldots$ of increasing log likelihood

Expectation Maximization

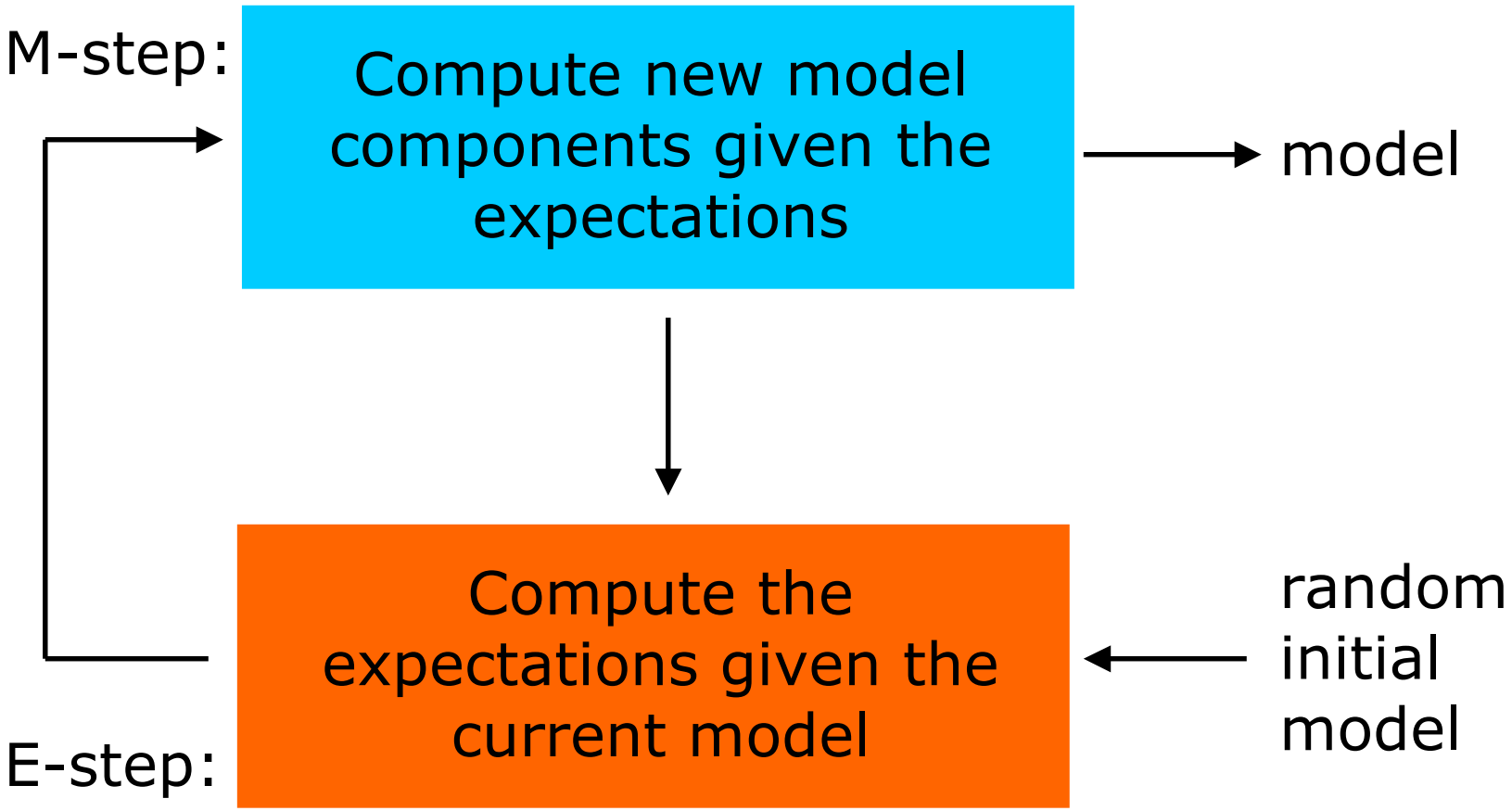
- Use so-called Q-function to find the model with the maximum expected data likelihood
- Estimation (E) step
  - Compute expected values for *c* given current model $\theta^{[j]}$
  - Define the expected data log likelihood as a function of θ

$$Q(\theta \mid \theta^{[j]}) = E_c[\ln P(d, c \mid \theta) \mid \theta^{[j]}, d]$$

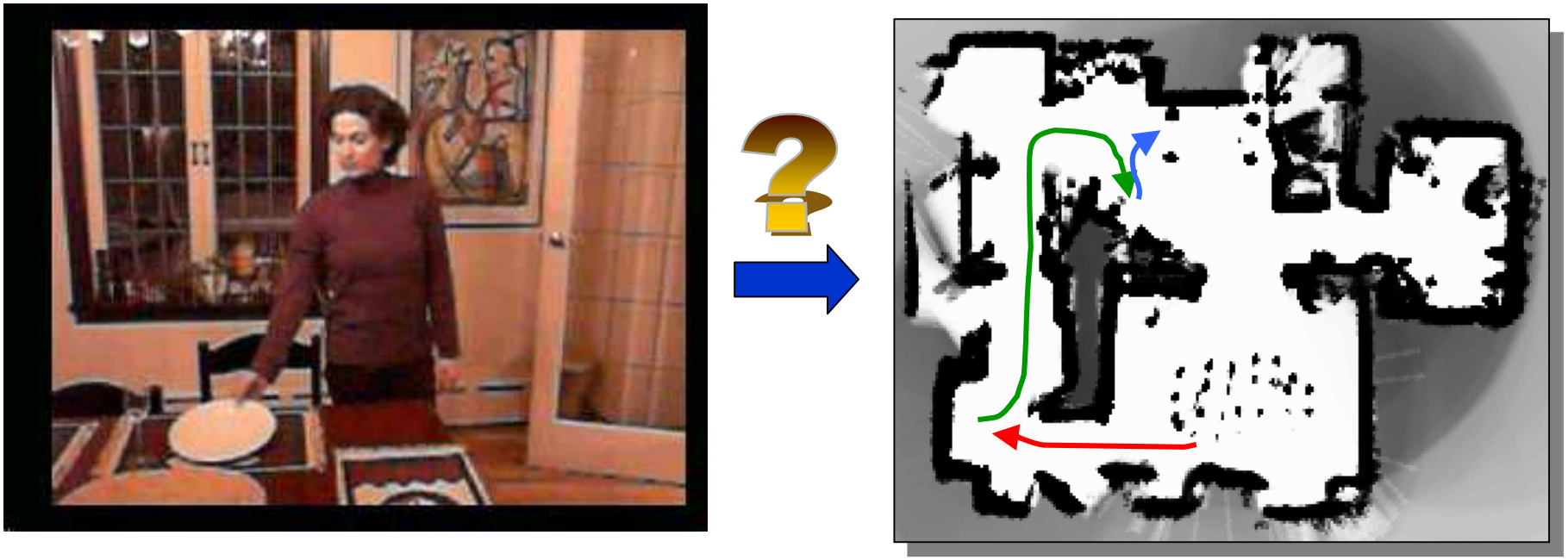
- Maximization (M) step
  - Maximize this expected likelihood

$$\theta^{[j+1]} = \underset{\theta'}{\operatorname{argmax}}\, Q(\theta' \mid \theta^{[j]})$$

Iterating E and M Steps



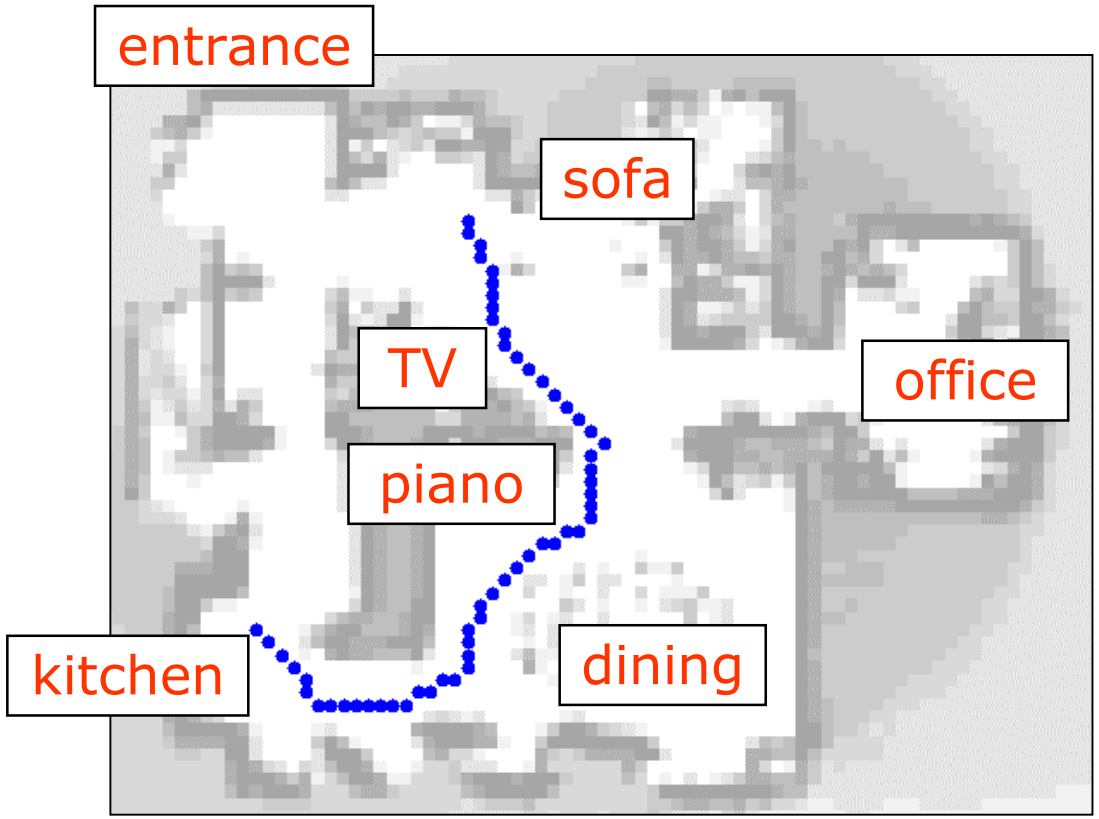
Application: Trajectory Clustering



How to learn typical motion patterns of people from observations?

Application: Trajectory Clustering

- Input: Set of trajectories $d_1,\ldots,d_I$



$$d_i = \{x_i^1, x_i^2, \ldots, x_i^T\}$$

What we are looking for

- Clustering of similar trajectories into motion patterns $\theta_1, \ldots, \theta_M$
  (Note: From now on M = #clusters)
- Binary correspondence variables c_{im} indicating which trajectory s_i belongs to which motion pattern θ_m

Problem:

How can we estimate c_{im}?

Motion Patterns

- Use T Gaussians with fixed variance to represent each motion pattern
- If we knew the values of the c_{im}, the computation of the motion patterns would be easy
- But: These values are hidden
- Use Expectation Maximization to compute
  - expected values for the c_{im}
  - the model θ (i.e., the set of motion patterns) which has the highest expected data likelihood

Likelihood of Trajectory d_i under Motion Pattern $\theta_m=\{\theta^1_m, \ldots, \theta^T_m\}$

$$P(d_i \mid \theta_m) = \prod_{t=1}^{T} P(x_i^t \mid \theta_m^t)$$

$$= \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}\|x_i^t - \mu_m^t\|^2)$$

probability that the person is at location x_i^t after t observations given it is engaged in motion pattern θ_m

Data Likelihood

- Joint likelihood of a single trajectory and its correspondence vector

$$P(d_i, c_i \mid \theta) = \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi}\sigma} \prod_{m=1}^{M} \exp(-\frac{1}{2\sigma^2} c_{im} \|x_i^t - \mu_m^t\|^2)$$

- Expected data log likelihood

$$E_c[\ln P(d, c \mid \theta) \mid \theta, d]$$

$$= E_c\Big[\sum_{i=1}^{I} \ln \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi}\sigma} \prod_{m=1}^{M} \exp(-\frac{1}{2\sigma^2} c_{im} \|x_i^t - \mu_m^t\|^2) \mid \theta, d\Big]$$

$$= E_c\Big[I \cdot T \cdot \ln \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^{I} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{im} \|x_i^t - \mu_m^t\|^2 \mid \theta, d\Big]$$

Q-Function

(Expectation is a linear operator, move it inside the sum)

$$Q(\theta' \mid \theta) \;=\; const - \frac{1}{2\sigma^2} \sum_{i=1}^{I} \sum_{m=1}^{M} E[c_{im} \mid \theta, d] \sum_{t=1}^{T} \|x_i^t - \mu_m'^t\|^2$$

E-Step: Compute the Expectations Given the Current Model $\theta^{[j]}$

$$
\begin{aligned}
E[c_{im} \mid \theta^{[j]}, d] &= P(c_{im} \mid \theta^{[j]}, d) \\
&= P(c_{im} \mid \theta^{[j]}, d_i) \\
&\overset{\text{Bayes'}}{=} \eta P(d_i \mid c_{im}, \theta^{[j]}) P(c_{im} \mid \theta^{[j]}) \\
&\overset{\text{uniform prior}}{=} \eta' P(d_i \mid \theta_m^{[j]}) \\
&= \eta'' \prod_{t=1}^{T} \exp\left(-\frac{1}{2\sigma^2} \| x_i^t - \mu_m^{t[j]} \|^2\right)
\end{aligned}
$$

normalizer

likelihood that the i-th trajectory belongs to the m-th model component

M-Step: Maximize the Expected Likelihood

$$
\begin{aligned}
&\theta^{[j+1]} \\
&\quad = \operatorname*{argmax}_{\theta'} Q(\theta' \mid \theta^{[j]}) \\
&\quad = \operatorname*{argmax}_{\theta'} \Big\{ const - \frac{1}{2\sigma^2} \sum_{i=1}^{I} \sum_{m=1}^{M} E[c_{im} \mid \theta, d] \sum_{t=1}^{T} \|x_i^t - \mu_m'^t\|^2 \Big\} \\
&\quad = \operatorname*{argmin}_{\theta'} \Big\{ \sum_{i=1}^{I} \sum_{m=1}^{M} E[c_{im} \mid \theta^{[j]}, d] \sum_{t=1}^{T} \|x_i^t - \mu_m'^t\|^2 \Big\}.
\end{aligned}
$$

Compute partial derivative with respect to μ_m^t

M-Step: Maximize the Expected Likelihood

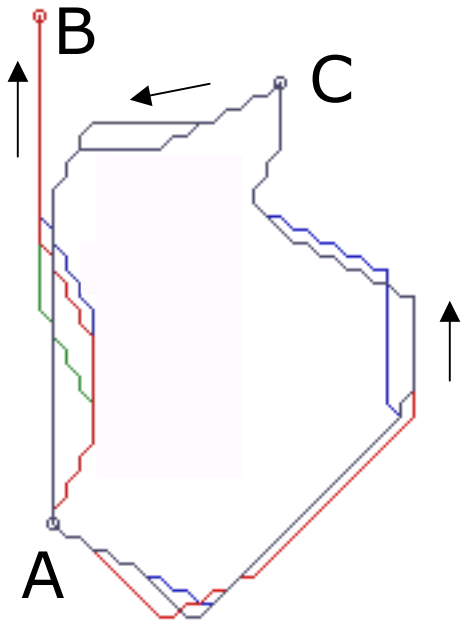
$$\sum_{i=1}^{I} E[c_{im} \mid \theta^{[j]}, d] \cdot 2 \cdot (x_i^t - \mu_m^{t[j+1]}) \stackrel{!}{=} 0 \iff$$

$$\sum_{i=1}^{I} E[c_{im} \mid \theta^{[j]}, d]\, x_i^t \stackrel{!}{=} \sum_{i=1}^{I} E[c_{im} \mid \theta^{[j]}, d]\, \mu_m^{t[j+1]} \iff$$

$$\mu_m^{t[j+1]} \stackrel{!}{=} \frac{\sum_{i=1}^{I} E[c_{im} \mid \theta^{[j]}, d]\, x_i^t}{\sum_{i=1}^{I} E[c_{im} \mid \theta^{[j]}, d]}$$

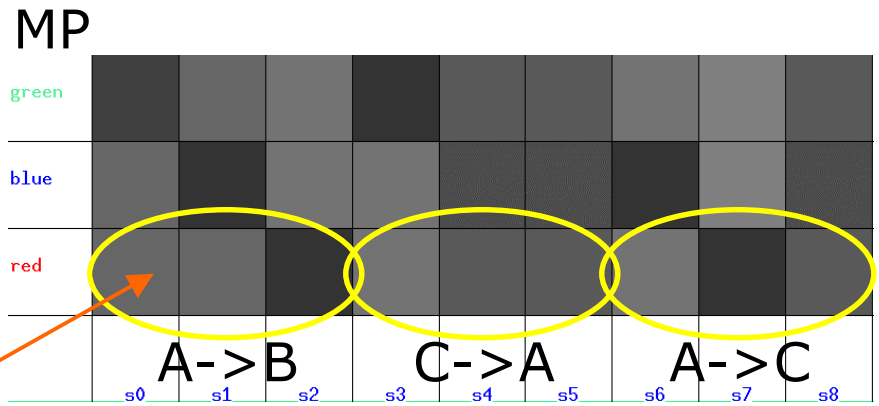
This is the mean update of soft k-means

EM Application Example:
9 Trajectories of 3 Motion Patterns



EM: Example (step 1)

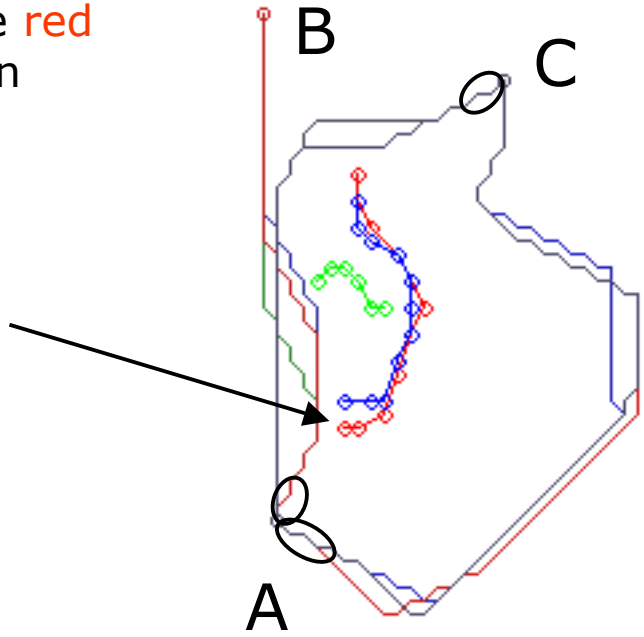
C_{im}:



trajectories

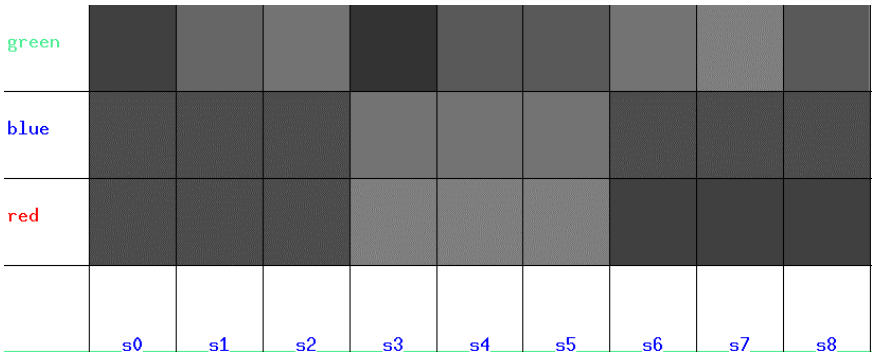
likelihood that s_0 belongs to the red motion pattern

θ^1:

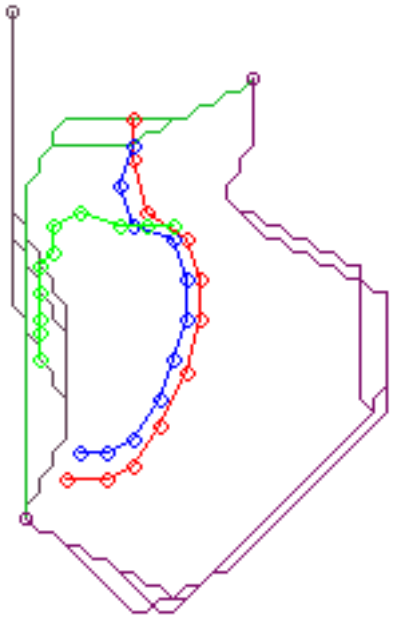


EM: Example (step 2)

C_{im}:

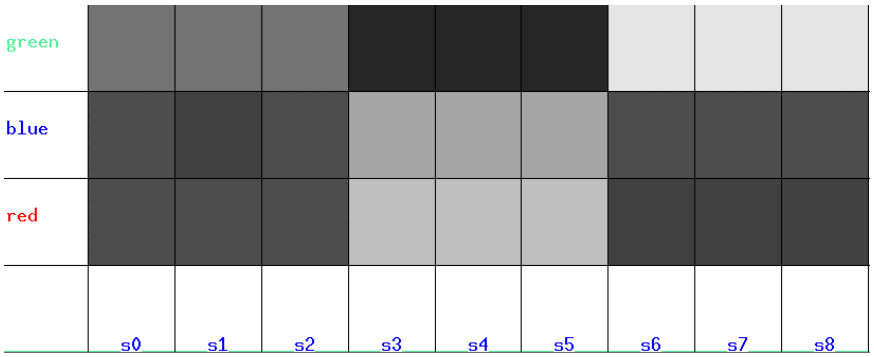


θ^2:

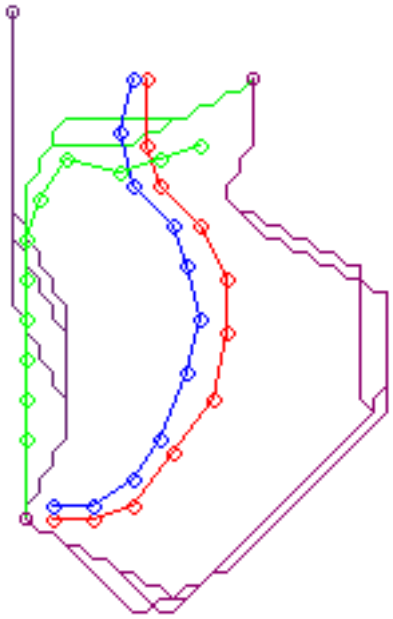


EM: Example (step 3)

C_{im}:

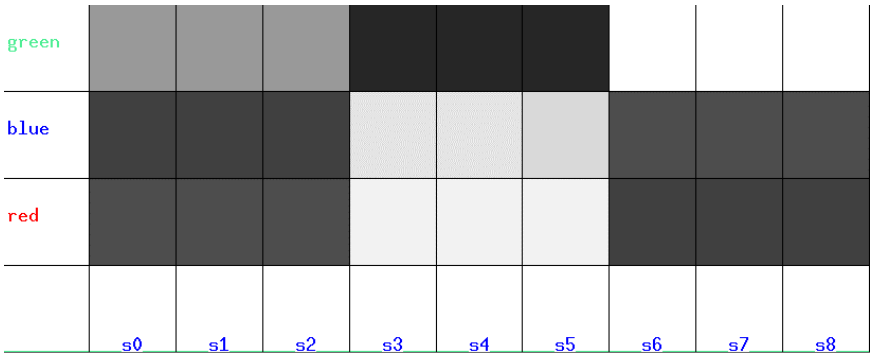


θ^3:

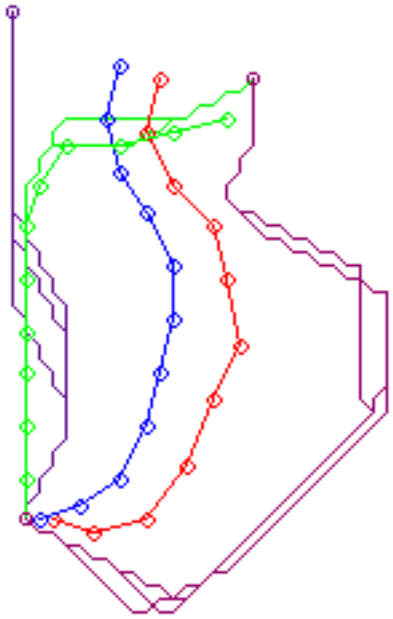


EM: Example (step 4)

C_{im}:

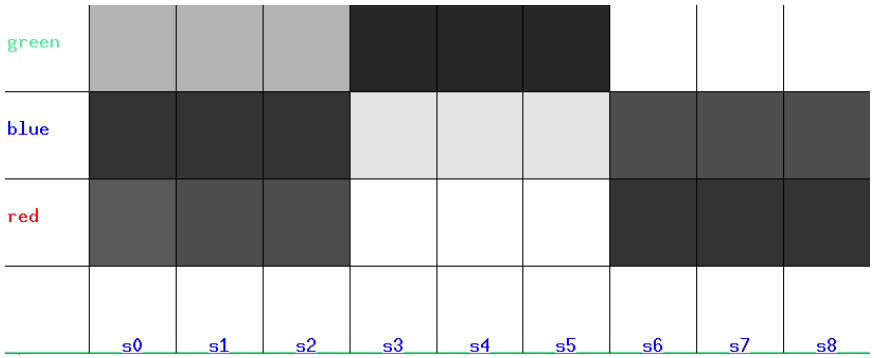


θ^4:

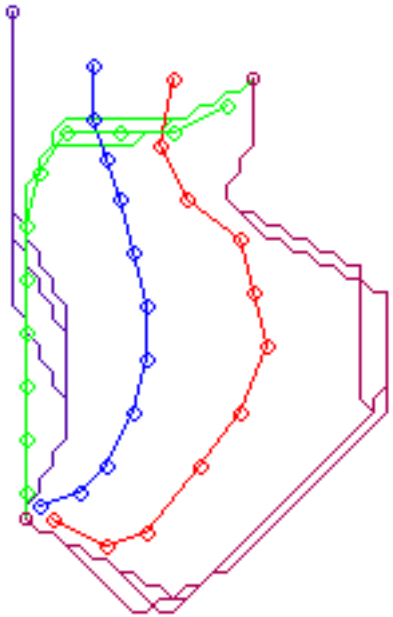


EM: Example (step 5)

C_{im}:

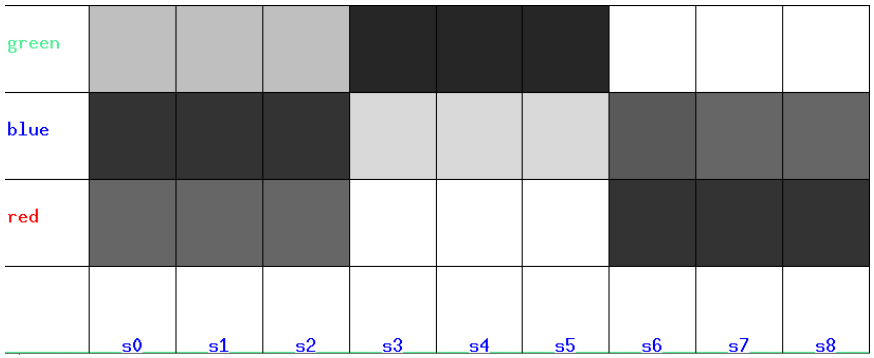


θ^5:

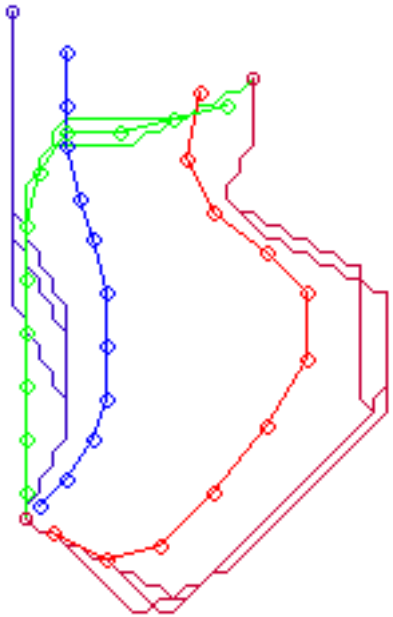


EM: Example (step 6)

C_{im}:

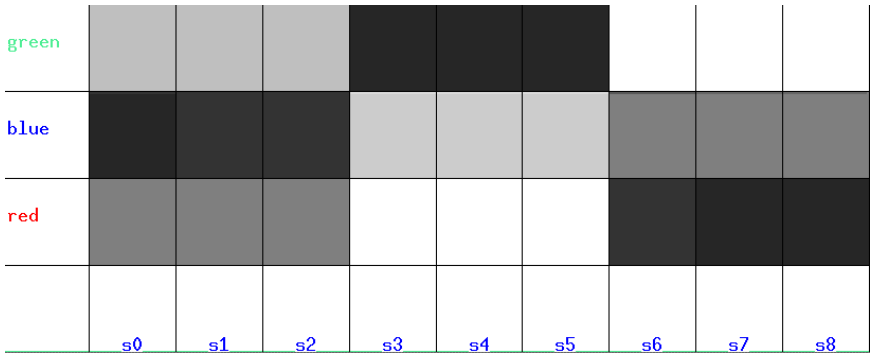


θ^6:

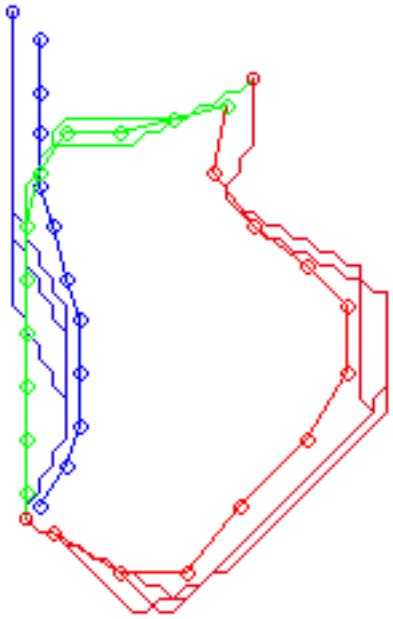


EM: Example (step 7)

C_{im}:

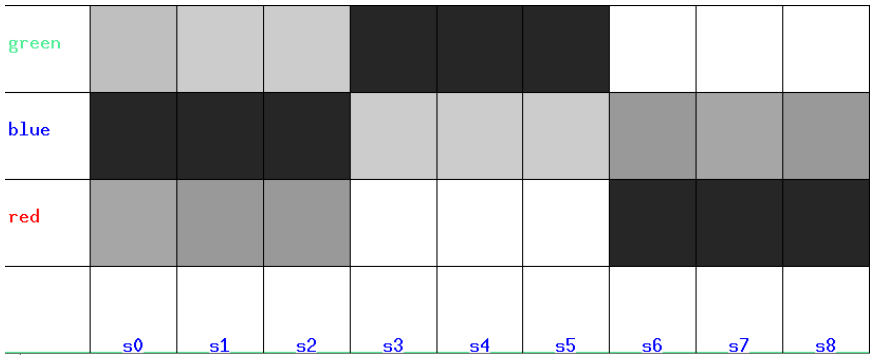


θ^7:

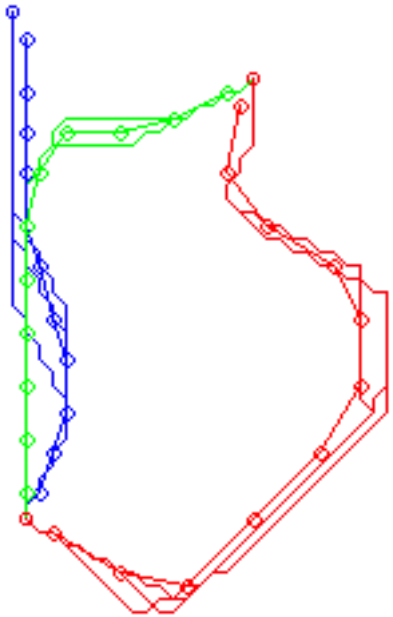


EM: Example (step 8)

C_{im}:

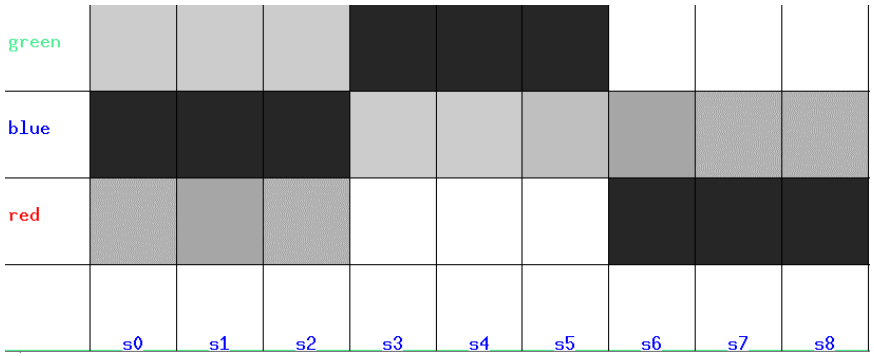


θ^8:

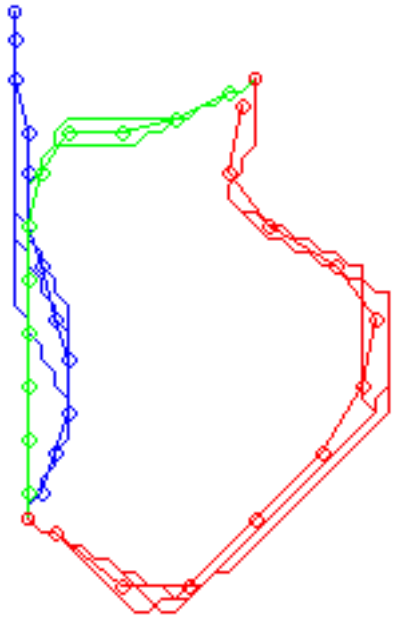


EM: Example (step 9)

C_{im}:



θ^9:



Estimating the Number of Model Components

After convergence of the EM check whether the model can be improved

- by introducing a new model component for the trajectory which has the lowest likelihood or
- by eliminating the model component which has the lowest utility.

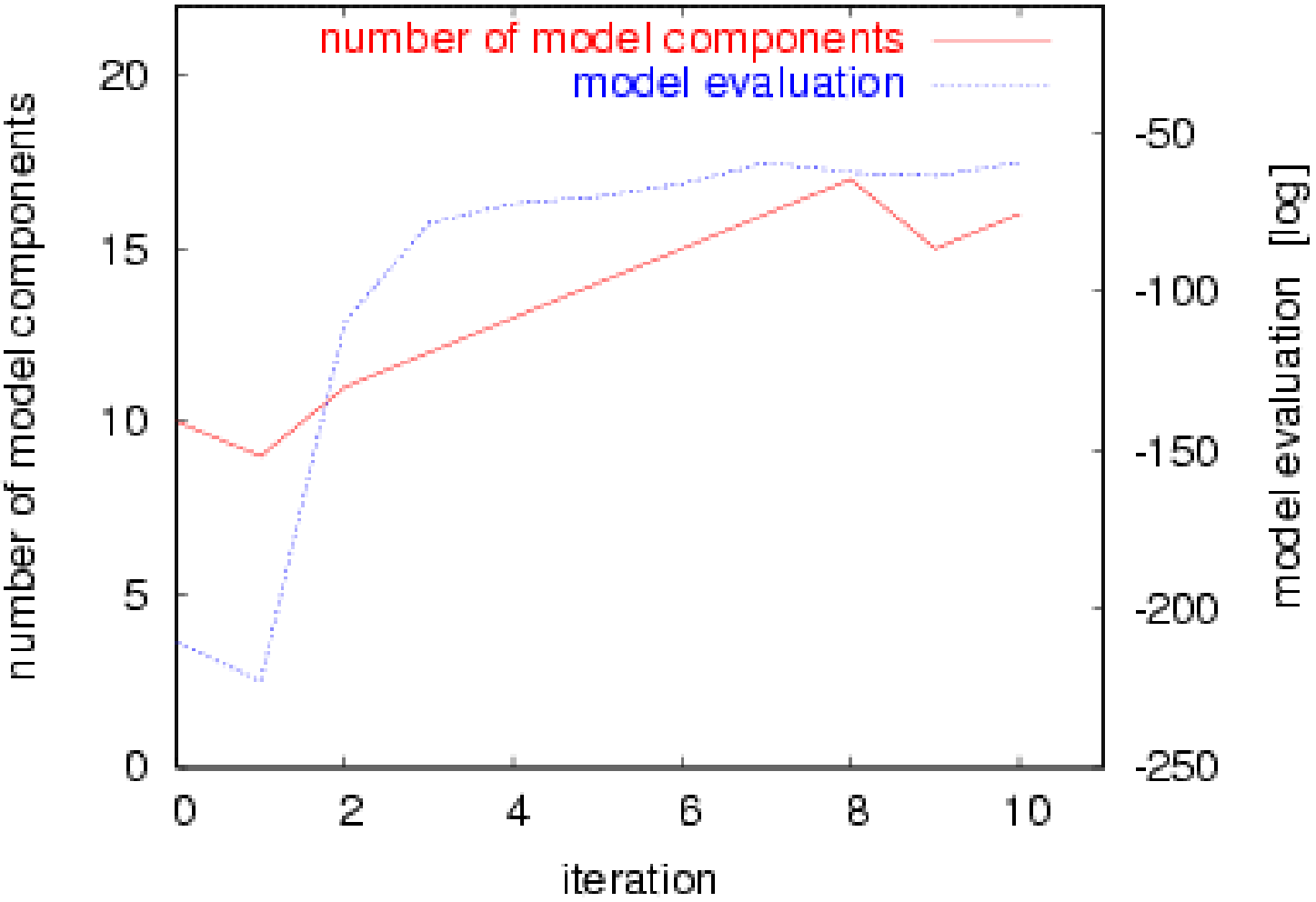
Select model θ which has the **highest evaluation**

$$E_c[\log P(d, c \mid \theta) \mid \theta, d] - \frac{M}{2} \log I$$

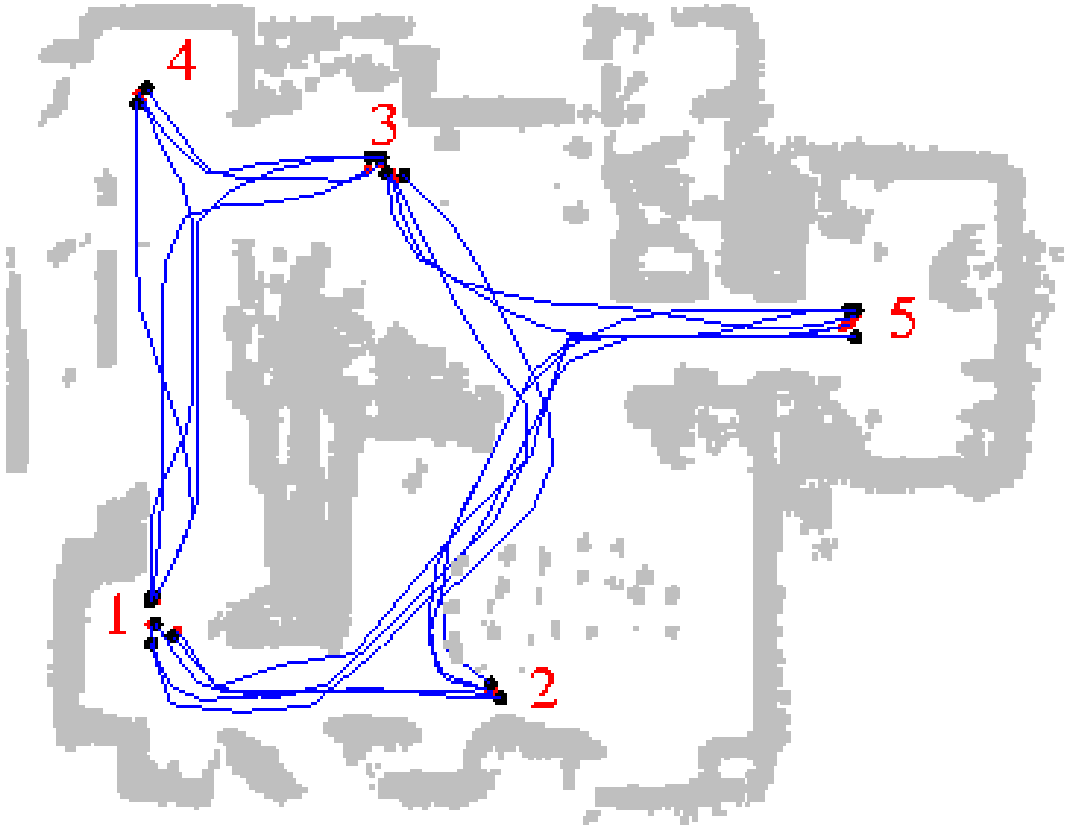
where M = #model components, I = #trajectories

Bayesian Information Criterion [Schwarz,`78]

Application Example



Learned Motion Patterns



Summary

- K-Means is the most popular clustering algorithm
- It is efficient and easy to implement
- Converges to a local optimum
- A variant of hard k-means exists allowing soft assignments
- Soft k-means corresponds to the EM algorithm which is a general optimization procedure

Further Reading

E. Alpaydin

Introduction to Machine Learning