Learning Issues in & Image Segmentation

Joachim M. Buhmann
jbuhmann@inf.ethz.ch

Swiss Federal Institute of Technology (ETH Zurich)
Institute for Computational Science, HRS-F31
8092 Zurich, Switzerland

The Problem of Data Clustering

Given: Set of n objects $x_1, \ldots, x_n$ (e. g. points in Euclidean space).

Clustering problem: Group $x_1, \ldots, x_n$ into k groups of similar objects. These groups are called *clusters*.

The Problem of Data Clustering

Given: Set of n objects $x_1, \ldots, x_n$ (e. g. points in Euclidean space).

Clustering problem: Group $x_1, \ldots, x_n$ into k groups of similar objects. These groups are called *clusters*.

Note: The number k of clusters is usually predefined, i. e. an input parameter.
The similarity measure depends on the problem.

Application: Image segmentation

Image segmentation problem:

Decompose a given image into *segments*, i. e. regions containing similar pixels.

Application: Image segmentation

Image segmentation problem:

Decompose a given image into *segments*, i. e. regions containing similar pixels.

Example: Segments might be regions of the image depicting the same object.

Application: Image segmentation

Image segmentation problem:

Decompose a given image into *segments*, i. e. regions containing similar pixels.

Example: Segments might be regions of the image depicting the same object.

Semantics Problem: *How should we infer objects from segments?*

Clustering Approach to Image Segmentation

Clustering objects: Pixel color/greyscale values or local image statistics (histograms).

Clustering Approach to Image Segmentation

Clustering objects: Pixel color/greyscale values or local image statistics (histograms).

Method: Apply appropriate clustering algorithm to image data.

Clustering Approach to Image Segmentation

Clustering objects: Pixel color/greyscale values or local image statistics (histograms).

Method: Apply appropriate clustering algorithm to image data.

Result: Segments are connected regions assigned to the same cluster.

Problem Formalization

Notation:

- Objects: Data set $(x_1, \ldots, x_n) =: \mathbf{x}$
- Clusters: $\mathcal{C}_1, \ldots, \mathcal{C}_k$

Problem Formalization

Notation:

- Objects: Data set $(x_1, \ldots, x_n) =: \mathbf{x}$
- Clusters: $\mathcal{C}_1, \ldots, \mathcal{C}_k$
- **Encoding assignments:** Use *assignment vector* $\mathbf{c} \in \{1, \ldots, k\}^n$, where

$$c_i = \alpha \;\Leftrightarrow\; x_i \in \mathcal{C}_\alpha \; .$$

Problem Formalization

Notation:

- Objects: Data set $(x_1, \ldots, x_n) =: \mathbf{x}$
- Clusters: $\mathcal{C}_1, \ldots, \mathcal{C}_k$
- **Encoding assignments:** Use *assignment vector* $\mathbf{c} \in \{1, \ldots, k\}^n$, where

$$c_i = \alpha \;\Leftrightarrow\; x_i \in \mathcal{C}_\alpha \; .$$

Clustering solutions are defined by instances of the assignment vector $\mathbf{c}$.

Cost Function Idea

Problem: Clustering problem (“group similar objects”) requires a notion of similarity.

Cost Function Idea

Problem: Clustering problem ("group similar objects") requires a notion of similarity.

Approach: Formalize similarity in terms of a cost function H, which assigns cost values to assignments.

$$H : \{1, \ldots, k\}^n \rightarrow \mathbb{R}_{\geq 0}$$

Cost Function Idea

Problem: Clustering problem ("group similar objects") requires a notion of similarity.

Approach: Formalize similarity in terms of a cost function H, which assigns cost values to assignments.

$$H : \{1, \ldots, k\}^n \rightarrow \mathbb{R}_{\geq 0}$$

Interpretation: Assignment of *dissimilar objects* to the same cluster produces *high costs*.

Cost Function-Based Clustering

Algorithmic solution: Compute assignment $\mathbf{c}^*$ for which costs $H(\mathbf{c})$ are minimal:

$$\mathbf{c}^* = \underset{\mathbf{c} \in \{1,\ldots,k\}^n}{\operatorname{argmin}} \; H(\mathbf{c}|\mathbf{x})$$

Cost Function-Based Clustering

Algorithmic solution: Compute assignment $\mathbf{c}^*$ for which costs $H(\mathbf{c})$ are minimal:

$$\mathbf{c}^* = \underset{\mathbf{c} \in \{1,\ldots,k\}^n}{\operatorname{argmin}} \; H(\mathbf{c}|\mathbf{x})$$

Two basic problems:

1) Choice of cost function.

Cost Function-Based Clustering

Algorithmic solution: Compute assignment $\mathbf{c}^*$ for which costs $H(\mathbf{c})$ are minimal:

$$\mathbf{c}^* = \underset{\mathbf{c}\in\{1,\ldots,k\}^n}{\operatorname{argmin}} H(\mathbf{c}|\mathbf{x})$$

Two basic problems:

1) Choice of cost function. 2) Algorithmic optimization.

Cost Function-Based Clustering

Algorithmic solution: Compute assignment $\mathbf{c}^*$ for which costs $H(\mathbf{c})$ are minimal:

$$\mathbf{c}^* = \underset{\mathbf{c}\in\{1,\ldots,k\}^n}{\operatorname{argmin}} \; H(\mathbf{c}|\mathbf{x})$$

Two basic problems:

1) Choice of cost function. 2) Algorithmic optimization.

Advantage of cost function approach: Attempts to separate *model design* from *algorithmic issues*.

Cost Function Optimization

Trade-off problem

- *Simple cost functions:* Easy to optimize, but not suitable to capture complex dependencies.
- *Complex cost functions:* Usually hard to optimize, due to local minima.

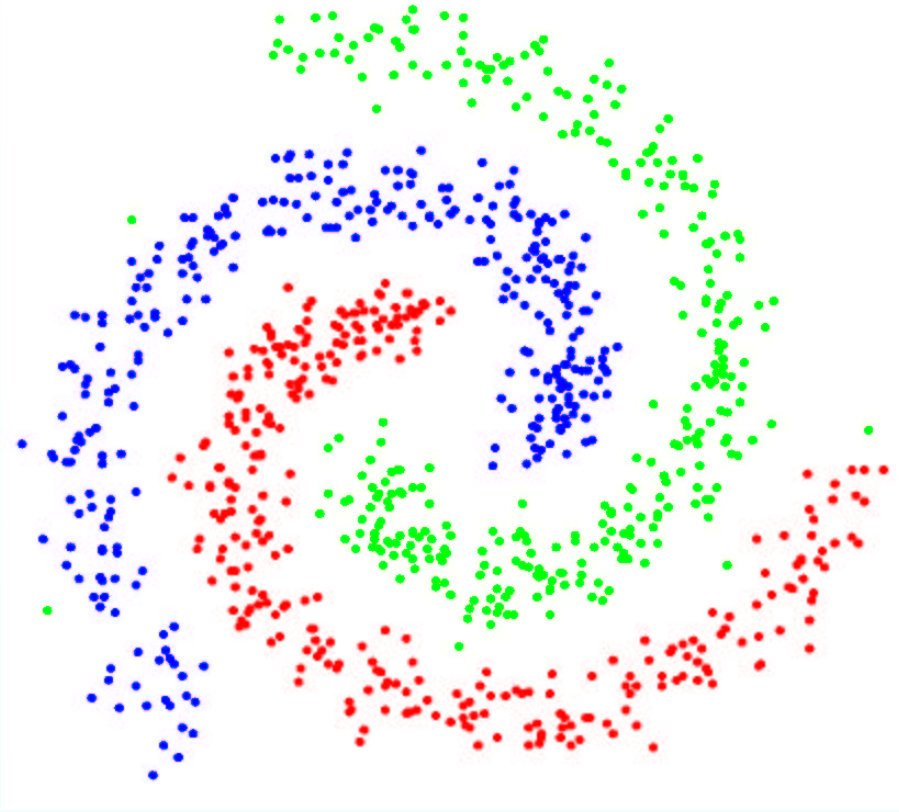
Cost Function Optimization

Trade-off problem

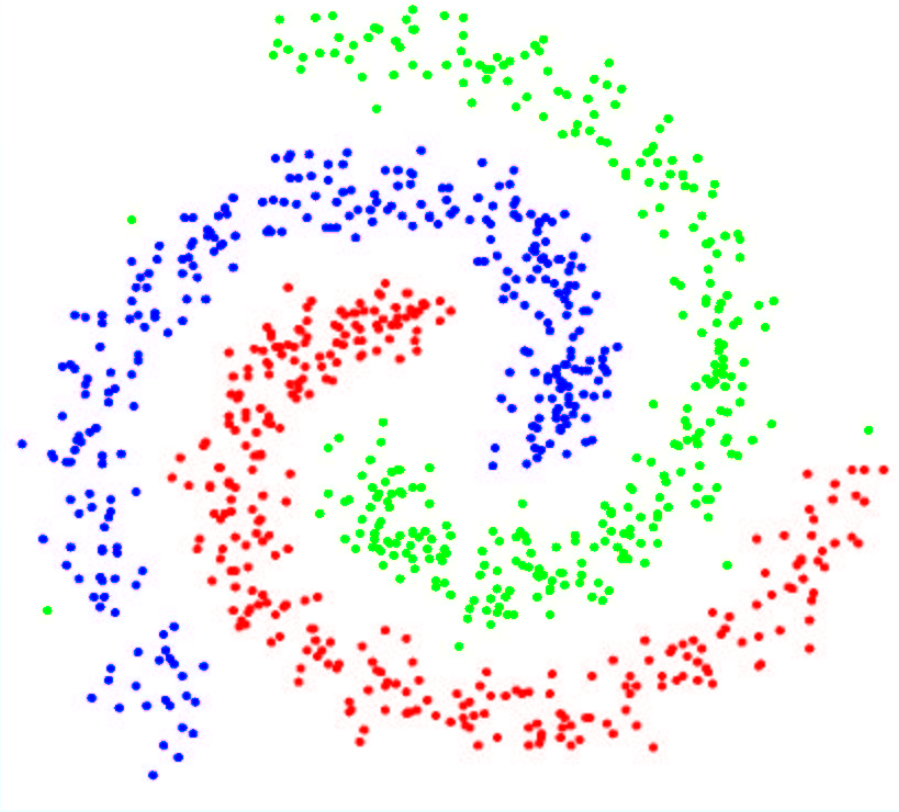
- *Simple cost functions:* Easy to optimize, but not suitable to capture complex dependencies.
- *Complex cost functions:* Usually hard to optimize, due to local minima.

Example: A cost function with simple structure may e. g. be of the form $H\left(\mathbf{c}|\mathbf{x}\right) = \sum_{i=1}^{n} f\left(c_i|x_i\right)$. Costs are evaluated separately for each object.

Extension: incorporate neighborhood information in cluster assignment



Extension: incorporate neighborhood information in cluster assignment



$\Rightarrow$ Object-wise evaluation of costs not sufficient, more complex cost functions required.

Unsupervised vs. supervised

Types of problems: Learning algorithms may be roughly divided into two classes: *Supervised* and *unsupervised* methods.

Unsupervised vs. supervised

Types of problems: Learning algorithms may be roughly divided into two classes: *Supervised* and *unsupervised* methods.

Supervised methods: Methods which are “trained” on a sample of labeled examples. “Training” describes e. g. the adjustment of cost function or classifier parameters. These are typically classification methods.

Unsupervised vs. supervised

Types of problems: Learning algorithms may be roughly divided into two classes: *Supervised* and *unsupervised* methods.

Supervised methods: Methods which are “trained” on a sample of labeled examples. “Training” describes e. g. the adjustment of cost function or classifier parameters. These are typically classification methods.

Clustering: Unsupervised methods; no previously labeled data available.

What is Data Clustering?

Problem Specification: What is given?

- n objects and an object space
- a quality criterion to partition the object space

What is Data Clustering?

Problem Specification: What is given?

- n objects and an object space
- a quality criterion to partition the object space

Classification: Partitioning of object space is exemplarily defined for training objects by supervisor.

How to generalize the partition to new objects?

What is Data Clustering?

Problem Specification: What is given?

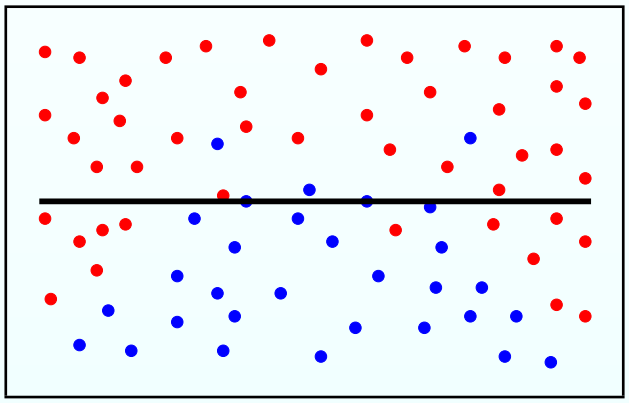
- n objects and an object space
- a quality criterion to partition the object space

Classification: Partitioning of object space is exemplarily defined for training objects by supervisor.
How to generalize the partition to new objects?

Clustering: Unsupervised partitioning of object space by quality criterion! **How to optimize cluster criterion?**

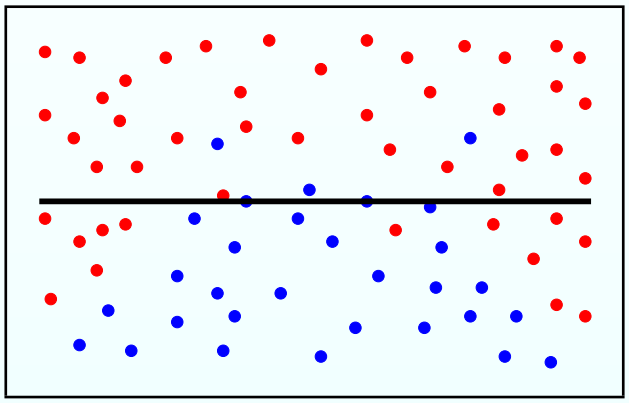
Generalization Problem in Classification

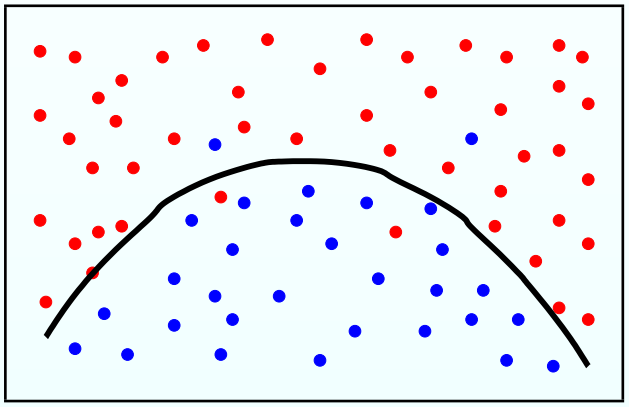
Underfitting



Generalization Problem in Classification

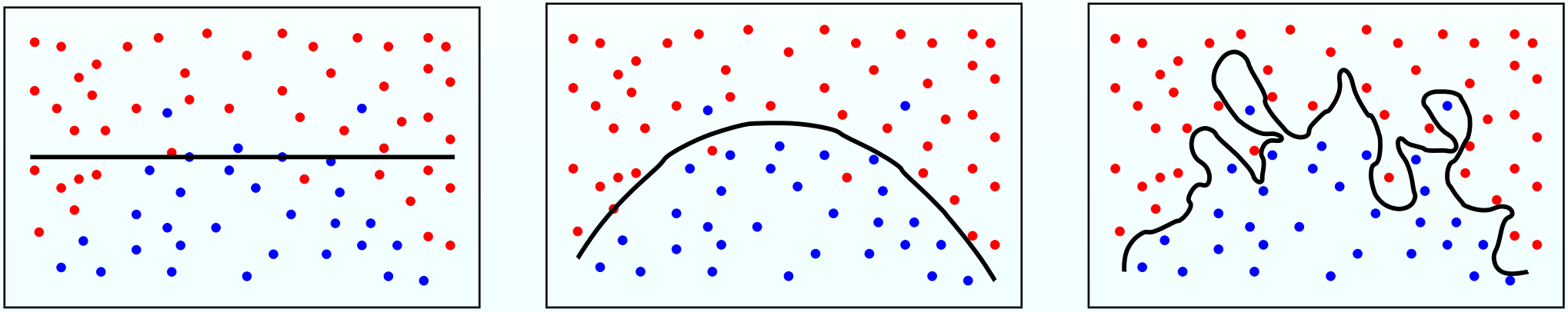
Underfitting $\longleftrightarrow$





Generalization Problem in Classification

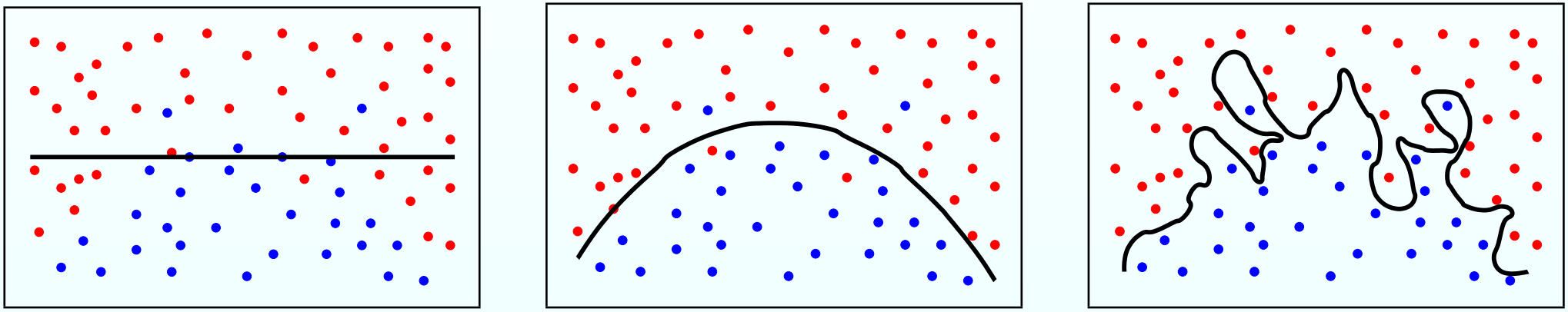
Underfitting $\longleftrightarrow$ Overfitting



Complexity of Hypothesis Class has to be **controlled**, e.g., restricted to avoid overfitting.

Generalization Problem in Classification

Underfitting $\longleftrightarrow$ Overfitting



Complexity of Hypothesis Class has to be **controlled**, e.g., restricted to avoid overfitting. **Key Problem in ML,**

Statistical and Computational Learning Theory

Structure of the Tutorial

Part 1: **Basic Concepts** of data clustering

- K-means clustering, histogram/distributional clustering
- graph theoretic approaches: pairwise clustering, NCut
- path-based clustering and perceptual organization

Structure of the Tutorial

Part 1: **Basic Concepts** of data clustering

- K-means clustering, histogram/distributional clustering
- graph theoretic approaches: pairwise clustering, NCut
- path-based clustering and perceptual organization

Part 2: **Optimization** of clustering cost functionals

- stochastic optimization, simulated and deterministic annealing
- multiscale optimization

Structure of the Tutorial

Part 1: **Basic Concepts** of data clustering

- K-means clustering, histogram/distributional clustering
- graph theoretic approaches: pairwise clustering, NCut
- path-based clustering and perceptual organization

Part 2: **Optimization** of clustering cost functionals

- stochastic optimization, simulated and deterministic annealing
- multiscale optimization

Part 3: **Validation** of clustering solutions

- agreement measure
- gap statistic
- stability analysis

Data Types in Clustering Problems

- **Unsupervised grouping** or **clustering**: extracting hidden structure from data.

Data Types in Clustering Problems

- **Unsupervised grouping** or **clustering**: extracting hidden structure from data.

- **Data Representations:**

  Vector data: n vectors in $\mathbb{R}^d$.

Data Types in Clustering Problems

- **Unsupervised grouping** or **clustering**: extracting hidden structure from data.

- **Data Representations:**

  Vector data: n vectors in $\mathbb{R}^d$.

  Histogram data: n histograms in $\mathbb{R}^d$.

Data Types in Clustering Problems

- **Unsupervised grouping** or **clustering**: extracting hidden structure from data.

- **Data Representations:**

Vector data:	n vectors in $\mathbb{R}^d$.
Histogram data:	n histograms in $\mathbb{R}^d$.
Proximity data:	$n \times n$ pairwise proximity matrix.
Much harder problem	structure hidden in n^2 pairwise relations.

Part I: Clustering Principles

- **Compactness**
  - K-Means
  - Histogram Clustering
  - Pairwise Data Clustering (Average Association)
  - Constant Shift Embedding
  - Parametric Distributional Clustering

Part I: Clustering Principles

- **Compactness**
  - K-Means
  - Histogram Clustering
  - Pairwise Data Clustering (Average Association)
  - Constant Shift Embedding
  - Parametric Distributional Clustering
- **Separation**
  - Average Cut
  - Normalized Cut

Part I: Clustering Principles

- **Compactness**
  - K-Means
  - Histogram Clustering
  - Pairwise Data Clustering (Average Association)
  - Constant Shift Embedding
  - Parametric Distributional Clustering
- **Separation**
  - Average Cut
  - Normalized Cut
- **Connectedness**
  - Mean Shift Clustering
  - Single Linkage
  - Path-Based Clustering

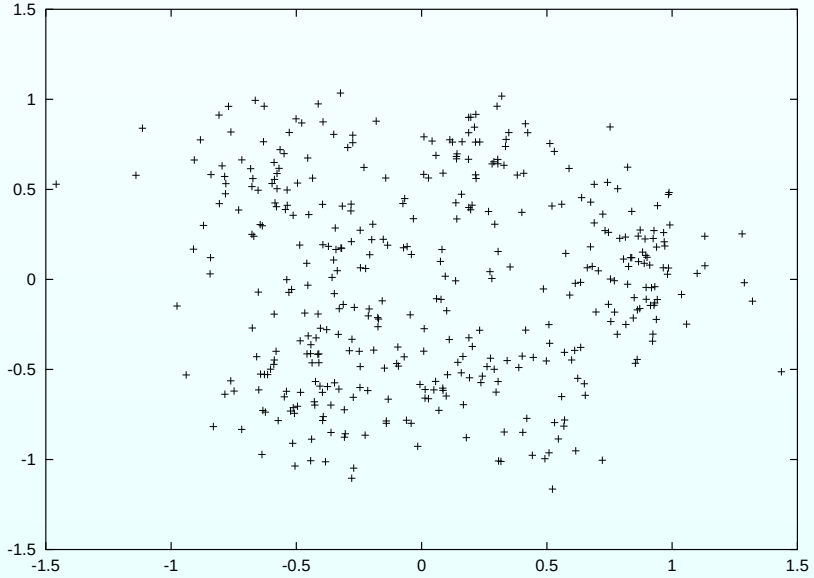
Vectorial Data:

Clustering: find **compact subsets** by way of k**-means**.

Vectorial Data:

Clustering: find **compact subsets** by way of k**-means**.

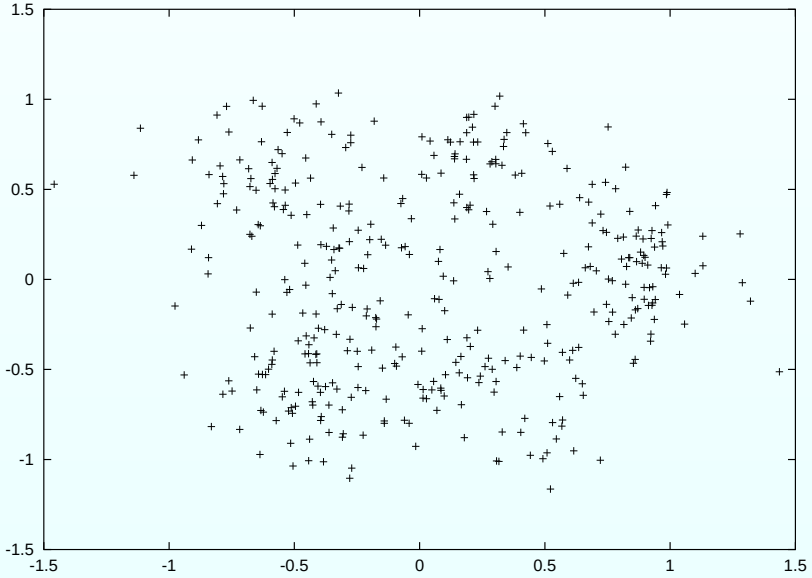
Raw data:



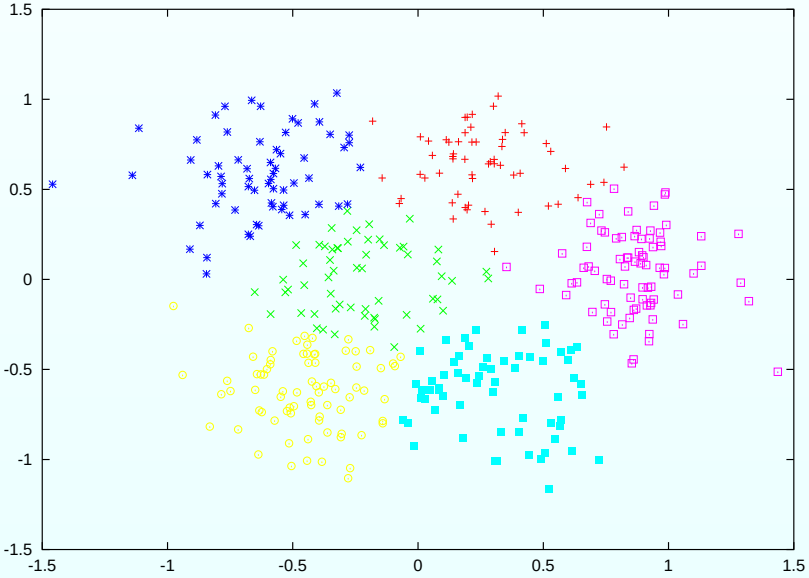
Vectorial Data:

Clustering: find **compact subsets** by way of k**-means**.

Raw data:



With cluster labels:



k-Means Problem

- Given d-dimensional sample vectors $x_1, \ldots, x_n \in \mathbb{R}^d$

k-Means Problem

- Given d-dimensional sample vectors $x_1, \ldots, x_n \in \mathbb{R}^d$
- Assignment vector $c \in \{1, \ldots, k\}^n$
- Prototypes $y_\nu \in \mathbb{R}^d$ ($\nu \in \{1, \ldots, k\}$)

k-Means Problem

- Given d-dimensional sample vectors $x_1, \ldots, x_n \in \mathbb{R}^d$
- Assignment vector $c \in \{1, \ldots, k\}^n$
- Prototypes $y_\nu \in \mathbb{R}^d$ ($\nu \in \{1, \ldots, k\}$)

Problem: Find c and y_ν that minimize

$$H^{\text{km}}(c, y) = \sum_{i=1}^{n} ||x_i - y_{c(i)}||^2$$

Mixed combinatorial and continuous optimization problem

k-Means Algorithm

1. **Choose** k sample objects randomly as prototypes

k-Means Algorithm

1. **Choose** k sample objects randomly as prototypes

2. **Iterate**:

   - Keep prototypes $y_{c(i)}$ fixed and assign sample vectors x_i to nearest prototype

$$c(i) = \arg \min_{\nu \in \{1,\ldots,k\}} ||x_i - y_{c(i)}||^2$$

k-Means Algorithm

1. **Choose** k sample objects randomly as prototypes

2. **Iterate**:

   - Keep prototypes $y_{c(i)}$ fixed and assign sample vectors x_i to nearest prototype

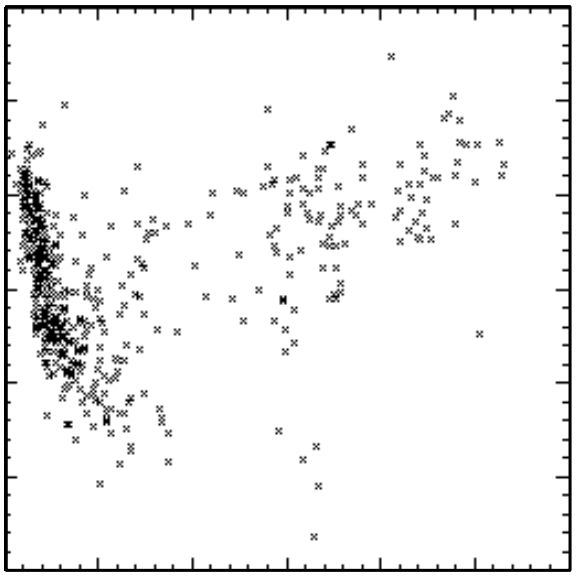
$$c(i) = \arg \min_{\nu \in \{1,\ldots,k\}} ||x_i - y_{c(i)}||^2$$

   - Keep assignments $c(i)$ fixed and estimate prototypes

$$y_\nu = \frac{1}{|\mathcal{C}_\nu|} \sum_{i:c(i)=\nu} x_i$$

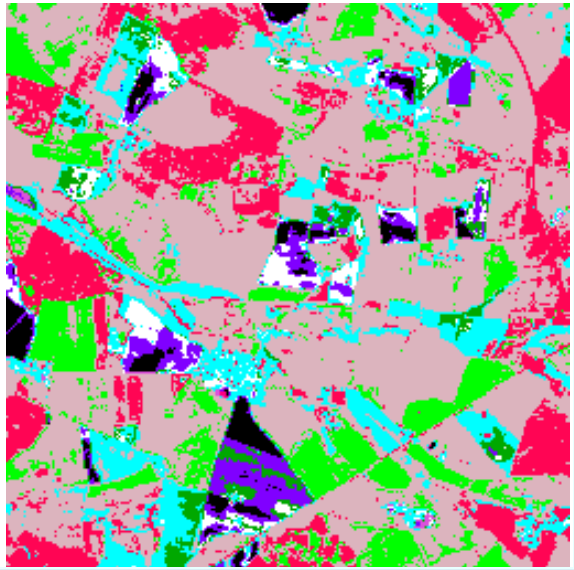
k-Means Segmentation of LANDSAT Images

Vectorial Data : $\mathbf{x}_i \in \mathbb{R}_+^6$

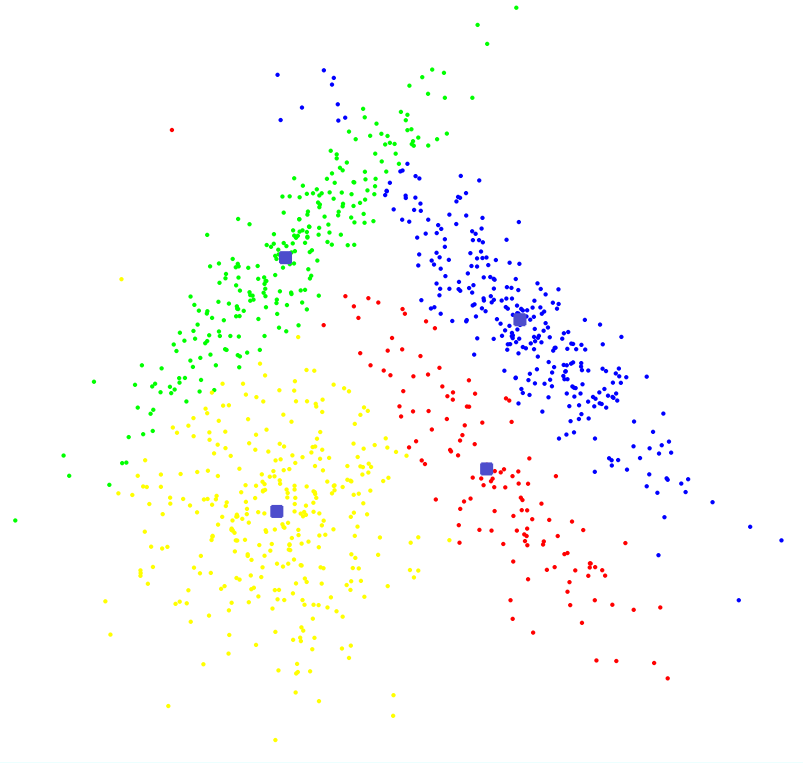


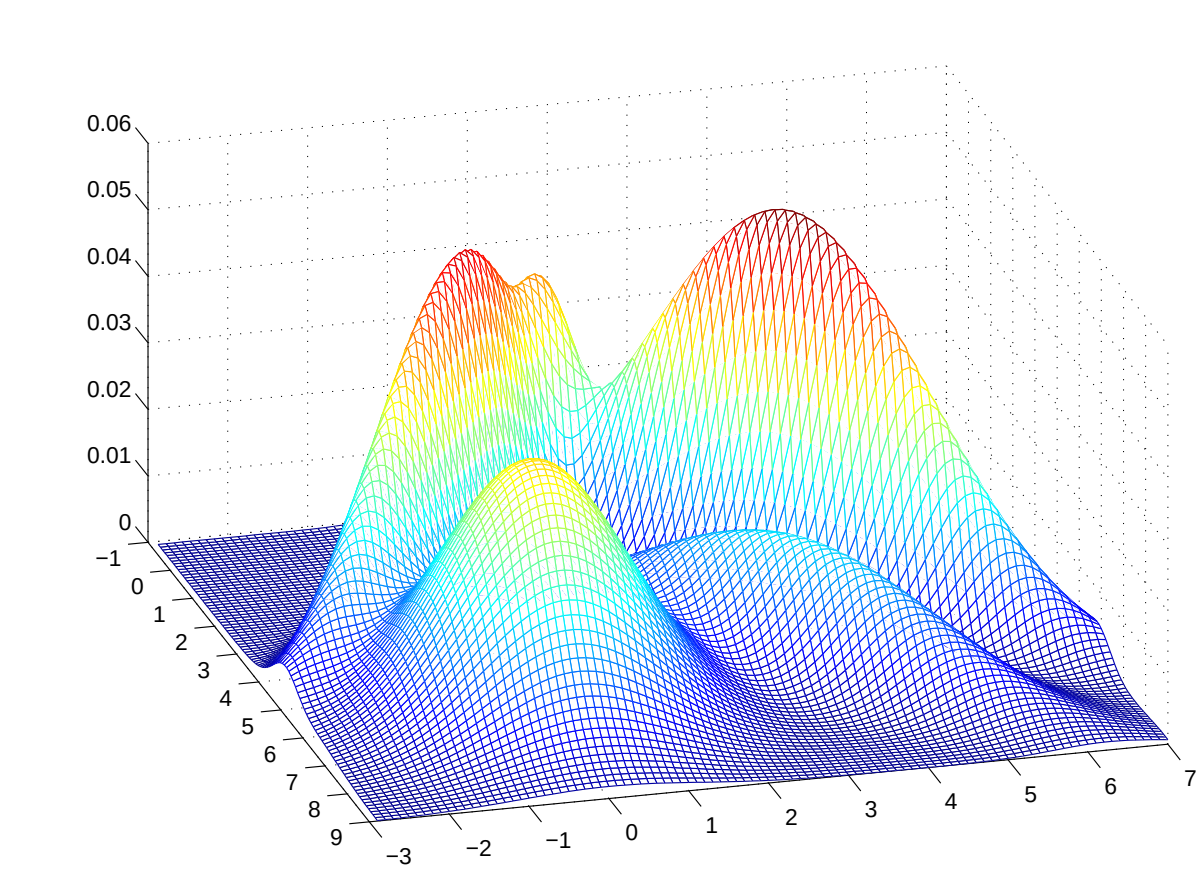
$k = 13$

$c : \mathbb{R}_+^6 \rightarrow \{1, \ldots, k\}$



Example Mixture Model

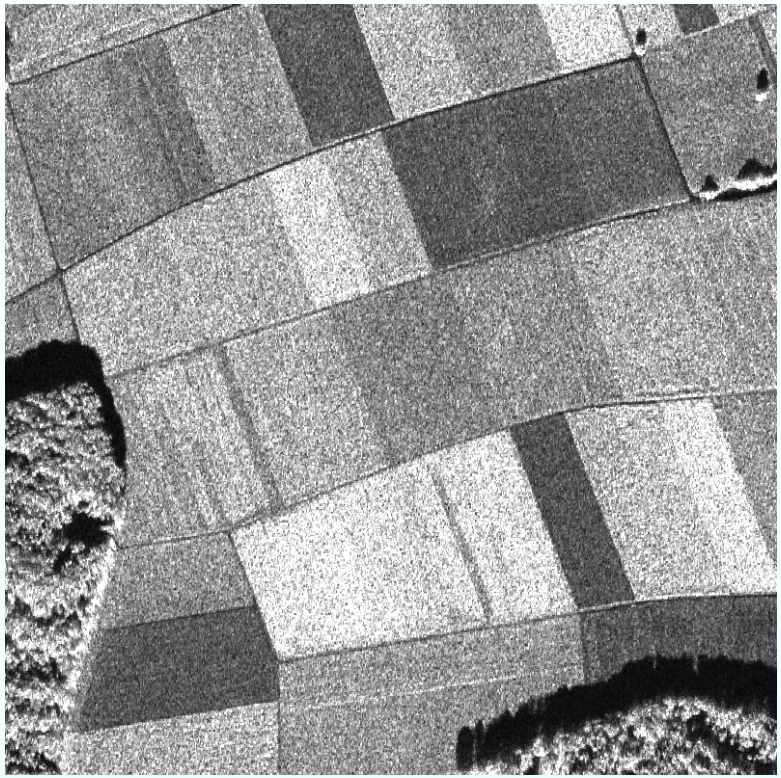




Parametric Distributional Clustering

related to

histogram clustering

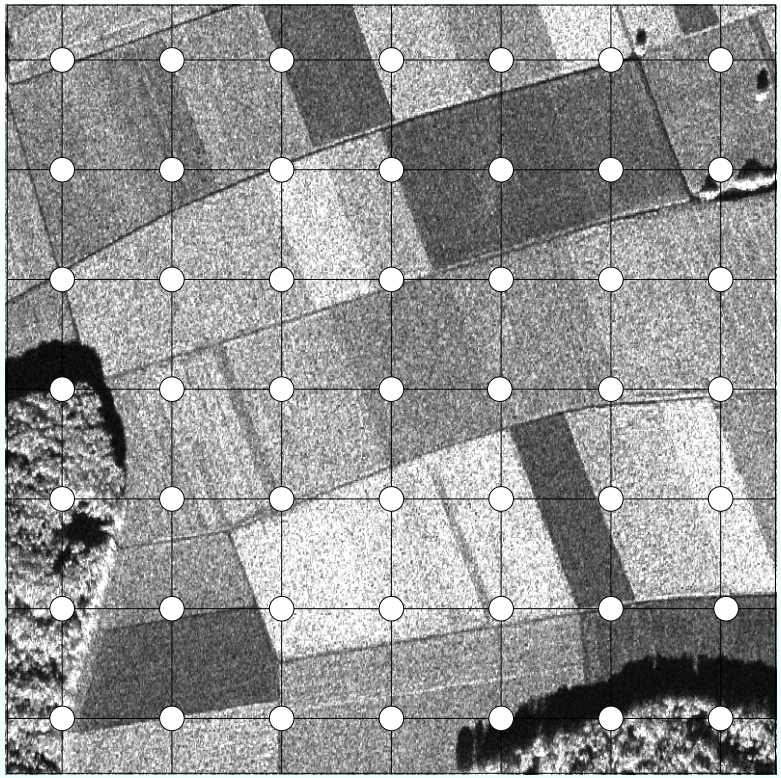


- Consider sites on a grid.

Parametric Distributional Clustering

related to

histogram clustering

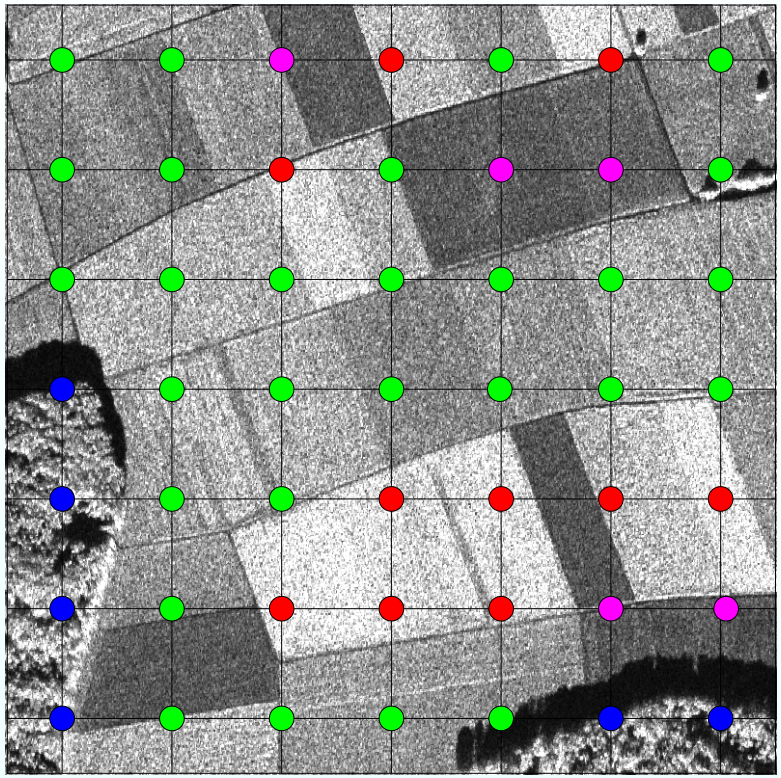


- Consider sites on a grid.

Parametric Distributional Clustering

related to

histogram clustering

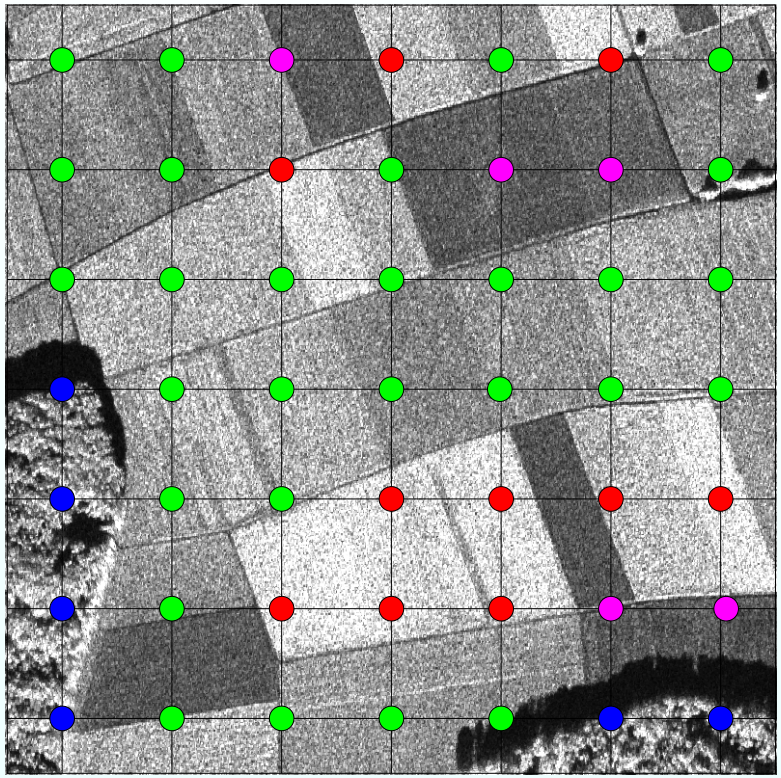


- Consider sites on a grid. Sites belong to clusters.

Parametric Distributional Clustering

related to

histogram clustering

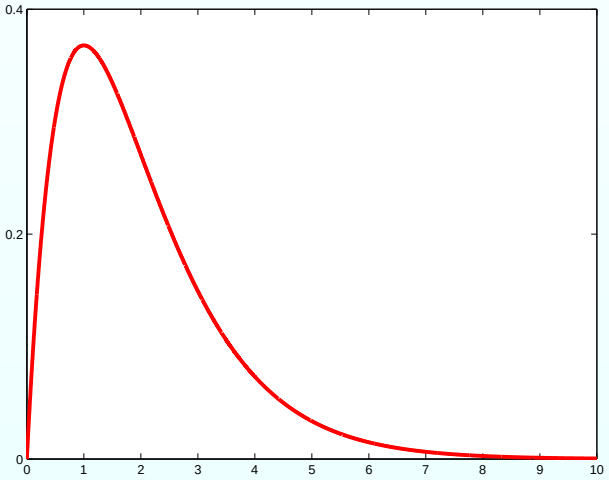


- Consider sites on a grid. Sites belong to clusters.
- Cluster memberships encoded by $\mathbf{c} \in \{1, \ldots, k\}^n$

Assumed Sampling Process

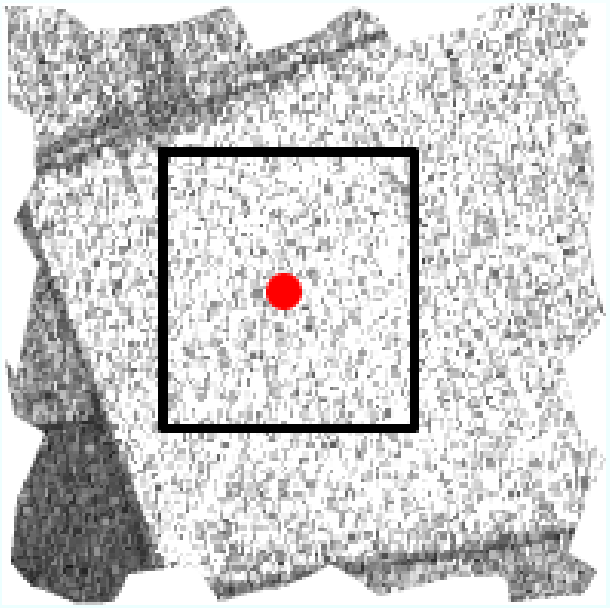
Mixture density

$p(x|\nu)$



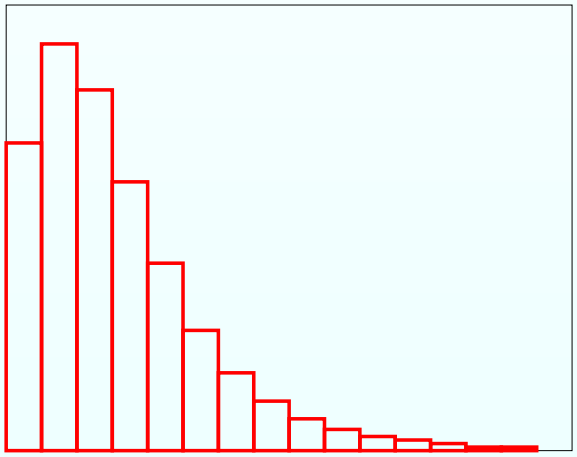
pixel values

$\{x \in \mathcal{N}_i\}$

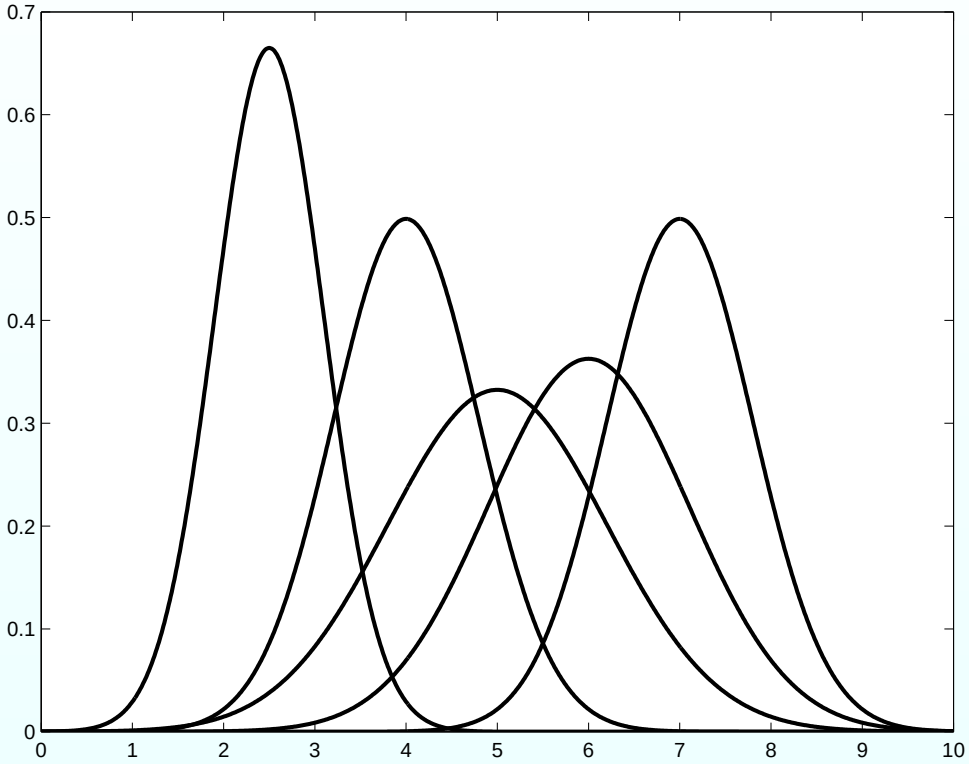


local histogram

$n_{ij}, 1 \leq j \leq m$

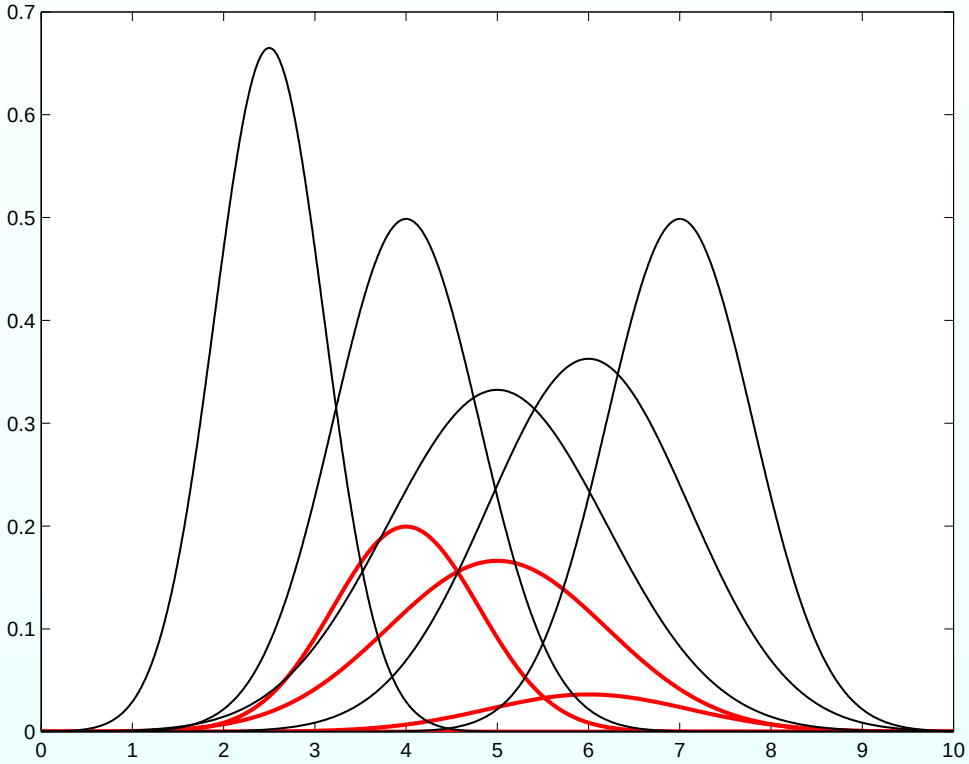


Gaussian Mixture Models



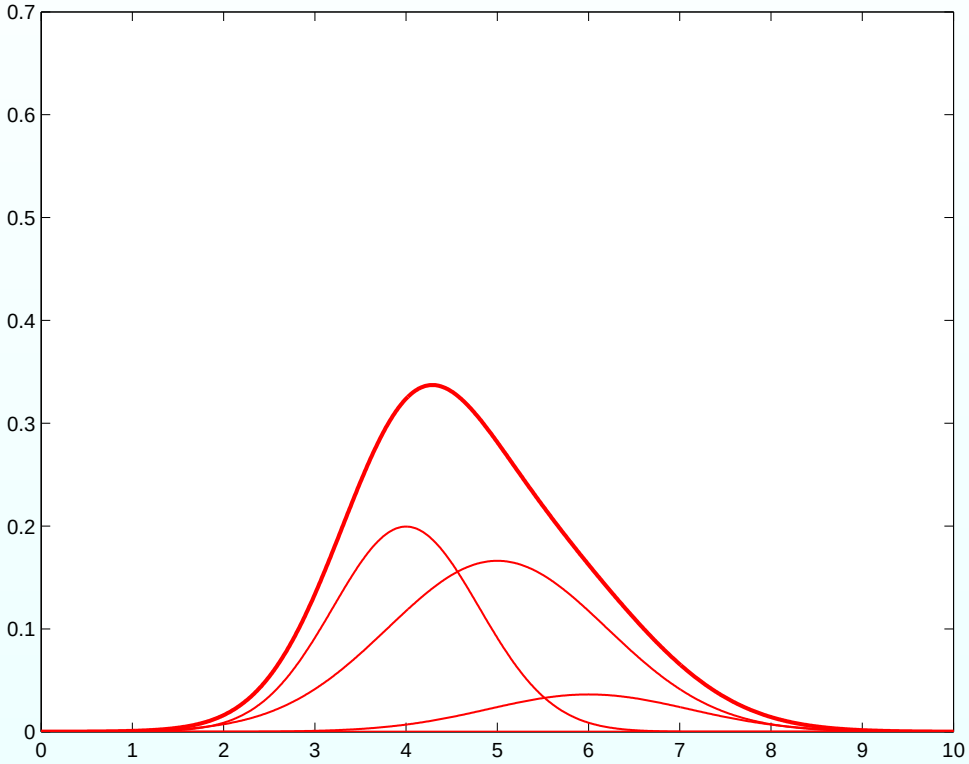
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p\left(x|\ \nu\right) = \sum_\alpha p_{\alpha|\ \nu} g_\alpha\left(x\right)$.

Gaussian Mixture Models



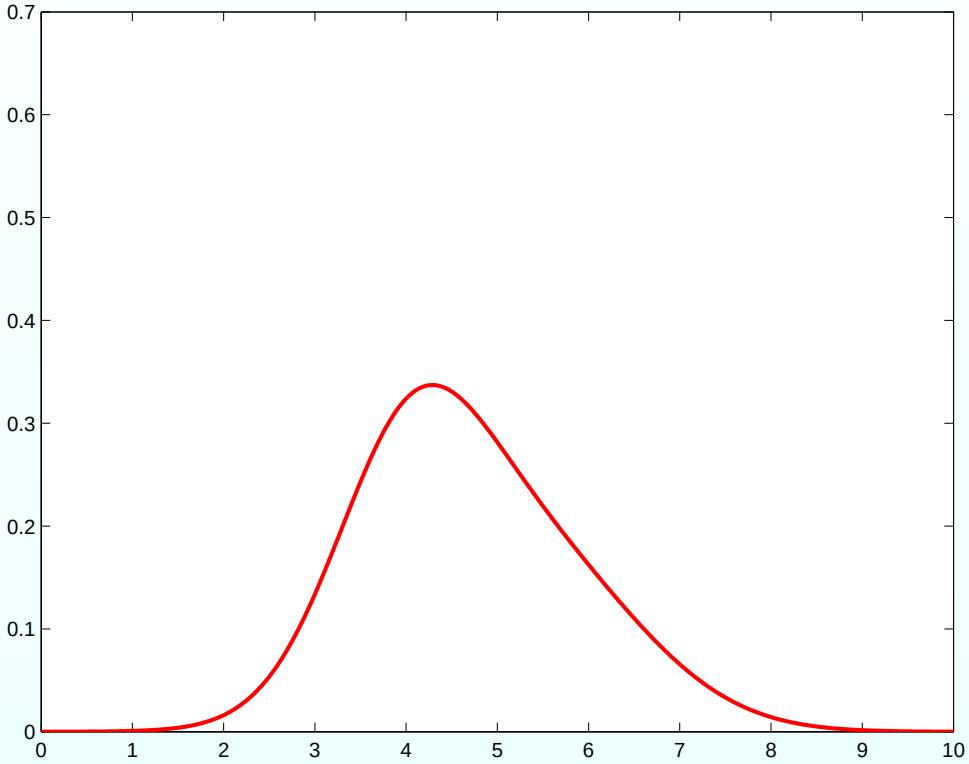
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p\left(x|\ \nu\right) = \sum_\alpha p_{\alpha|\ \nu} g_\alpha\left(x\right)$.

Gaussian Mixture Models



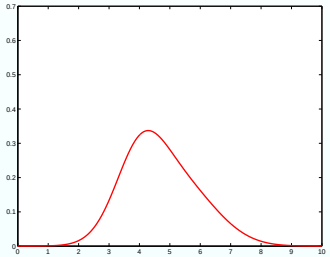
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p\left(x|\ \nu\right) = \sum_\alpha p_{\alpha|\ \nu} g_\alpha\left(x\right)$.

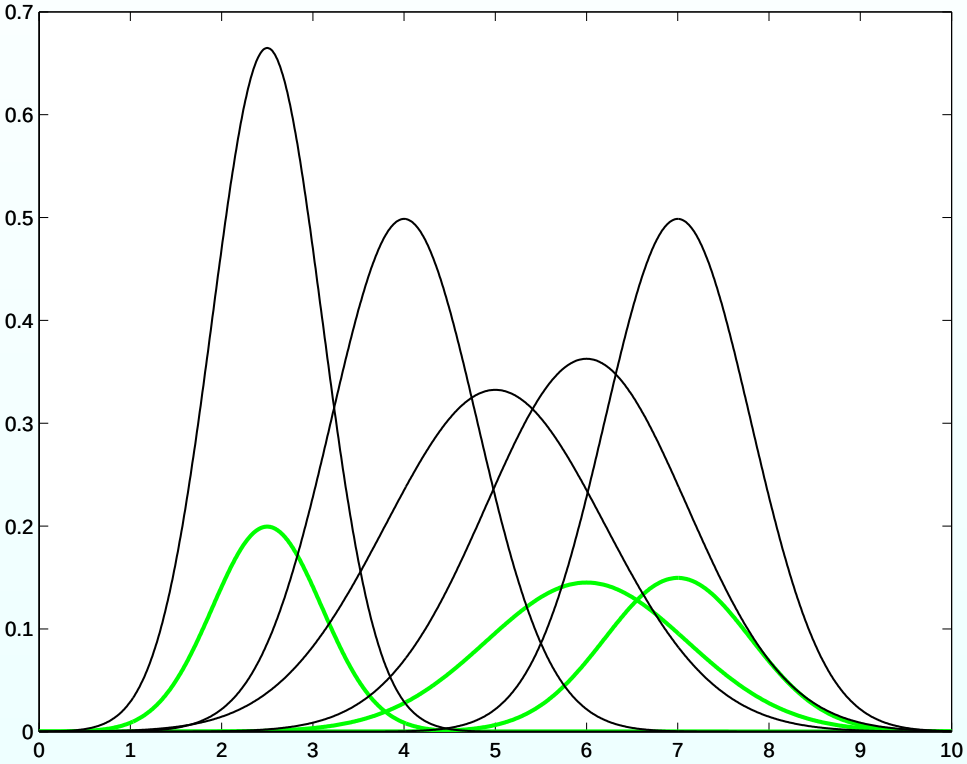
Gaussian Mixture Models



A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p\left(x|\ \nu\right) = \sum_\alpha p_{\alpha|\ \nu} g_\alpha\left(x\right)$.

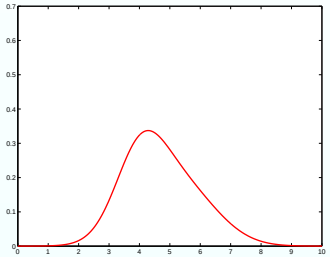
Gaussian Mixture Models

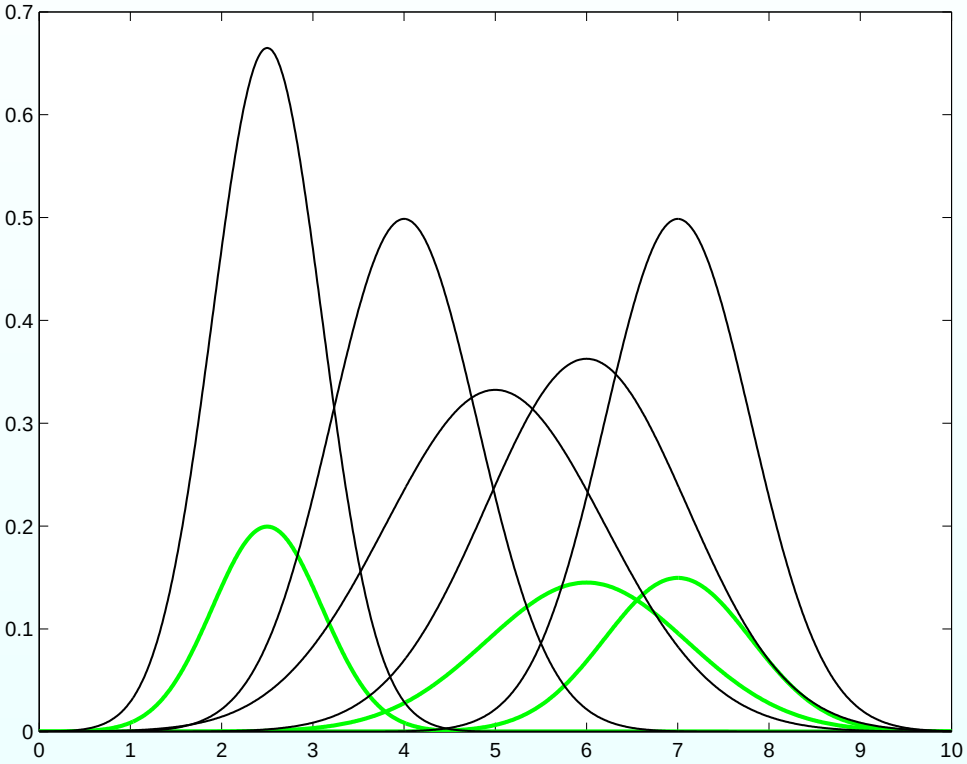




A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

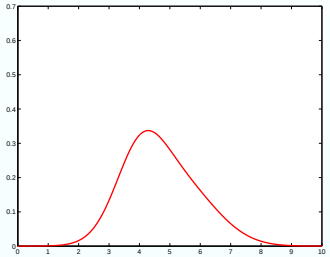
Gaussian Mixture Models

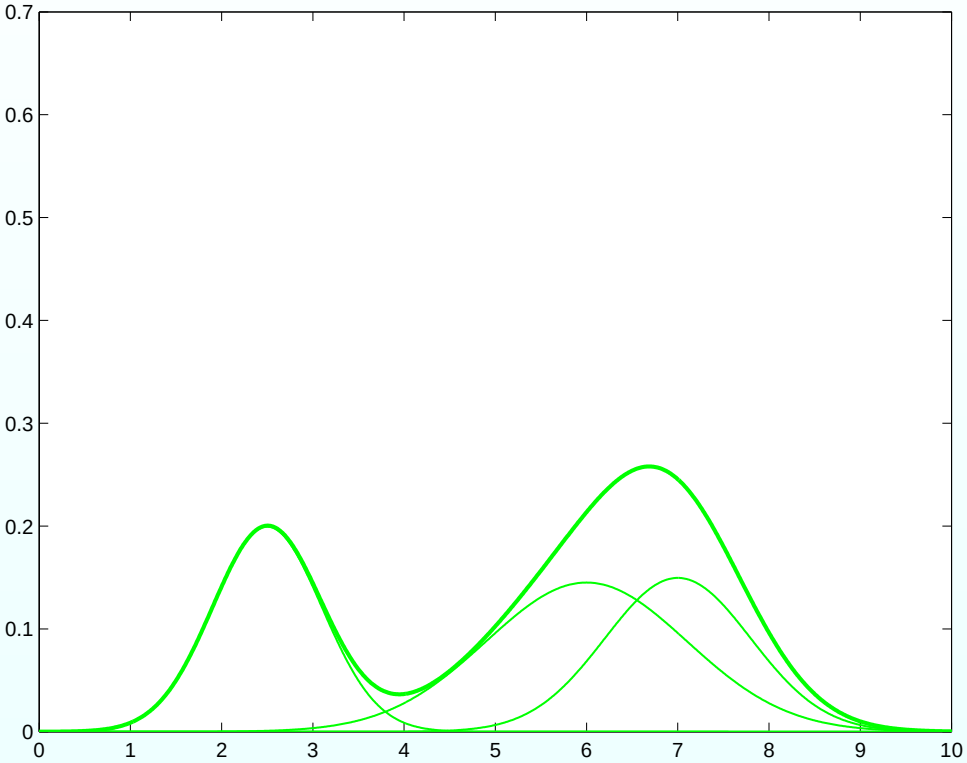




A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p\left(x|\ \nu\right) = \sum_\alpha p_{\alpha|\ \nu} g_\alpha\left(x\right)$.

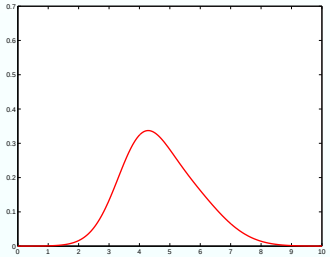
Gaussian Mixture Models

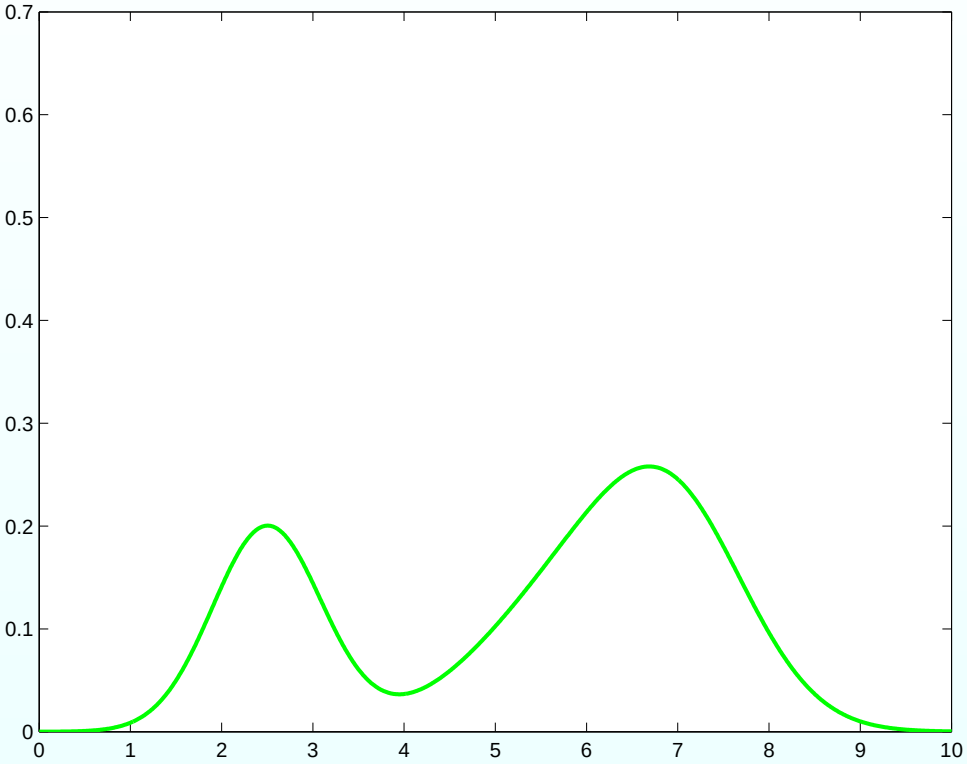




A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

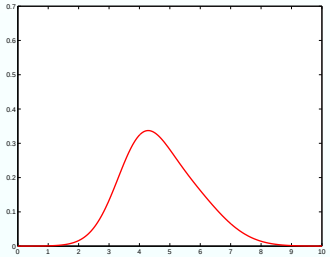
Gaussian Mixture Models

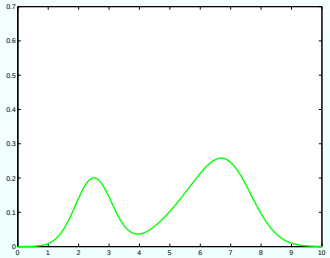


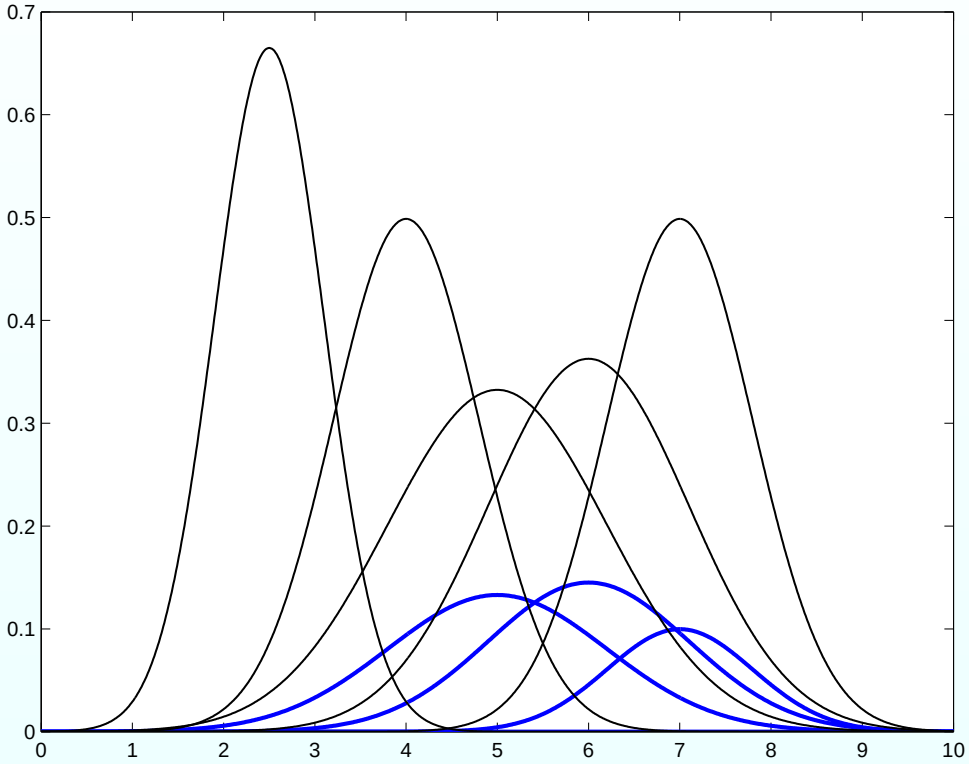


A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

Gaussian Mixture Models

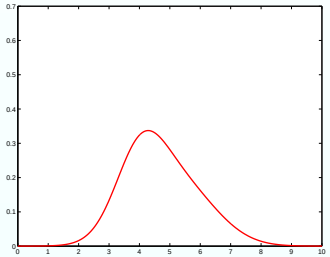


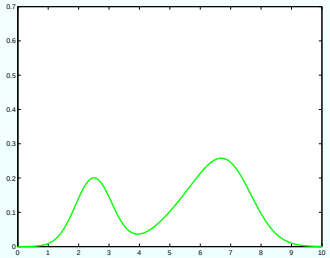


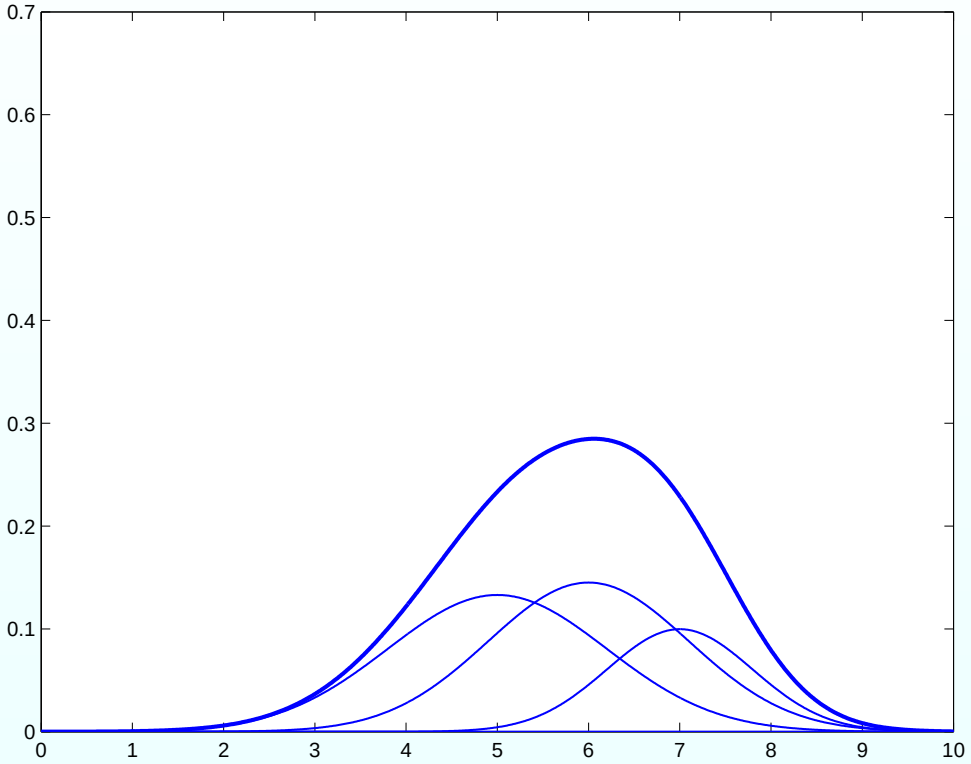


A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

Gaussian Mixture Models

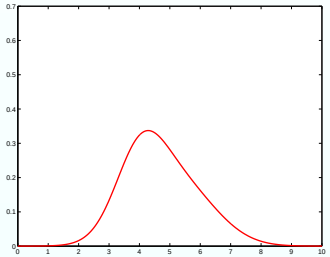


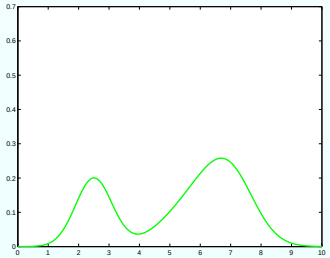


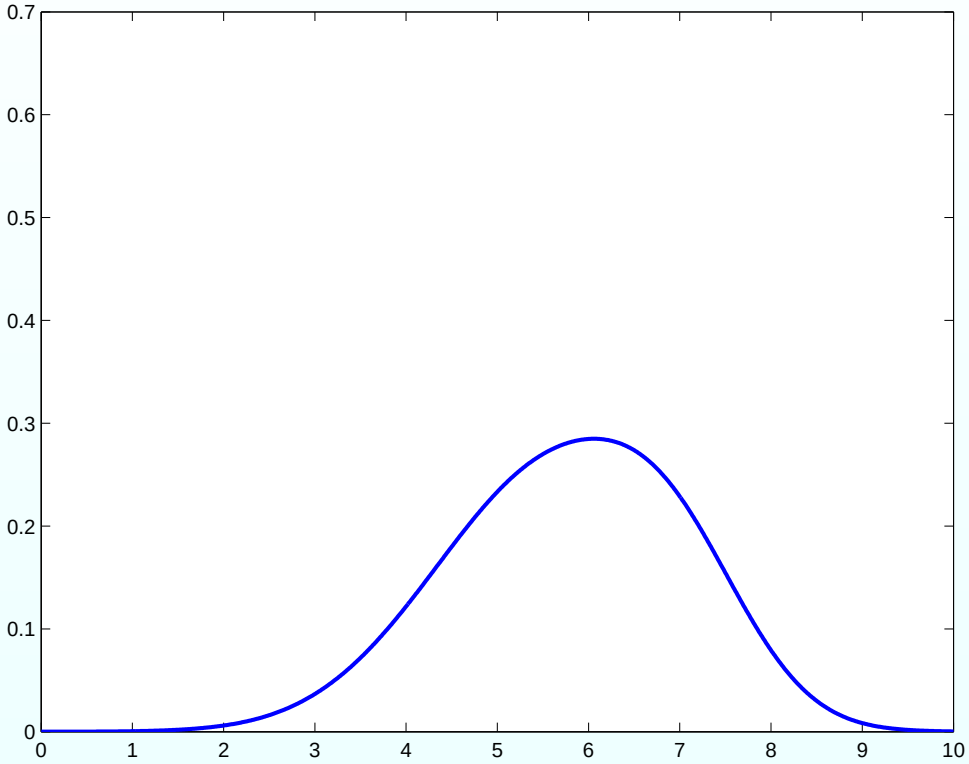


A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

Gaussian Mixture Models

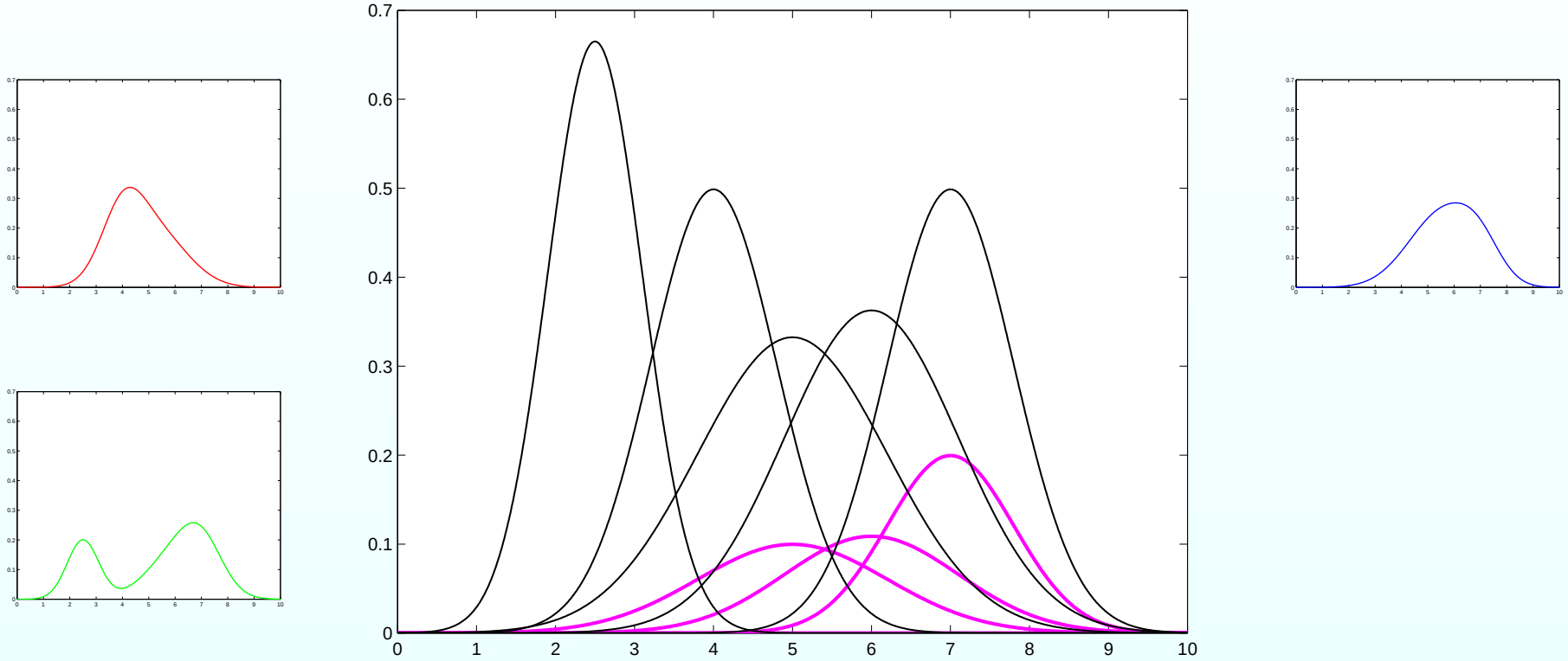






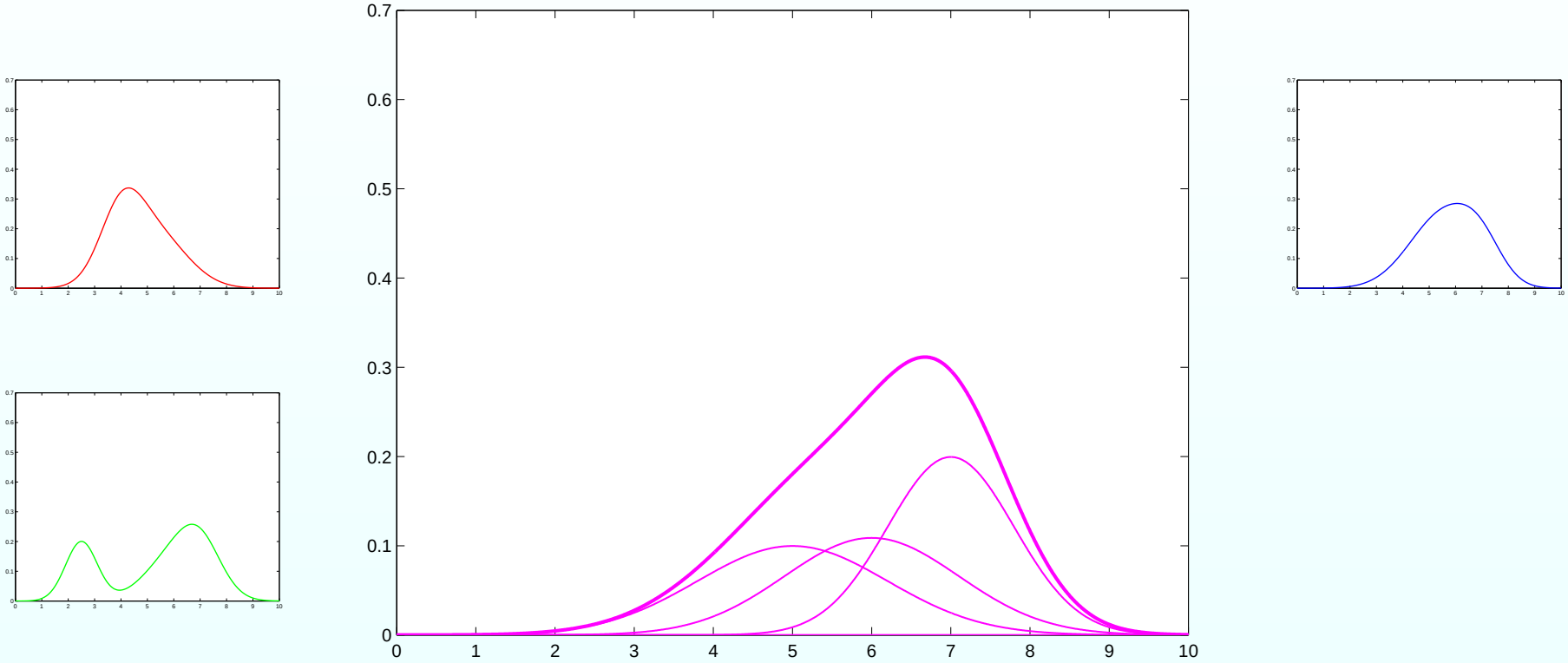
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\,\nu) = \sum_\alpha p_{\alpha|\,\nu} g_\alpha(x)$.

Gaussian Mixture Models



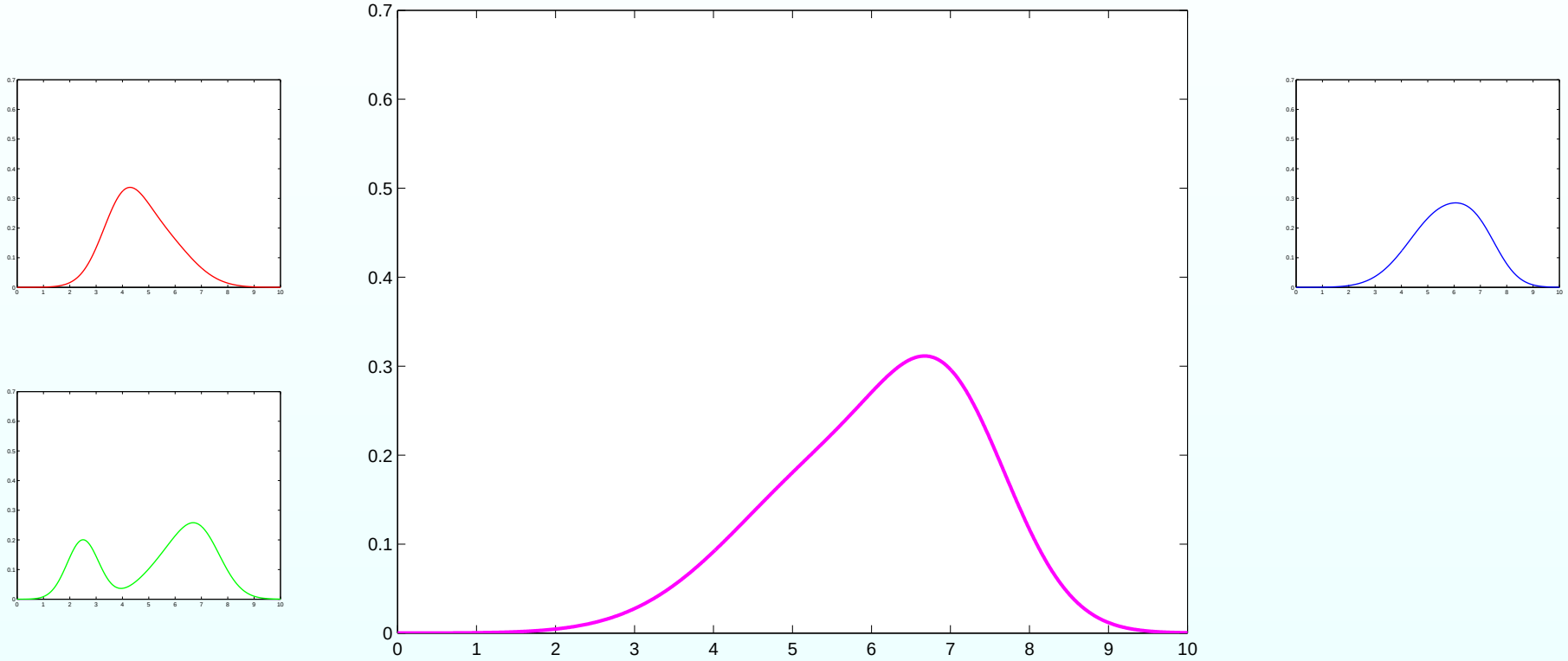
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\nu) = \sum_\alpha p_{\alpha|\nu} g_\alpha(x)$.

Gaussian Mixture Models



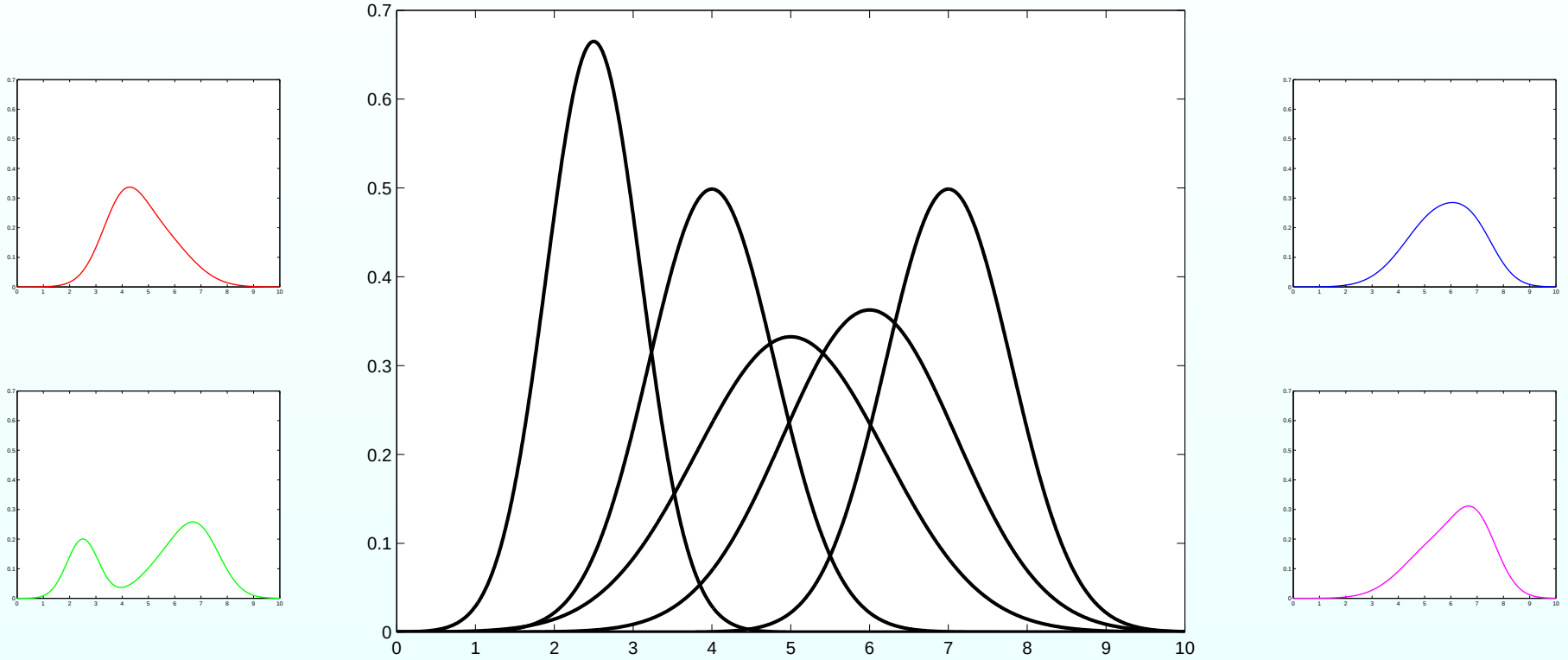
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\nu) = \sum_\alpha p_{\alpha|\nu} g_\alpha(x)$.

Gaussian Mixture Models



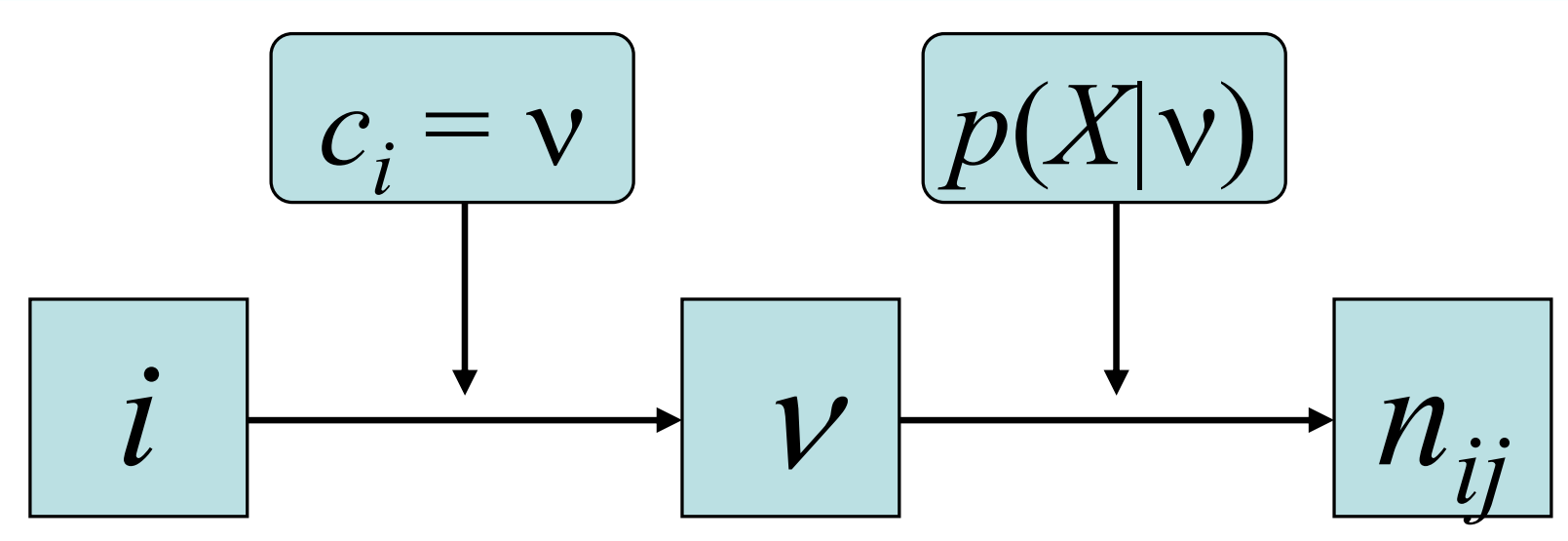
A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\nu) = \sum_\alpha p_{\alpha|\nu} g_\alpha(x)$.

Gaussian Mixture Models



A single selection of Gaussian prototypes $g_\alpha(x)$ is used to create mixture densities $p(x|\nu) = \sum_\alpha p_{\alpha|\nu} g_\alpha(x)$.

Generative Model



Color values only depend on cluster membership!

Maximum Likelihood Approach

Prior of assignment function $\mathbf{c}$:

$$p\left(\mathbf{c}|\ \theta\right) = \prod_{i=1}^{n} p_{\mathbf{c}(i)}$$

Data likelihood for given $\mathbf{c}$:

$$p\left(\mathcal{X}|\ \mathbf{c}, \theta\right) \ = \ \prod_{i=1}^{n} \left[\prod_{j=1}^{m} \left(\sum_{\alpha=1}^{\ell} p_{\alpha|\ \mathbf{c}(i)} \tilde{G}_{\alpha}(j) \right)^{n_{ij}} \right]$$

Cost Function for PDC

Cost function = negative log-likelihood:

$$H = -\sum_i \left[\log p_{\mathbf{c}(i)} + \sum_j n_{ij} \log \left(\sum_\alpha p_{\alpha|\,\mathbf{c}(i)} \tilde{G}_\alpha(j) \right) \right]$$

Interpretation as **two-part coding scheme**:
Expected codelength when encoding the cluster memberships and, based on that information, encoding the individual color values.

Hermes, Zöller, Buhmann, 2002

Information Bottleneck

Essential idea:

- Find efficient code $X \mapsto \tilde{X}$ ($\tilde{X}$ is a codebook vector)
- Preserve relevant information about context variable Y

Information Bottleneck

Essential idea:

- Find efficient code $X \mapsto \tilde{X}$ ($\tilde{X}$ is a codebook vector)
- Preserve relevant information about context variable Y

Tradeoff is made explicit by cost functional

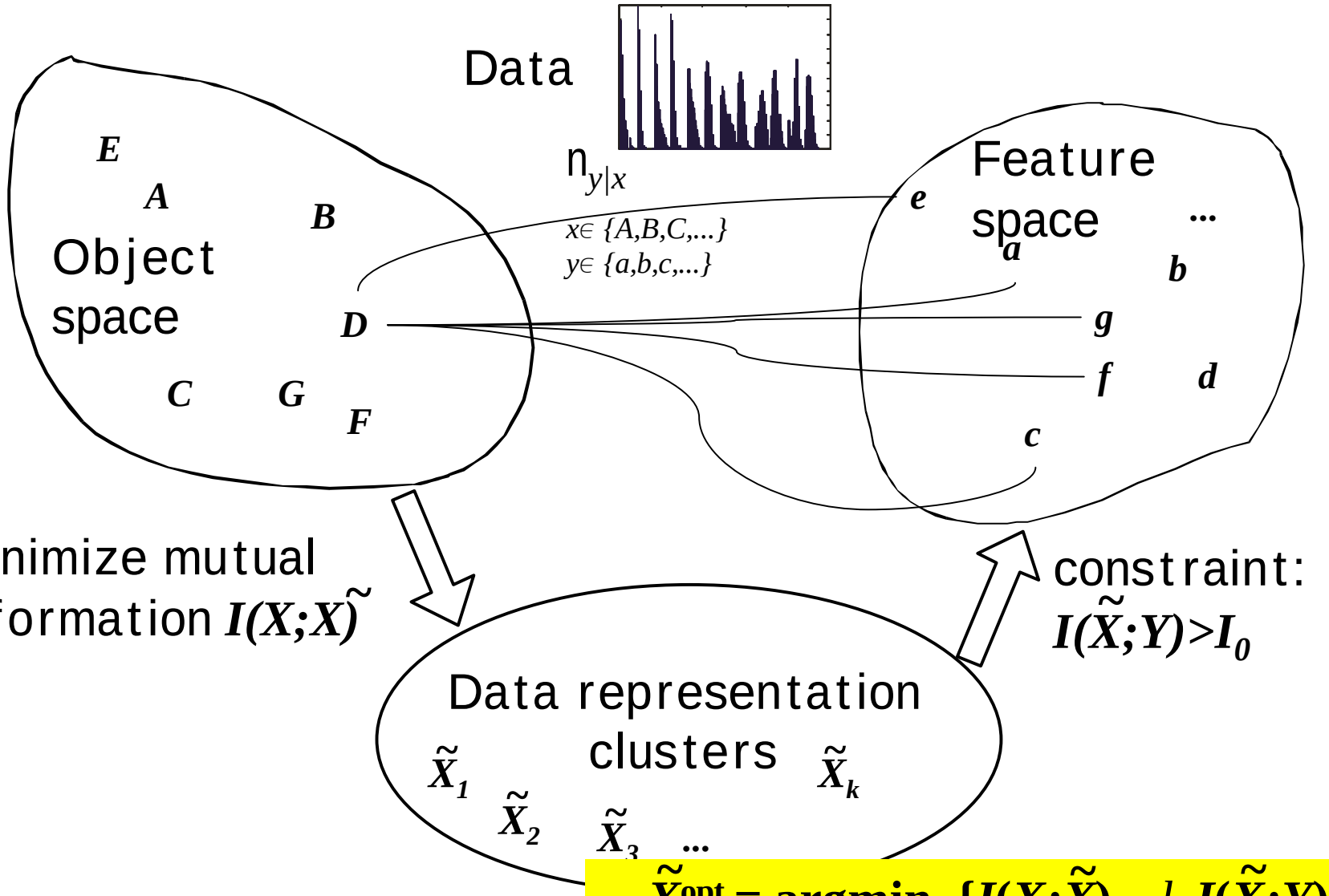
$$H^{IB} = I\left(X;\, \tilde{X}\right) - \lambda I\left(\tilde{X};\, Y\right) \;,$$

where $I(A;\, B)$ is the mutual information between two random variables A and B, $\lambda > 0$.

Tishby et al., 1999

Information Bottleneck

(**Tishby, Pereira, Bialek** IT Allerton Conf. 1999)

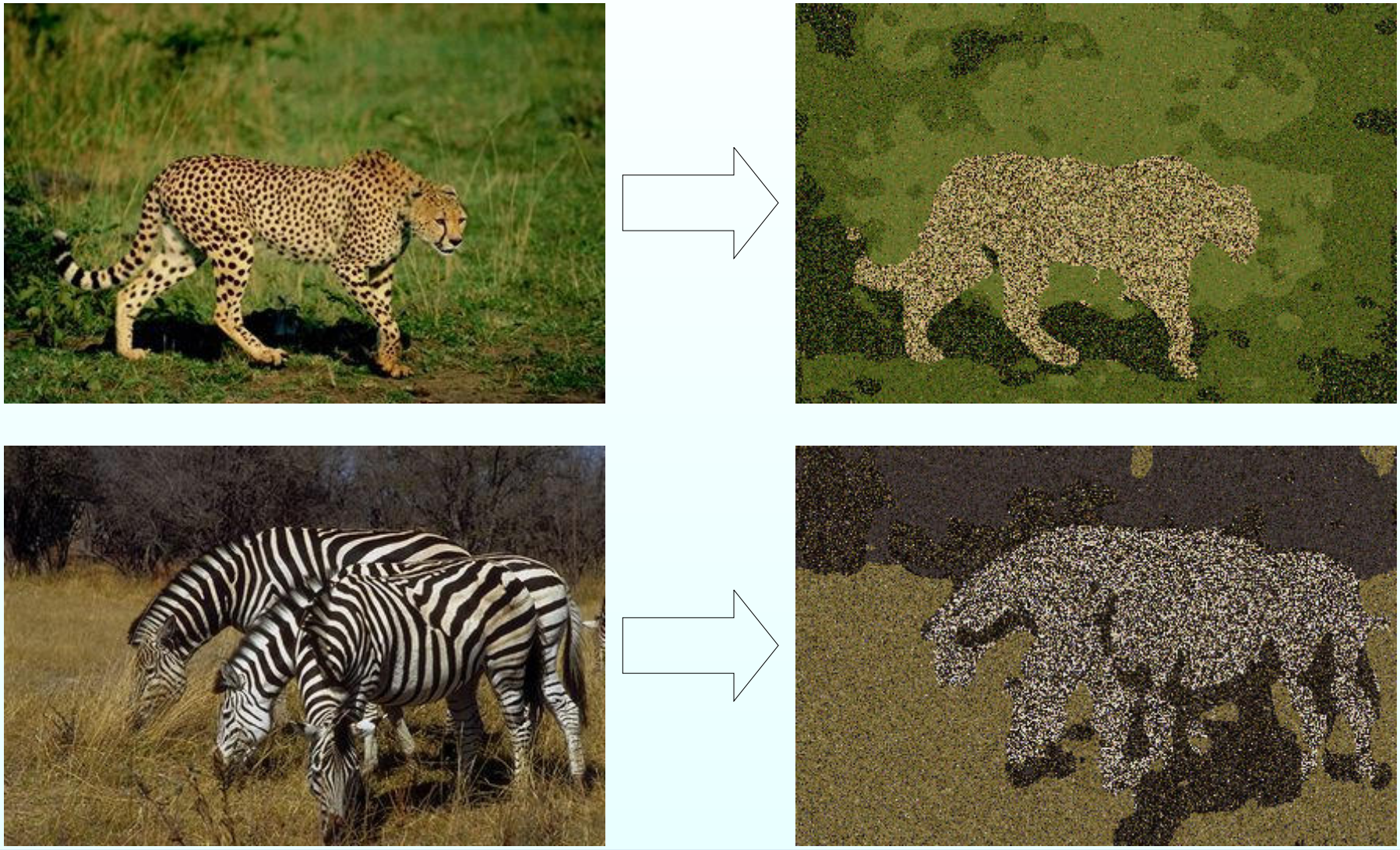


$$\tilde{X}^{opt} = \text{argmin}_{\tilde{X}}\{I(X;\tilde{X}) - \lambda\, I(\tilde{X};Y)\}$$

PDC Segmentation

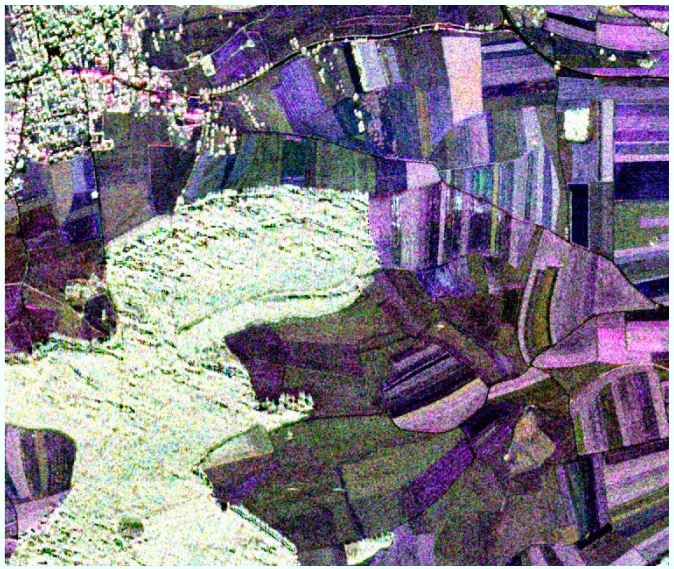
PDC Segmentation

PDC Resampling

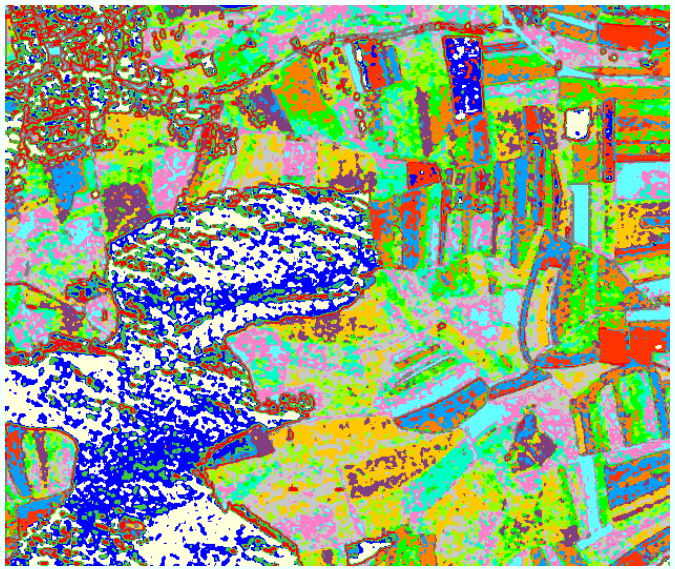


SAR Imagery

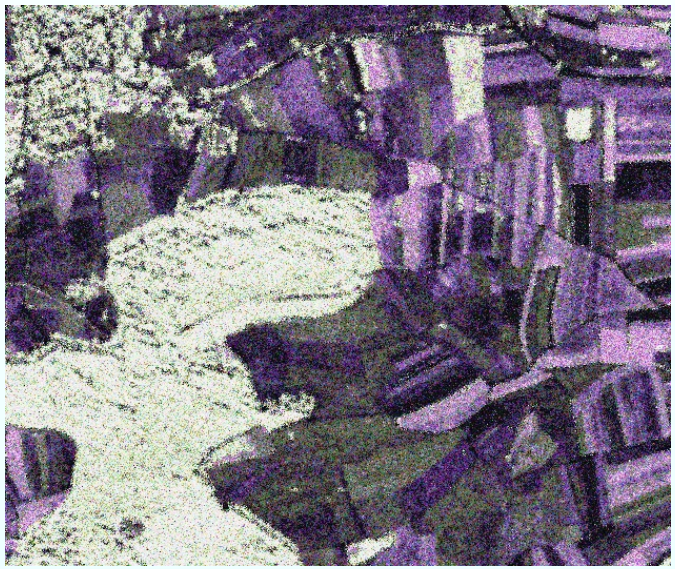
Polarimetric synthetic aperture radar image, L-band



original



segmentation

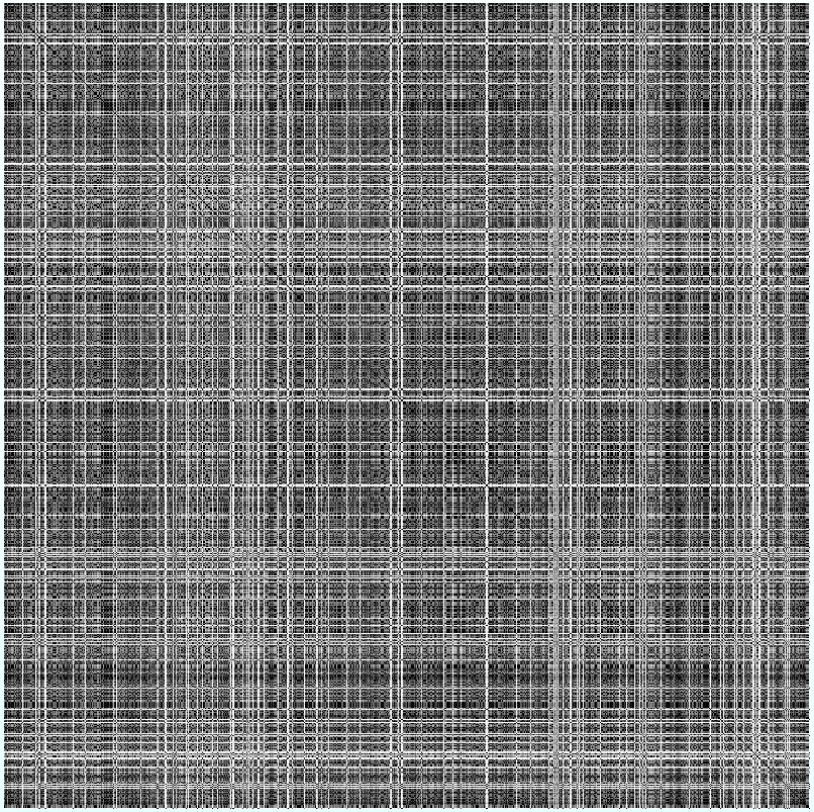


resampled

Proximity Data

Clustering: find **compact subsets** in dissimilarity data

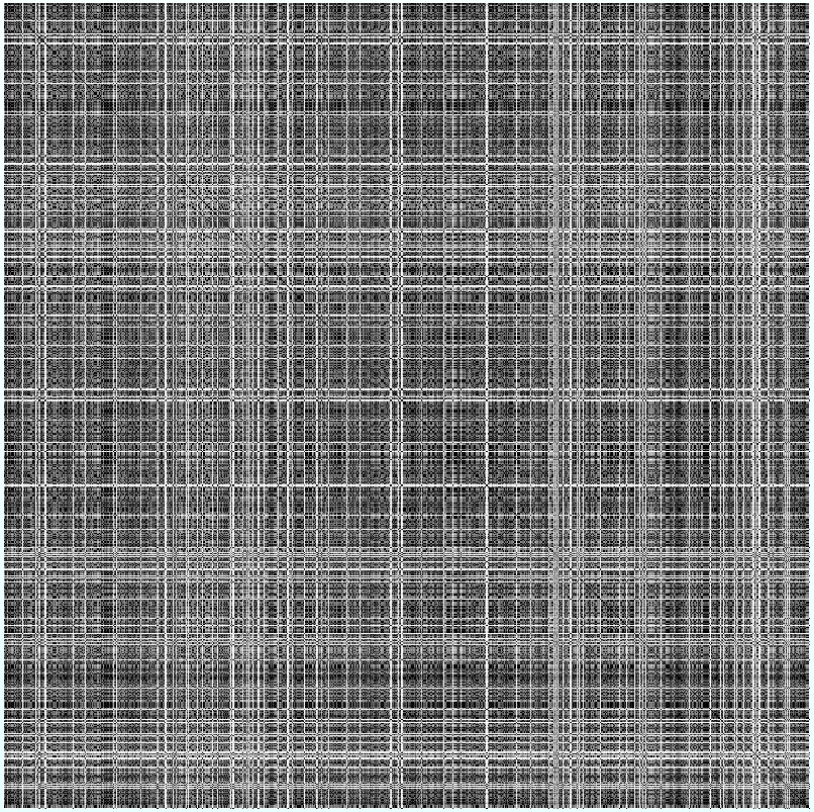
Raw prox. data:



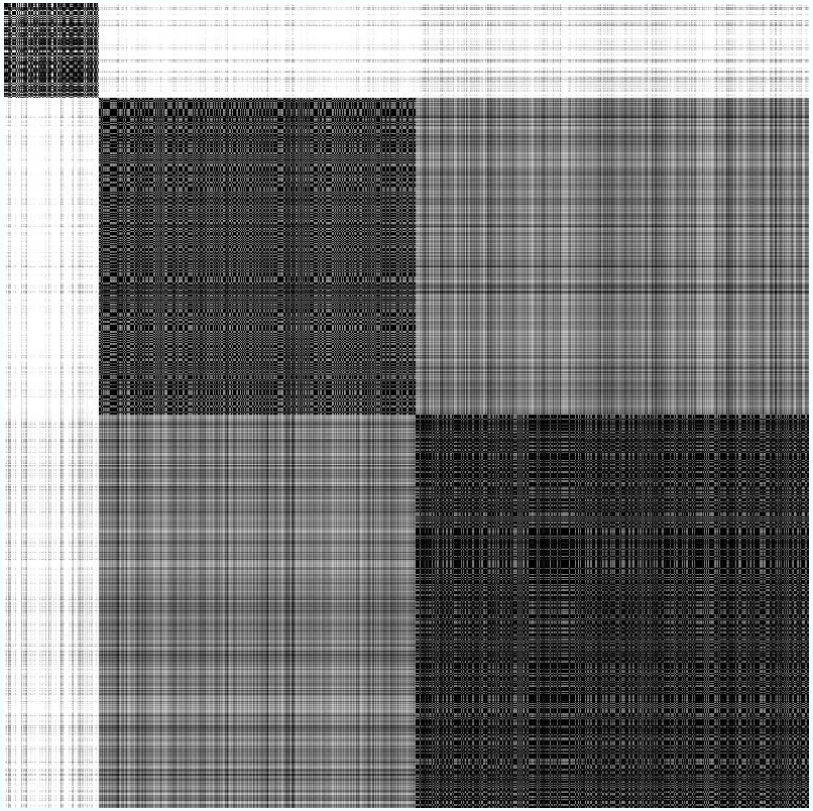
Proximity Data

Clustering: find **compact subsets** in dissimilarity data

Raw prox. data:



Permuted according to cluster labels:



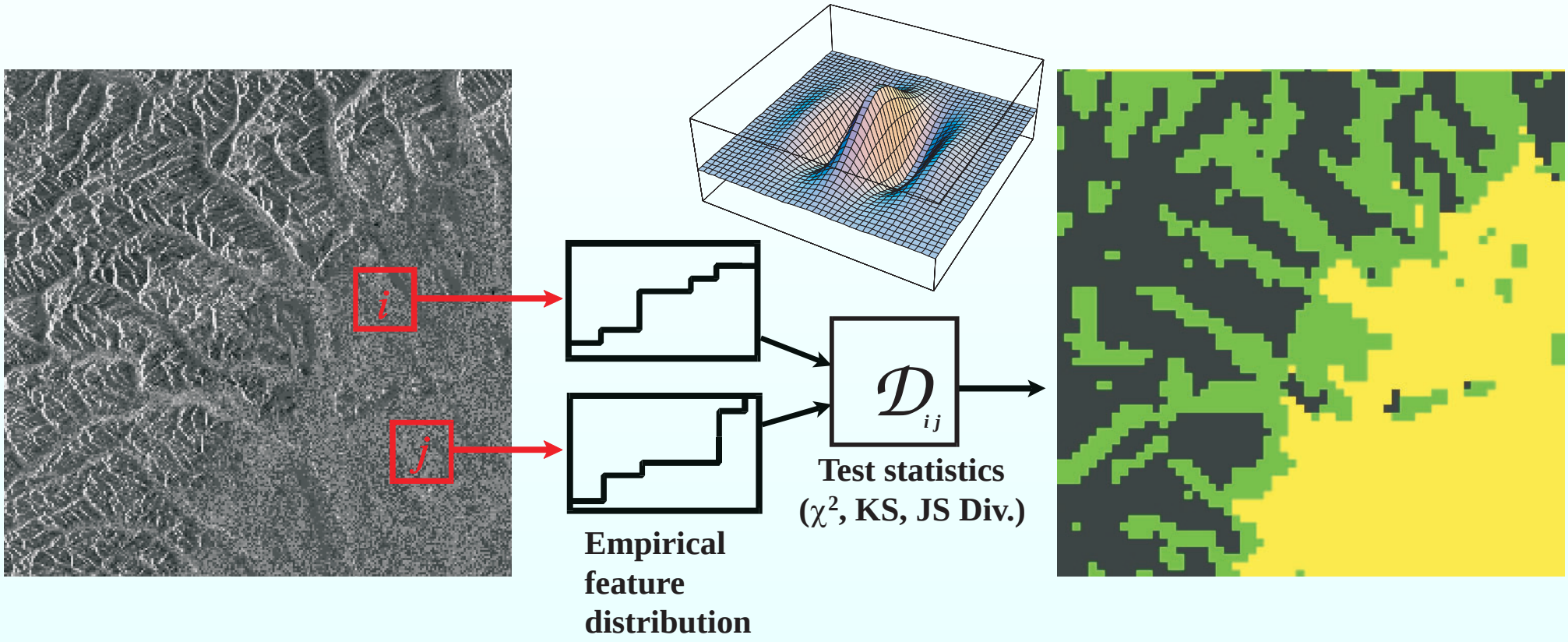
Proximity Data: Example

- Abundant in many applications, e.g. linguistics, psychology, **molecular biology**.
- **Example:** pairwise alignment scores between sequences.

```
       130       140       150       160       170
SDEGSDLAHAVDAEQAFAEGAQAADAVEATPVEPVKVRERKRTFHYPDGLKDYVTAI
 ::       .: :  ..:  ...::    : .: : ::.:. ::::.::.:::  .
LDE-------IDNETELVE--ETTDA----PKKPKK-REKKKIFHYPNGLEDYVHYL
120                 130           140         150       160
```

Proximity Data in Segmentation

SAR imagery analysed by Gabor filters



The Pairwise Clustering Cost Function

Idea: emphasize **compact** clusters by minimizing normalized sum of **intra-cluster dissimilarities**

$$H^{\text{pc}}(c; D) = \sum_{\nu=1}^{k} \frac{1}{|\mathcal{C}_\nu|} \sum_{(i,j):c_i=c_j=\nu} D_{ij},$$

with assignment vector $c \in \{1, \ldots, k\}^n$ and $\mathcal{C}_\nu = \{i : c_i = \nu\}$.

The Pairwise Clustering Cost Function

Idea: emphasize **compact** clusters by minimizing normalized sum of **intra-cluster dissimilarities**

$$H^{\text{pc}}(c; D) = \sum_{\nu=1}^{k} \frac{1}{|\mathcal{C}_\nu|} \sum_{(i,j):c_i=c_j=\nu} D_{ij},$$

with assignment vector $c \in \{1, \ldots, k\}^n$ and $\mathcal{C}_\nu = \{i : c_i = \nu\}$.

Euclidean distances: if $D_{ij} = \|x_i - x_j\|^2$ then

$$H^{\text{pc}}(c; \mathcal{X}) = H^{\text{km}} = \sum_{\nu=1}^{k} \sum_{i:c_i=\nu} \|x_i - y_\nu\|^2 \quad \text{for means } y_\nu.$$

Invariance Properties of H^{pc}

H^{pc} **invariant** under...

Invariance Properties of H^{pc}

H^{pc} **invariant** under...

symmetrizing transformations:

$$\tilde{D}_{ij} = (1/2)(D_{ij} + D_{ji}) \quad \Rightarrow \quad \tilde{H} = H.$$

$\Rightarrow$ **consider only symmetric matrices**.

Invariance Properties of H^{pc}

H^{pc} **invariant** under...

symmetrizing transformations:

$$\tilde{D}_{ij} = (1/2)(D_{ij} + D_{ji}) \quad \Rightarrow \quad \tilde{H} = H.$$

$\Rightarrow$ **consider only symmetric matrices**.

additive shifts of the off-diagonal elements D:

$$\tilde{D}_{ij} = D_{ij} + D_0(1 - \delta_{ij}) \quad \Rightarrow \quad \tilde{H} = H + \text{const.}$$

$\Rightarrow$ **shift does not influence assignments!**

Constant Shift Embedding

Define $\tilde{D} = D + \lambda_0(1 - I)$ with smallest eigenvalue λ_0 of centralized matrix $D^c_{ij} = D_{ij} - \frac{1}{n}\sum_{k=1}^{n} D_{ik} - \frac{1}{n}\sum_{k=1}^{n} D_{kj} - \frac{1}{n^2}\sum_{k,l=1}^{n} D_{kl}$, then

Roth et al., IEEE-TPAMI 2003

Constant Shift Embedding

Define $\tilde{D} = D + \lambda_0(1 - I)$ with smallest eigenvalue λ_0 of centralized matrix $D^c_{ij} = D_{ij} - \frac{1}{n}\sum_{k=1}^{n} D_{ik} - \frac{1}{n}\sum_{k=1}^{n} D_{kj} - \frac{1}{n^2}\sum_{k,l=1}^{n} D_{kl}$, then

1. $\tilde{D}_{ij}$ are **squared Euclidean distances** between vectors $\{x_i\}_{i=1}^{n} \in \mathbb{R}^{n-1}$,

Roth et al., IEEE-TPAMI 2003

Constant Shift Embedding

Define $\tilde{D} = D + \lambda_0(1 - I)$ with smallest eigenvalue λ_0 of centralized matrix $D^c_{ij} = D_{ij} - \frac{1}{n}\sum_{k=1}^{n} D_{ik} - \frac{1}{n}\sum_{k=1}^{n} D_{kj} - \frac{1}{n^2}\sum_{k,l=1}^{n} D_{kl}$, then

1. $\tilde{D}_{ij}$ are **squared Euclidean distances** between vectors $\{x_i\}_{i=1}^{n} \in \mathbb{R}^{n-1}$,
2. optimal assignments for k-means based on $\{x_i\}_{i=1}^{n}$ are **identical** to those of the pairwise problem,

Roth et al., IEEE-TPAMI 2003

Constant Shift Embedding

Define $\tilde{D} = D + \lambda_0(1 - I)$ with smallest eigenvalue λ_0 of centralized matrix $D^c_{ij} = D_{ij} - \frac{1}{n}\sum_{k=1}^{n} D_{ik} - \frac{1}{n}\sum_{k=1}^{n} D_{kj} - \frac{1}{n^2}\sum_{k,l=1}^{n} D_{kl}$, then

1. $\tilde{D}_{ij}$ are **squared Euclidean distances** between vectors $\{x_i\}_{i=1}^{n} \in \mathbb{R}^{n-1}$,
2. optimal assignments for k-means based on $\{x_i\}_{i=1}^{n}$ are **identical** to those of the pairwise problem,
3. $\{x_i\}_{i=1}^{n}$ are explicitly found by eigenvalue decomposition.

Roth et al., IEEE-TPAMI 2003

Constant Shift Embedding

Define $\tilde{D} = D + \lambda_0(1 - I)$ with smallest eigenvalue λ_0 of centralized matrix $D^c_{ij} = D_{ij} - \frac{1}{n}\sum_{k=1}^{n} D_{ik} - \frac{1}{n}\sum_{k=1}^{n} D_{kj} - \frac{1}{n^2}\sum_{k,l=1}^{n} D_{kl}$, then

1. $\tilde{D}_{ij}$ are **squared Euclidean distances** between vectors $\{x_i\}_{i=1}^{n} \in \mathbb{R}^{n-1}$,
2. optimal assignments for k-means based on $\{x_i\}_{i=1}^{n}$ are **identical** to those of the pairwise problem,
3. $\{x_i\}_{i=1}^{n}$ are explicitly found by eigenvalue decomposition.
4. **optimal approximative vectors** (in least-squares sense): projecting on leading eigenvectors (**kernel PCA**).

Roth et al., IEEE-TPAMI 2003

Clustering of Bacterial *GyrB* Sequences

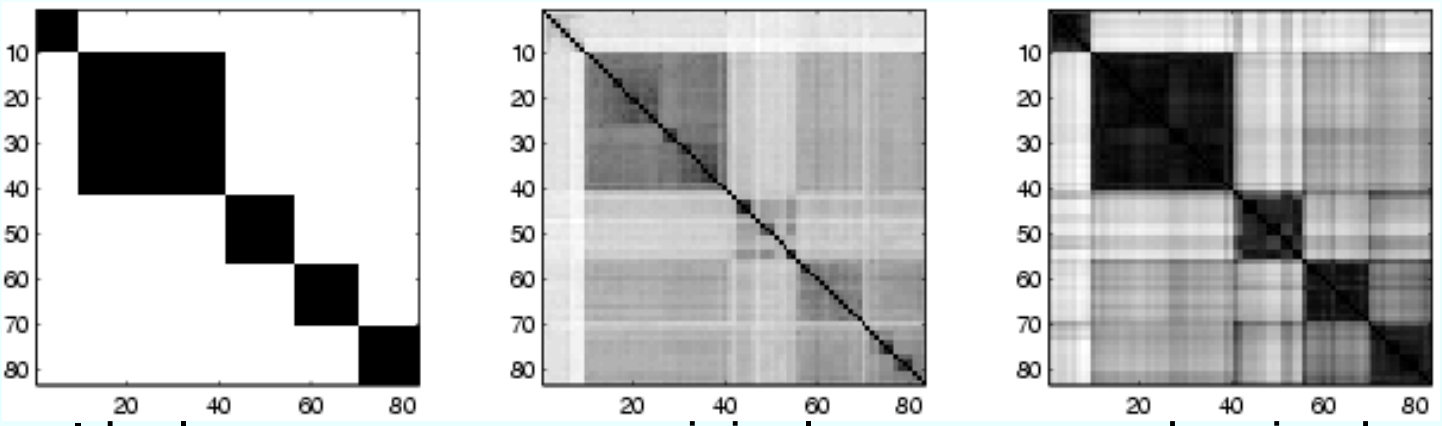
- **Objects:** 84 amino acid sequences from 5 genera.

Clustering of Bacterial *GyrB* Sequences

- **Objects:** 84 amino acid sequences from 5 genera.
- **Representations:** pairwise alignment scores

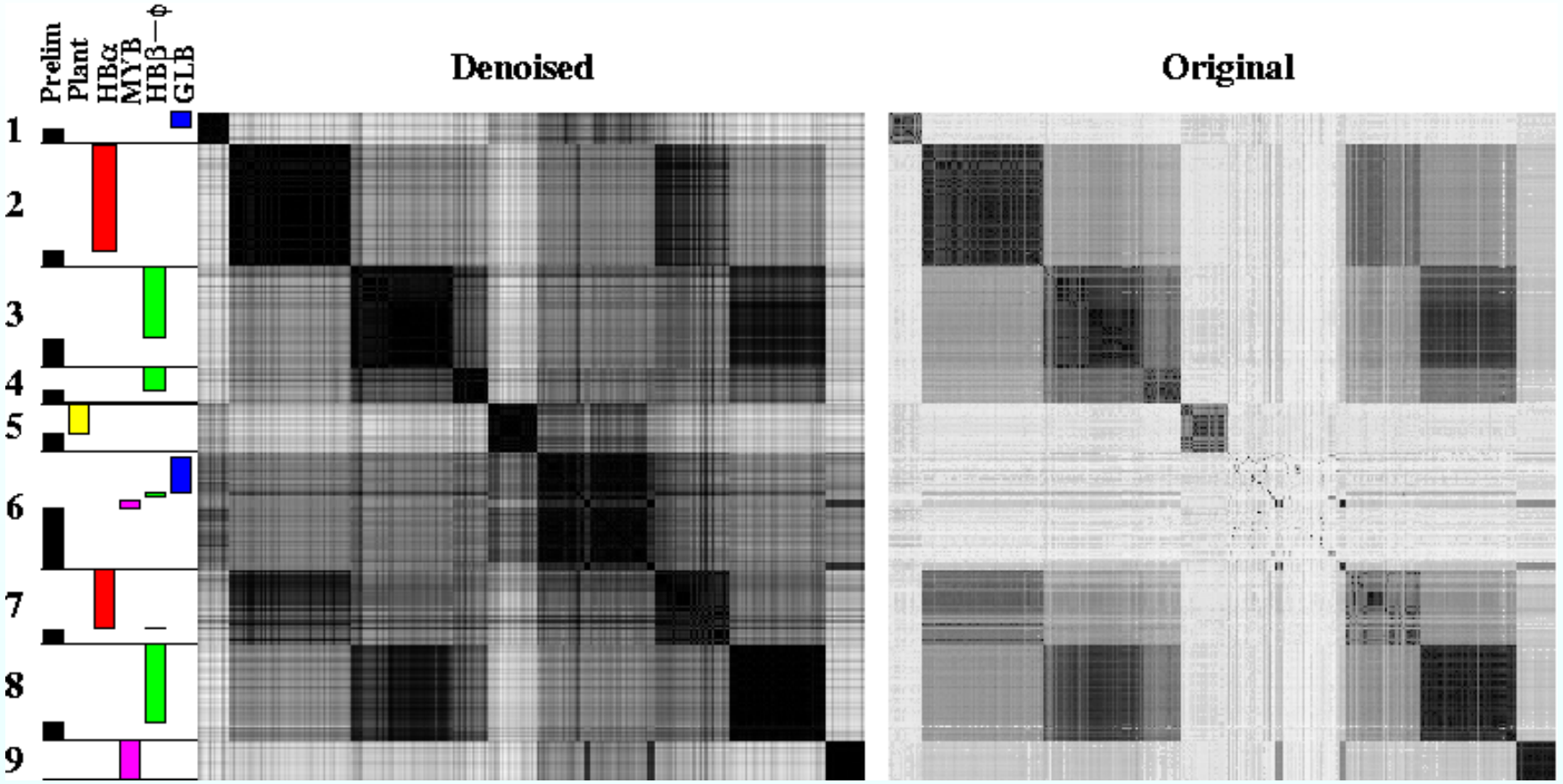
Clustering of Bacterial *GyrB* Sequences

- **Objects:** 84 amino acid sequences from 5 genera.
- **Representations:** pairwise alignment scores
- k**-means clustering:** denoised (5 dimensions): 3 misclassifications w.r.t. known ground truth, original (83 dimensions): 17 misclassifications.



Ideal original denoised

Globin Proteins: Cluster Solution



Interpretation: biologically relevant clusters!

Normalized Cut

Minimize the cut, while maximize the association

$$H^{\text{NCut}}(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(B,A)}{assoc(B,V)}$$

Shi & Malik, 2000

Normalized Cut

Minimize the cut, while maximize the association

$$H^{\text{NCut}}(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(B,A)}{assoc(B,V)}$$

With partition vector $c \in \{-1,1\}^n$ and association matrix $W = (w)_{ij}$:

$$H^{\text{NCut}}(c,W) = \frac{\sum_{c_i>0,c_j<0} -w_{ij}c_ic_j}{\sum_{c_i>0}\sum_{j=1}^{n} w_{ij}} + \frac{\sum_{c_i<0,c_j>0} -w_{ij}c_ic_j}{\sum_{c_i<0}\sum_{j=1}^{n} w_{ij}}$$

Shi & Malik, 2000

Relaxation of NCut

Minimize Rayleigh quotient

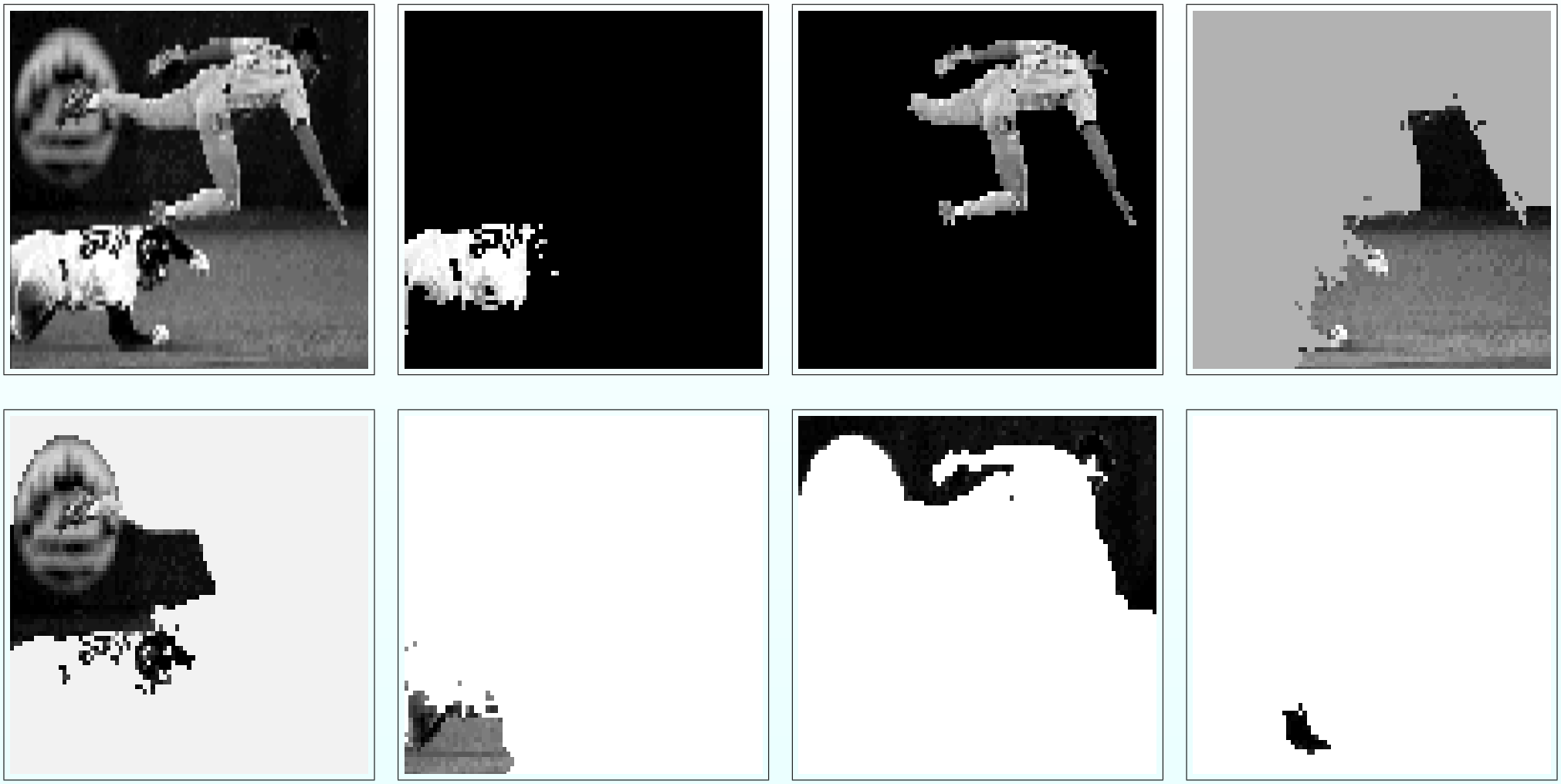
$$\min_c H^{\text{NCut}}(c, W) = \min_y \frac{y^t(D-W)y}{y^t y}$$

$$\text{subject to} \quad y \in \left\{1, \frac{-\sum_{x_i>0} d_i}{\sum_{x_i<0} d_i}\right\}$$

where $D = diag(d_1, \ldots, d_n)$ and $d_i = \sum_{j=1}^{n} w_{ij}$

$x \in [-1, 1]^n$ is the relaxation of the variable vector $c \in \{-1, 1\}^n$.

Example Normalized Cut



Shi & Malik, 2000

Part II: Optimization Methods

Given: cost function to rank different object partitions

Robustness: cost function depends on noisy data

$\Rightarrow$ partition with minimal costs is a r.v. of noisy data

Part II: Optimization Methods

Given: cost function to rank different object partitions

Robustness: cost function depends on noisy data

$\Rightarrow$ partition with minimal costs is a r.v. of noisy data

$\Rightarrow$ **estimate clustering solution in a noise insensitive / robust way!**

Part II: Optimization Methods

Given: cost function to rank different object partitions

Robustness: cost function depends on noisy data

$\Rightarrow$ partition with minimal costs is a r.v. of noisy data

$\Rightarrow$ **estimate clustering solution in a noise insensitive / robust way!**

Candidate Solutions: typical or averages of partitions

The Maximum Entropy Principle

Principle: (Jaynes 1957) **Estimate expectation values of optimization variables which are maximally non-committal with respect to missing data.**

The Maximum Entropy Principle

Principle: (Jaynes 1957) **Estimate expectation values of optimization variables which are maximally non-committal with respect to missing data.**

Strategy: (i) Stochastic search through space of partitions with expected costs $\mathrm{E}\{H\} \leq const(T)$.
(ii) Estimate or approximate probability distribution with $\mathrm{E}\{H\} \leq const(T)$

The Maximum Entropy Principle

Principle: (Jaynes 1957) **Estimate expectation values of optimization variables which are maximally non-committal with respect to missing data.**

Strategy: (i) Stochastic search through space of partitions with expected costs $\mathrm{E}\{H\} \leq const(T)$.
(ii) Estimate or approximate probability distribution with $\mathrm{E}\{H\} \leq const(T)$

Algorithm: Markov Chain Monte Carlo or Mean Field Approximation

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution
3. $p \leftarrow \min\{\exp(-(H(c') - H(c))/T), 1\}$;

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution
3. $p \leftarrow \min\{\exp(-(H(c') - H(c))/T), 1\}$;
4. draw $b \sim$ Bernoulli(p);

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution
3. $p \leftarrow \min\{\exp(-(H(c') - H(c))/T), 1\}$;
4. draw $b \sim$ Bernoulli(p);
5. if $b = 1$ then $c \leftarrow c'$;

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution
3. $p \leftarrow \min\{\exp(-(H(c') - H(c))/T), 1\}$;
4. draw $b \sim$ Bernoulli(p);
5. if $b = 1$ then $c \leftarrow c'$;
6. $t \leftarrow t + 1$; if $\neg$ converged goto 2 else stop

Metropolis Sampler for Clustering

Input: n objects, cost function $H(c)$
Output: partition $c:$ objects $\rightarrow$ clusters

MCMC Algorithm:

1. initialize $c(i) \in \{1, \ldots, k\}$ randomly;
2. draw $c' \sim Q(c)$; // $Q(c)$: proposal distribution
3. $p \leftarrow \min\{\exp(-(H(c') - H(c))/T), 1\}$;
4. draw $b \sim$ Bernoulli(p);
5. if $b = 1$ then $c \leftarrow c'$;
6. $t \leftarrow t + 1$; if $\neg$ converged goto 2 else stop

Variants: Gibbs sampling, importance sampling, (competitive!?)

Maximum Entropy Distribution

ME Distribution: $\mathbf{P}^\star = \arg\max_{\mathbf{P}}\{S(\mathbf{P}) : \mathsf{E}\{H\} = const(T)\}$

Maximum Entropy Distribution

ME Distribution: $\mathbf{P}^\star = \arg\max_{\mathbf{P}} \{ S(\mathbf{P}) : \mathsf{E}\{H\} = const(T) \}$

Free Energy: $\underbrace{F(\mathbf{P}(\theta))}_{\text{free energy}} := \mathsf{E}_{\mathbf{P}(\theta)}\{H\} - T \underbrace{S(\mathbf{P}(\theta))}_{\text{entropy}}$

Maximum Entropy Distribution

ME Distribution: $\mathbf{P}^\star = \arg\max_{\mathbf{P}} \{ S(\mathbf{P}) : \mathsf{E}\{H\} = const(T) \}$

Free Energy: $\underbrace{F(\mathbf{P}(\theta))}_{\text{free energy}} := \mathsf{E}_{\mathbf{P}(\theta)}\{H\} - T \underbrace{S(\mathbf{P}(\theta))}_{\text{entropy}}$

- free energy minimization equals maximization of entropy $S(\mathbf{P}) = -\sum_{c \in \mathcal{C}} \mathbf{P}(\theta) \log \mathbf{P}(\theta)$ with fixed expected costs.

$$\left. \frac{\partial}{\partial \mathbf{P}} F \right|_{\sum \mathbf{P} = 1} = H(c) + T \log \mathbf{P}(\theta) + const = 0$$

Maximum Entropy Distribution

ME Distribution: $\mathbf{P}^\star = \arg\max_{\mathbf{P}} \{ S(\mathbf{P}) : \mathsf{E}\{H\} = const(T) \}$

Free Energy: $\underbrace{F(\mathbf{P}(\theta))}_{\text{free energy}} := \mathsf{E}_{\mathbf{P}(\theta)}\{H\} - T \underbrace{S(\mathbf{P}(\theta))}_{\text{entropy}}$

- free energy minimization equals maximization of entropy $S(\mathbf{P}) = -\sum_{c \in \mathcal{C}} \mathbf{P}(\theta) \log \mathbf{P}(\theta)$ with fixed expected costs.

$$\left. \frac{\partial}{\partial \mathbf{P}} F \right|_{\sum \mathbf{P} = 1} = H(c) + T \log \mathbf{P}(\theta) + const = 0$$

$\Rightarrow$ Gibbs distribution $\mathbf{P}(\tilde{\theta}) = \exp\left(-\big(H(c) - F(\tilde{\theta}) \big) / T \right)$

Algorithm Design for Maximum Entropy Estimation

Calculate the Gibbs distribution of object partitions.

$\Rightarrow$ expectation values of optimization variables

Algorithm Design for Maximum Entropy Estimation

Calculate the Gibbs distribution of object partitions.

$\Rightarrow$ expectation values of optimization variables

Maximize entropy w.r.t. additional free parameters.

Algorithm Design for Maximum Entropy Estimation

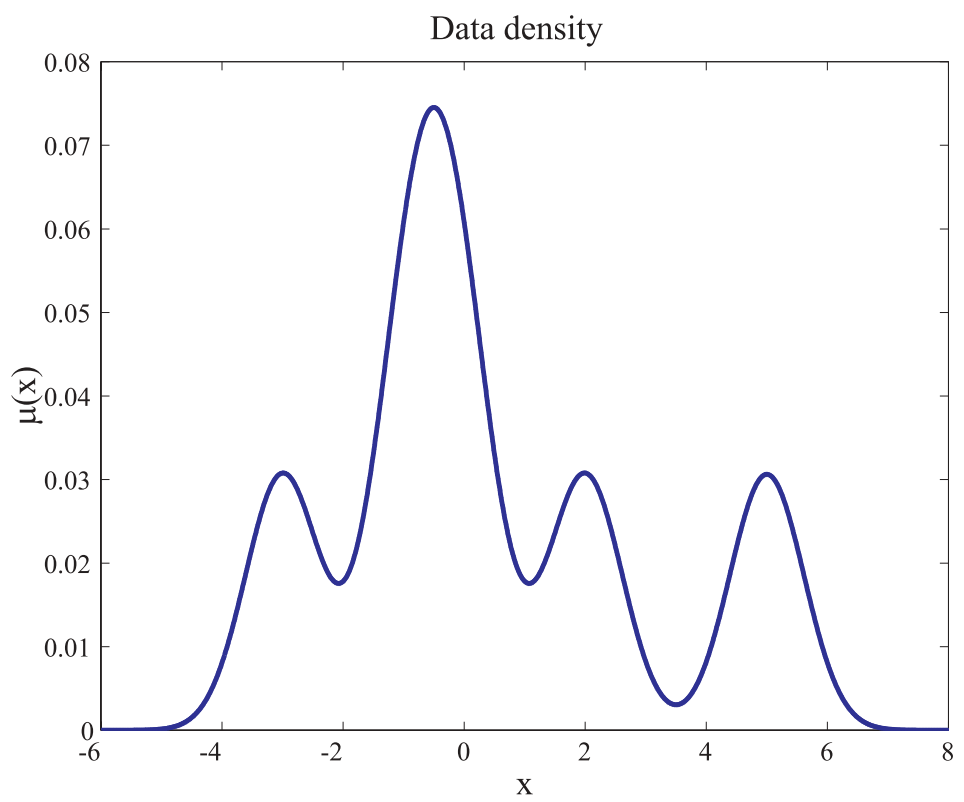
Calculate the Gibbs distribution of object partitions.
$\Rightarrow$ expectation values of optimization variables

Maximize entropy w.r.t. additional free parameters.

EM-like Iteration: Both steps are iterated until convergence to a local maximum of the entropy is achieved.

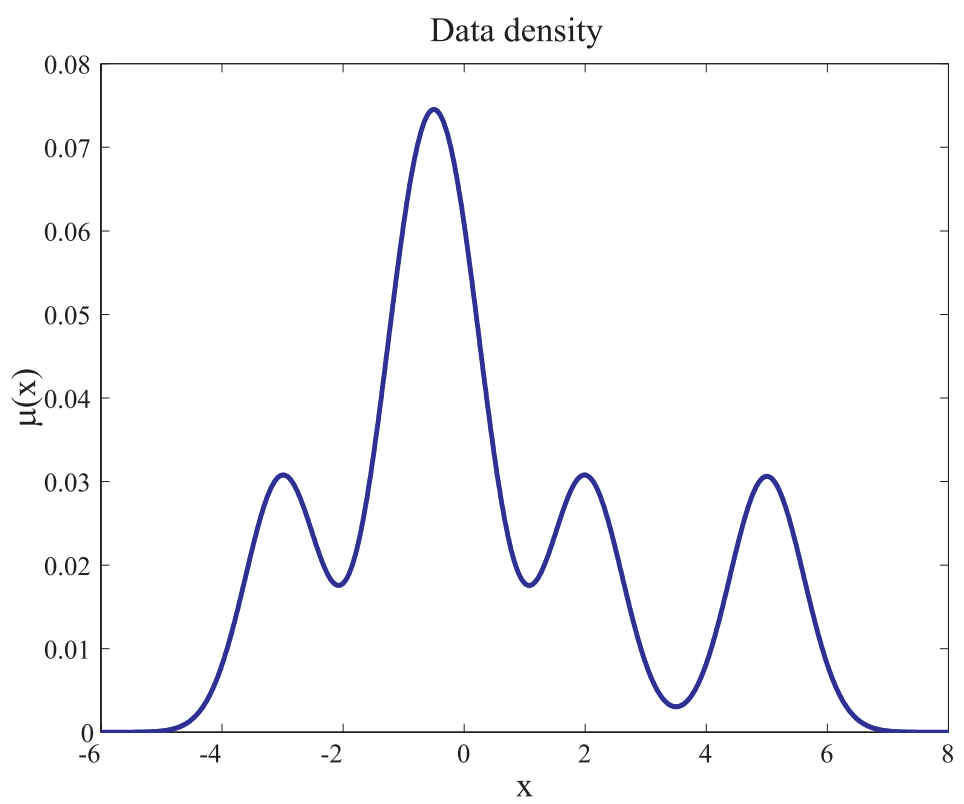
Phase Transitions in K-means Clustering

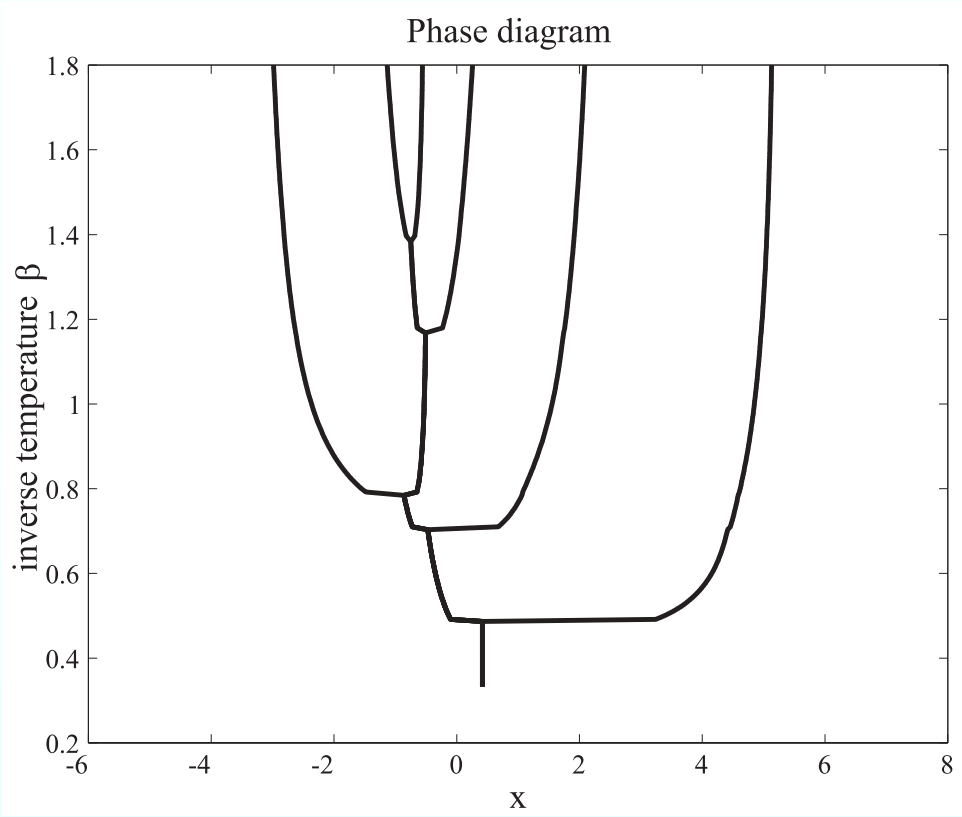
1-dim. mixture model; track centroids as a function of temperature.



Phase Transitions in K-means Clustering

1-dim. mixture model; track centroids as a function of temperature.





EM Update Scheme for PDC

E-step

$$h_{i\nu} = -\log p_\nu - \sum_j n_{ij} \log \left(\sum_\alpha p_{\alpha|\,\nu} \tilde{G}_\alpha (j) \right)$$

$$q_{i\nu} = E\left[M_{i\nu}\right] \propto \exp\left(-\frac{h_{i\nu}}{T}\right)$$

EM Update Scheme for PDC

E-step

$$h_{i\nu} = -\log p_\nu - \sum_j n_{ij} \log \left(\sum_\alpha p_{\alpha|\,\nu} \tilde{G}_\alpha(j) \right)$$

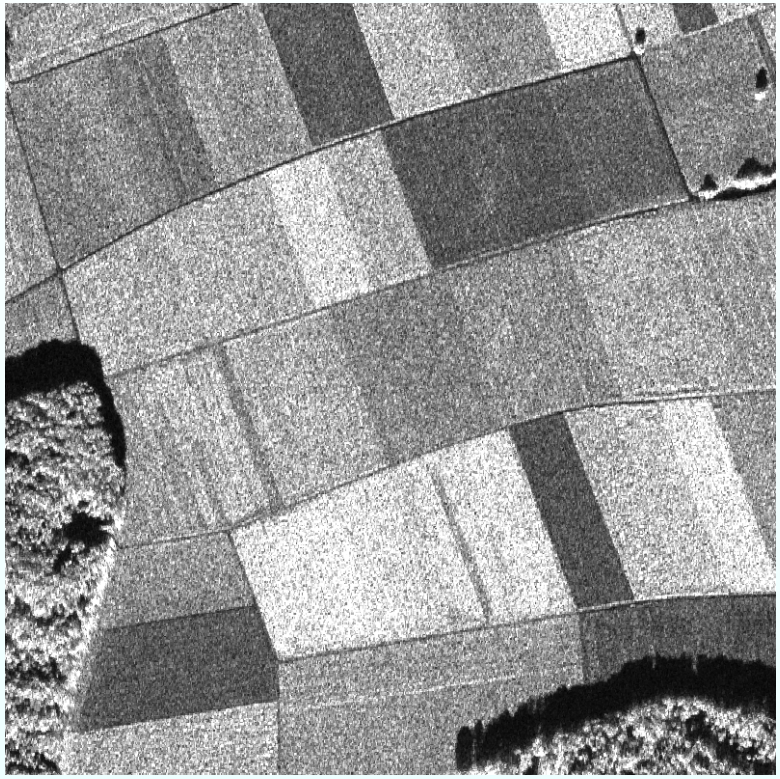
$$q_{i\nu} = E[M_{i\nu}] \propto \exp\left(-\frac{h_{i\nu}}{T} \right)$$

M-step

$$p_\nu = \frac{1}{n} \sum_i q_{i\nu}$$

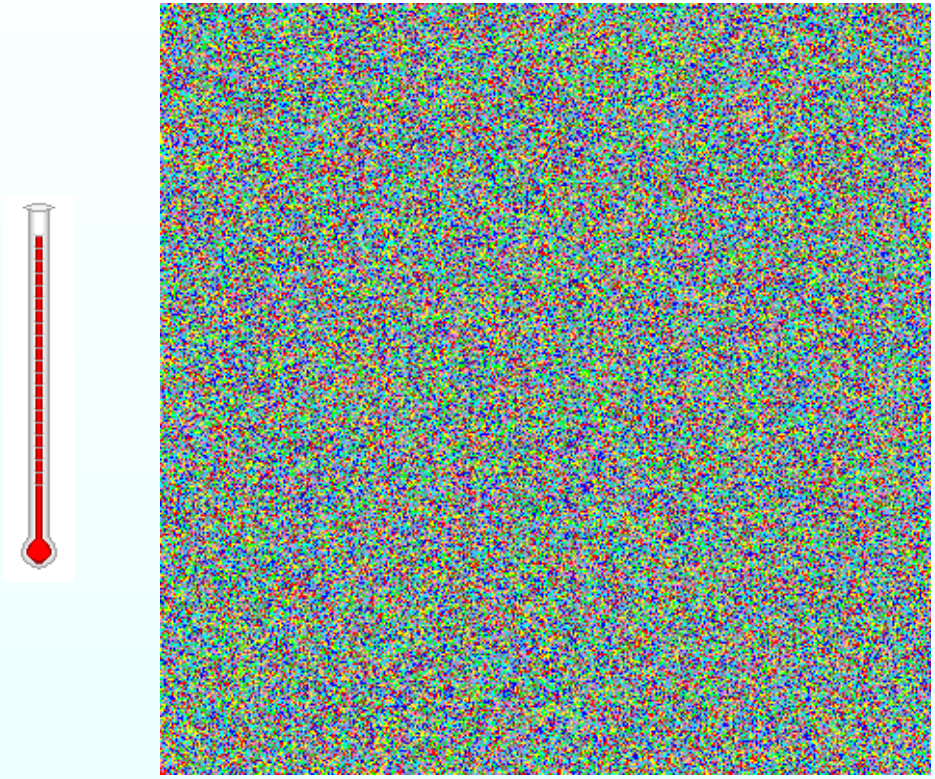
No closed formula for $p_{\alpha|\,\nu}$, nor for μ_α! We iteratively optimize pairs $p_{\alpha_1|\,\nu}$, $p_{\alpha_2|\,\nu}$ until convergence. Interval bisection is used to optimize Gaussian means μ_α.

Cooling Dynamics of PDC



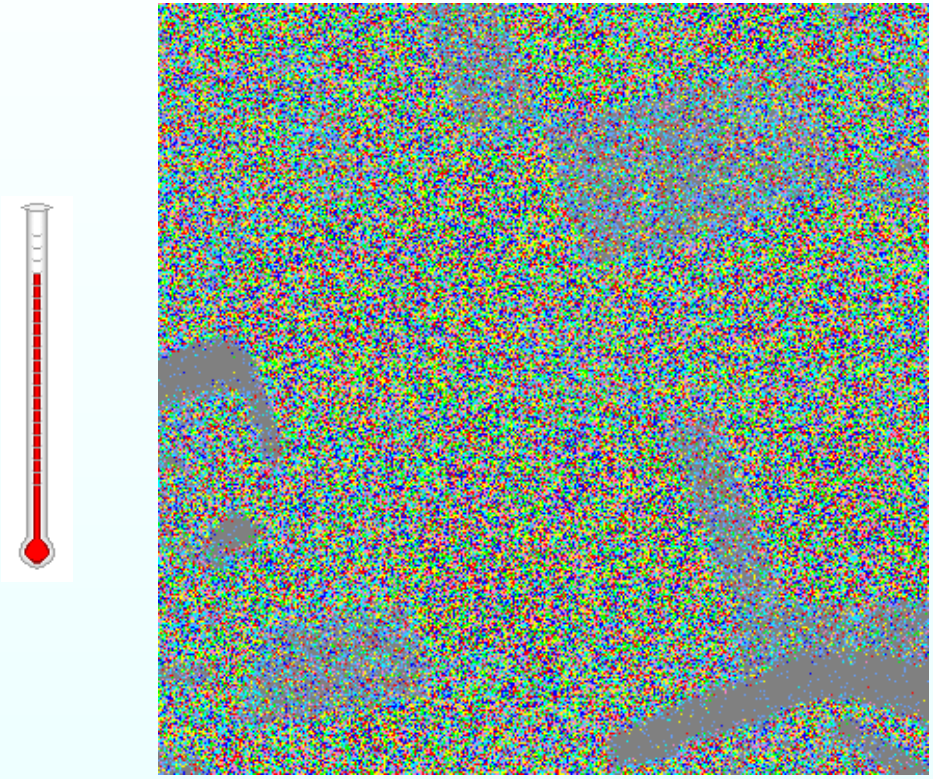
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



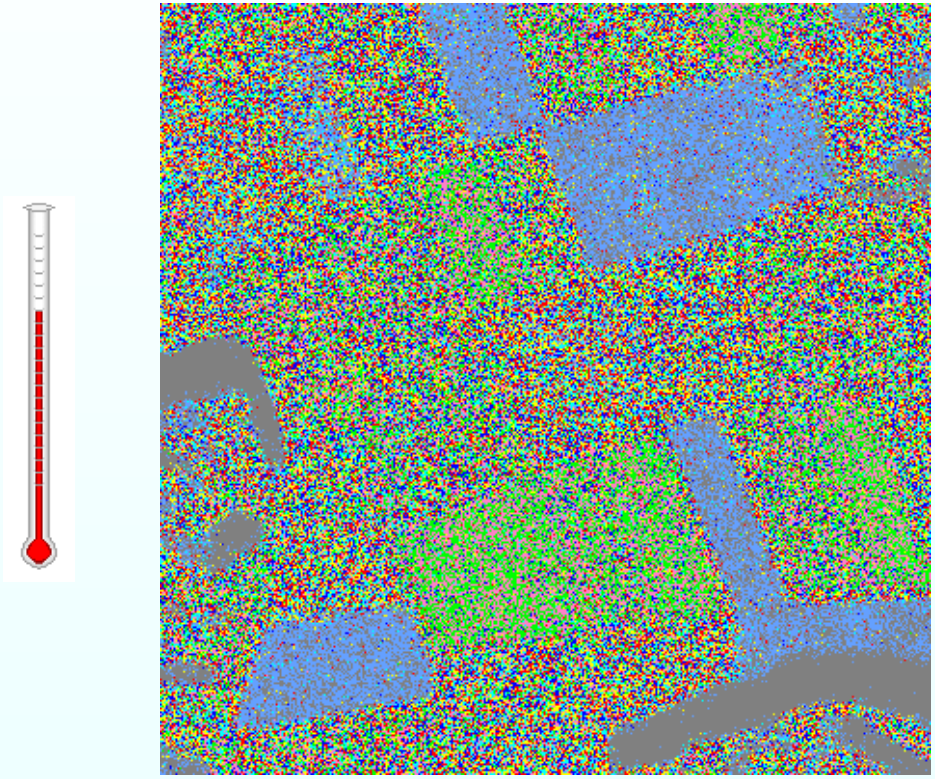
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



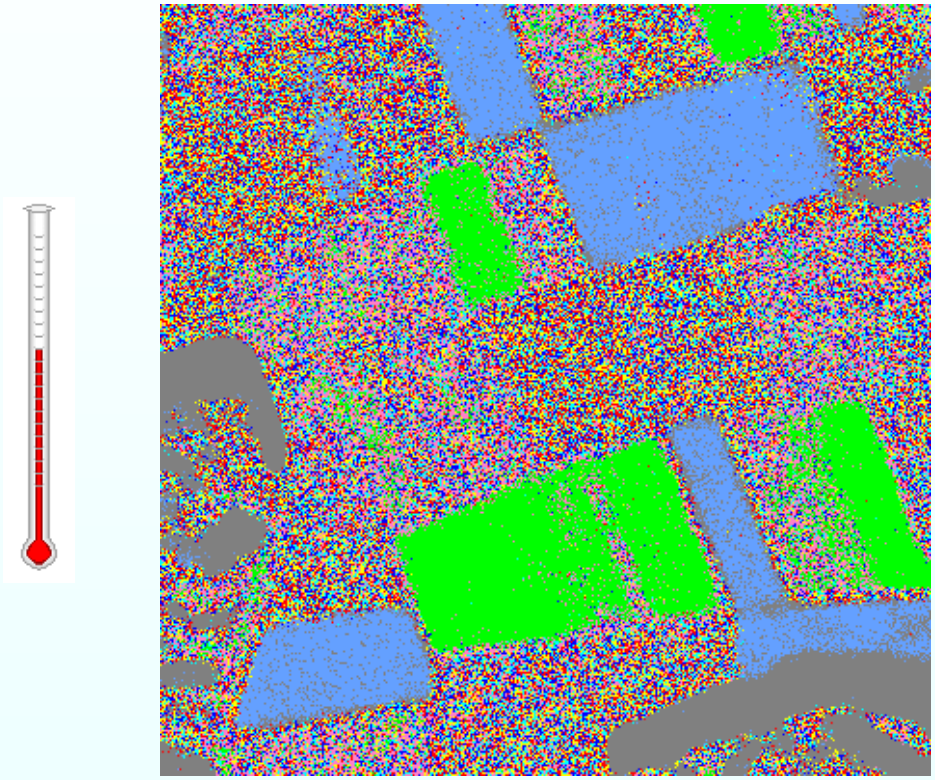
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



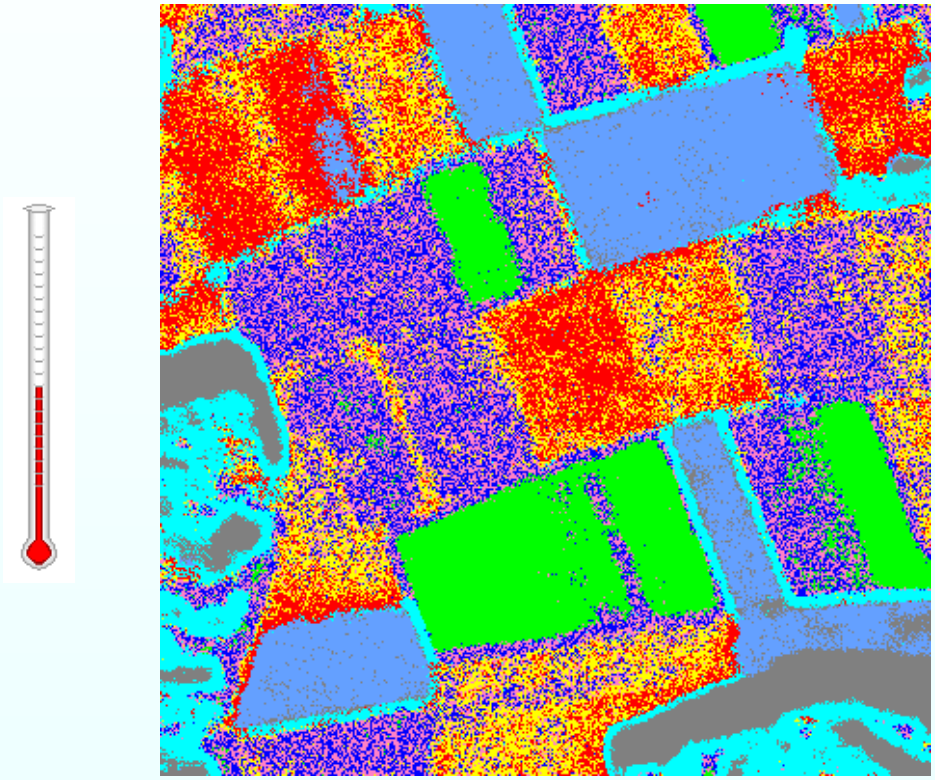
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



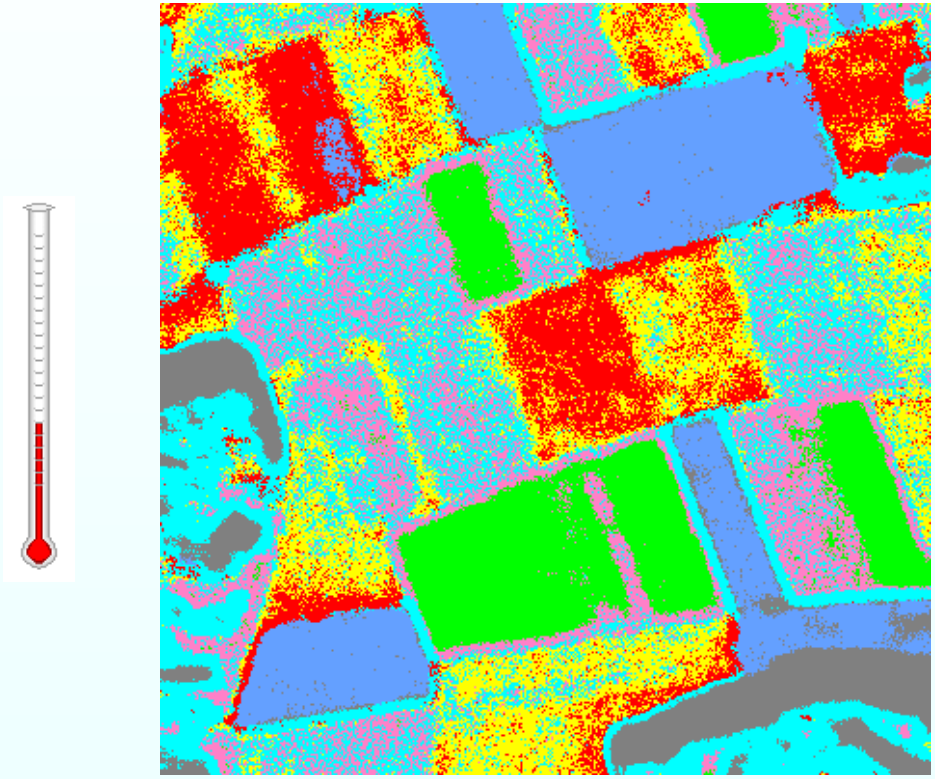
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



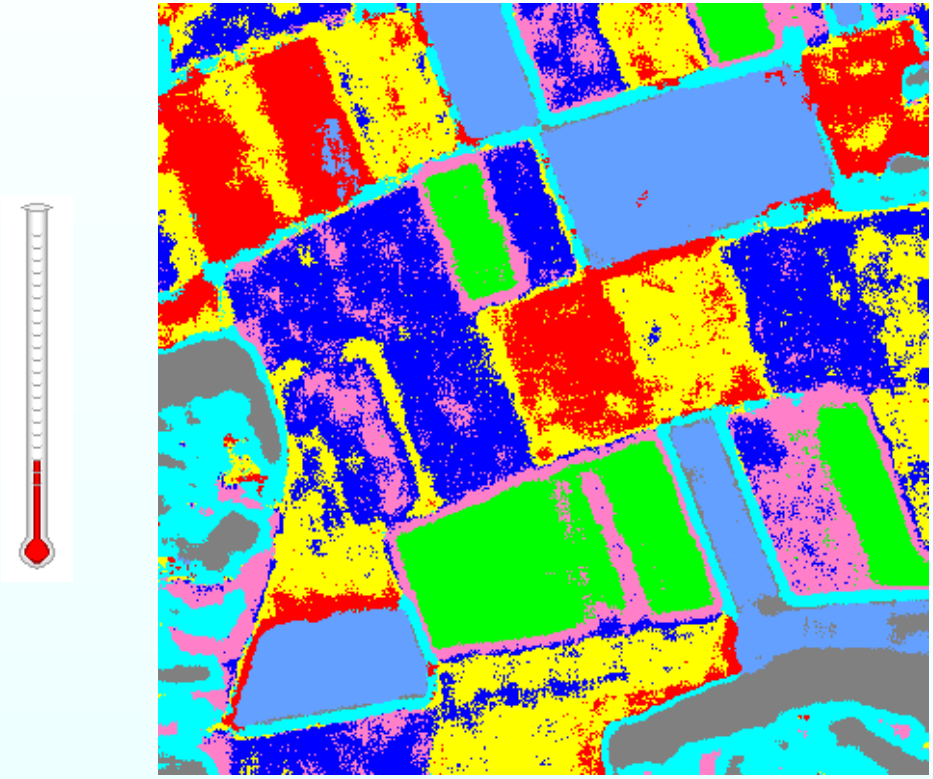
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



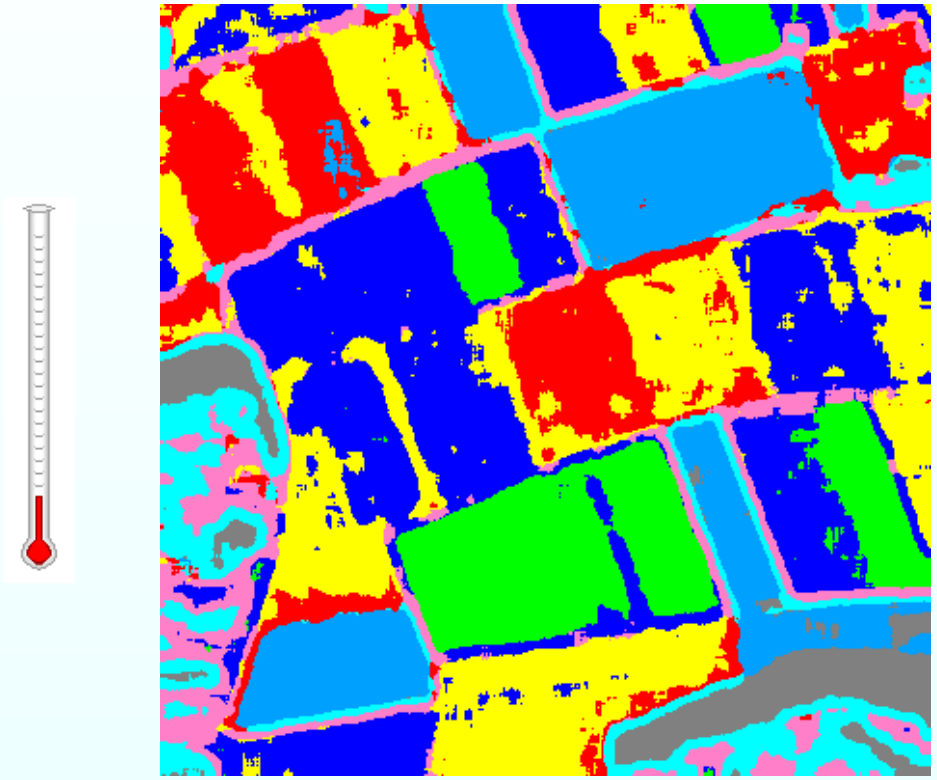
Level of randomness decreases while lowering the computational temperature.

Cooling Dynamics of PDC



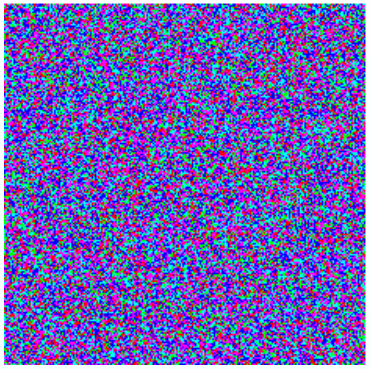
Level of randomness decreases while lowering the computational temperature.

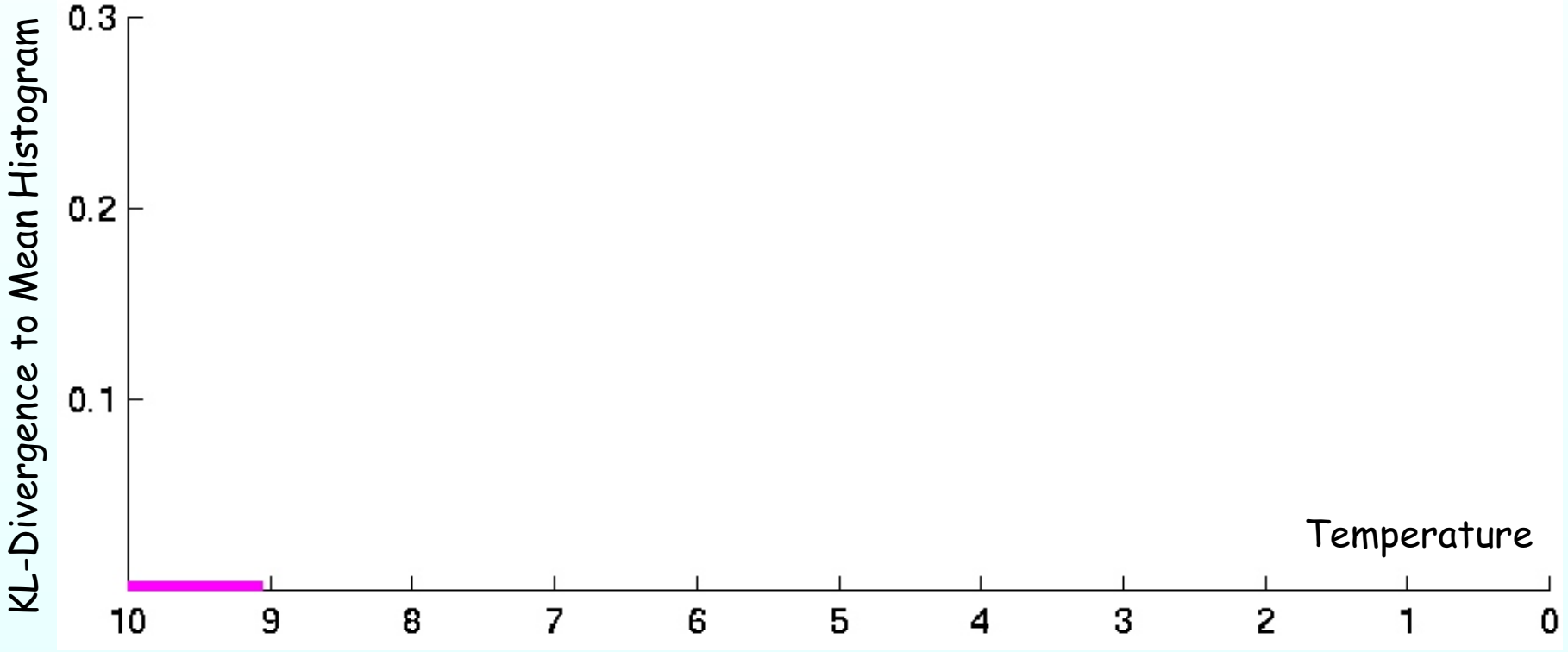
Cooling Dynamics of PDC



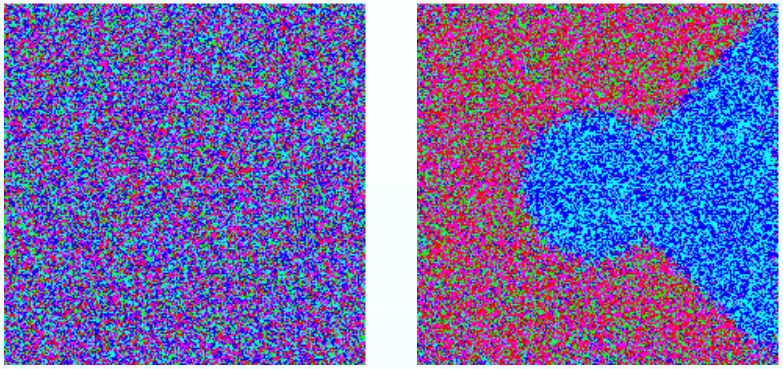
Level of randomness decreases while lowering the computational temperature.

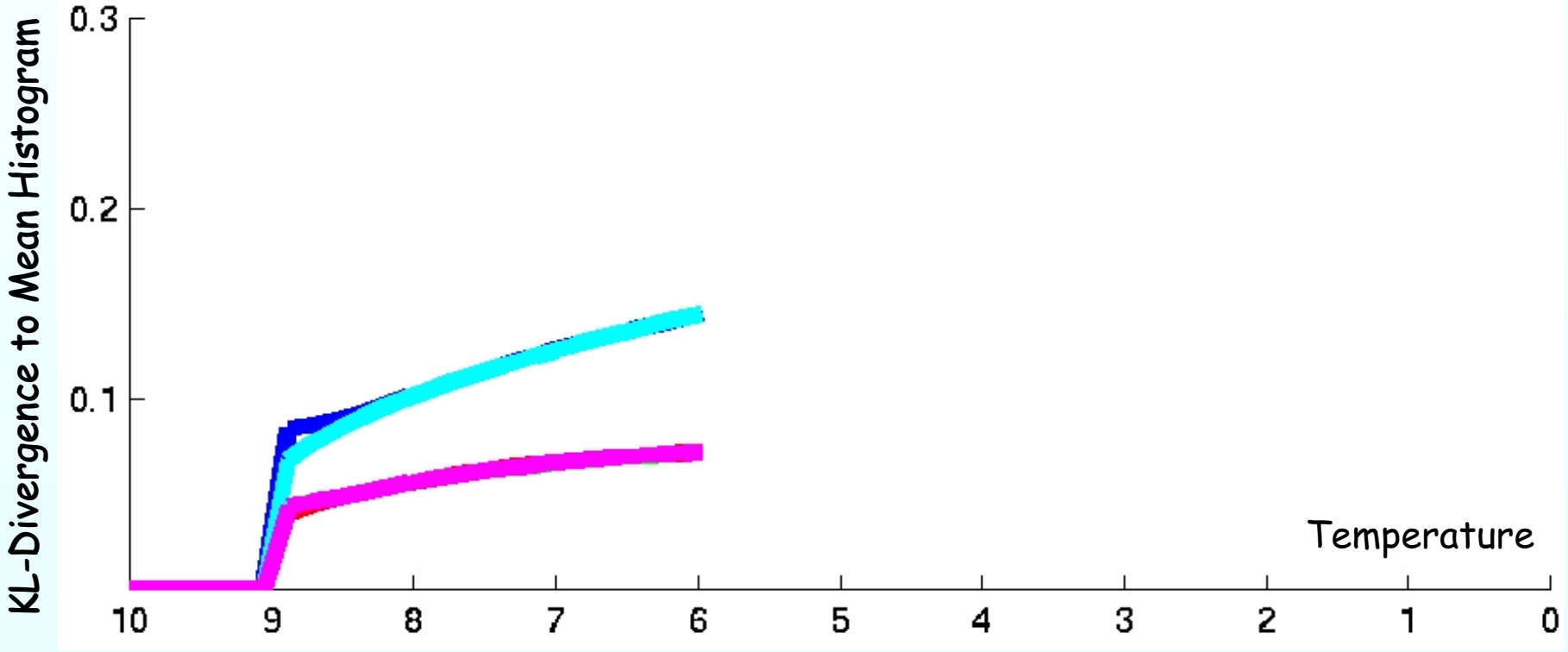
Phase Transitions in Segmentation



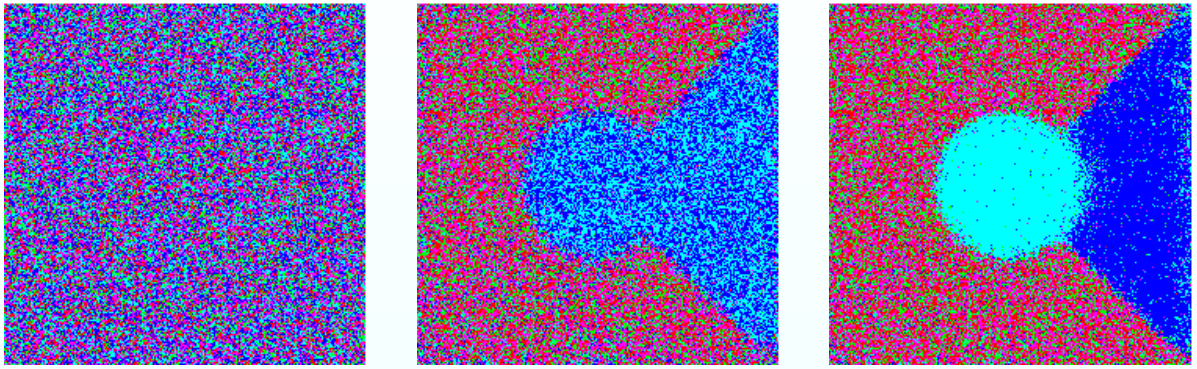


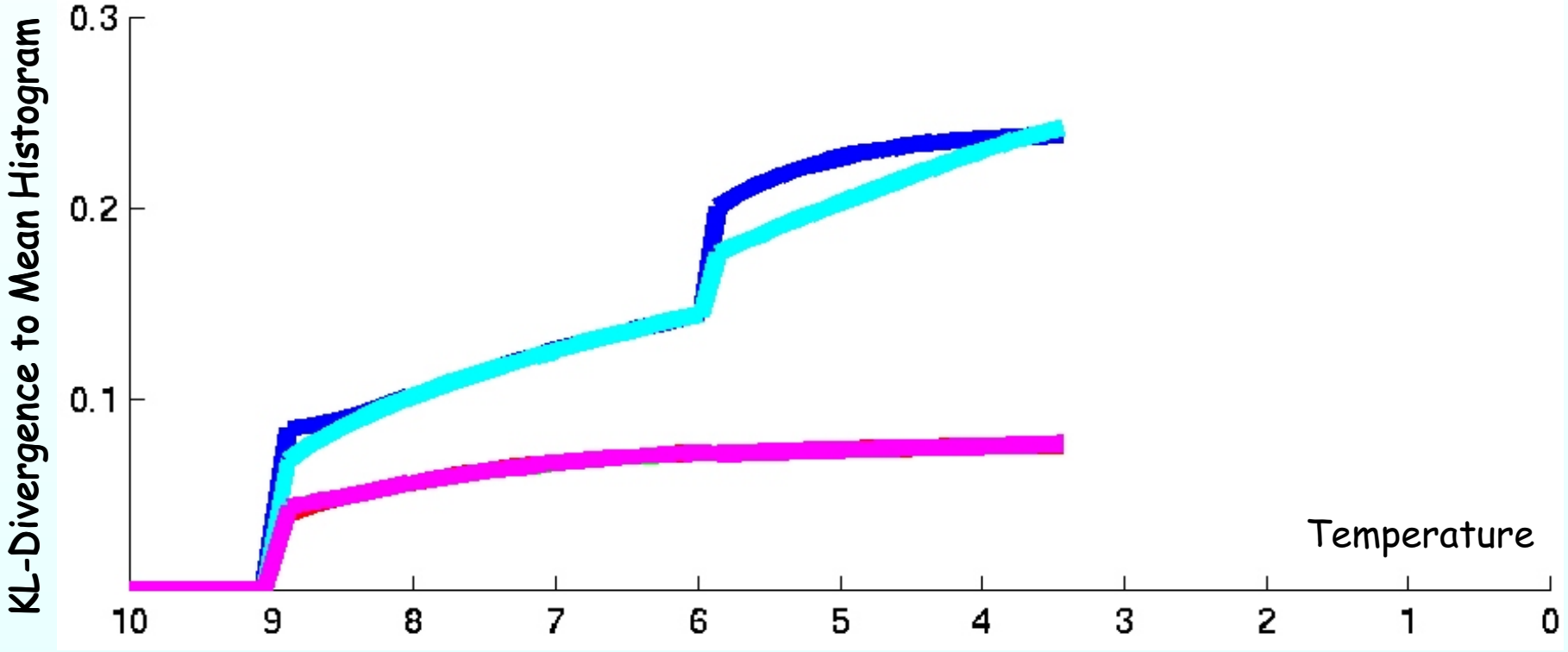
Phase Transitions in Segmentation



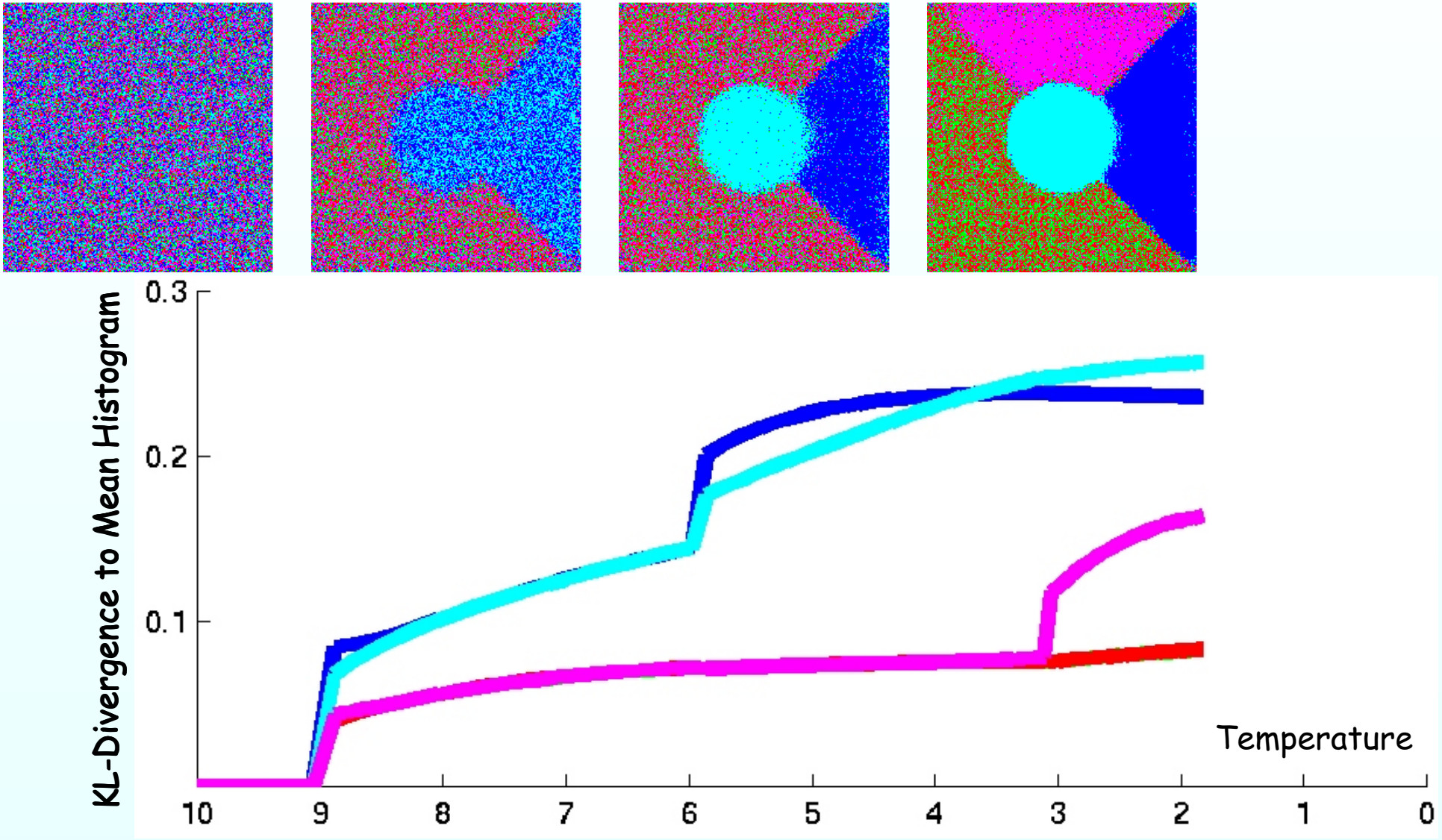


Phase Transitions in Segmentation

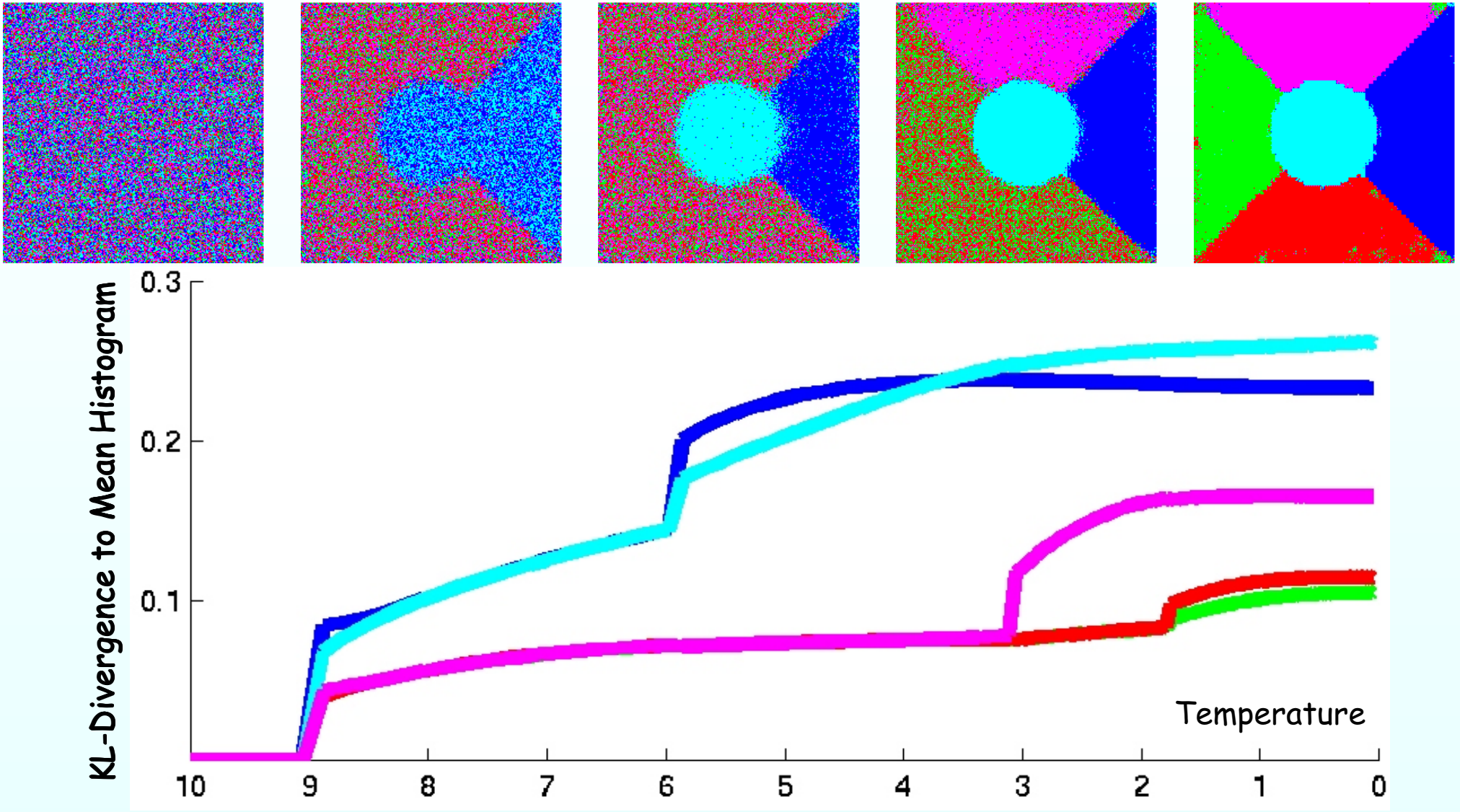




Phase Transitions in Segmentation

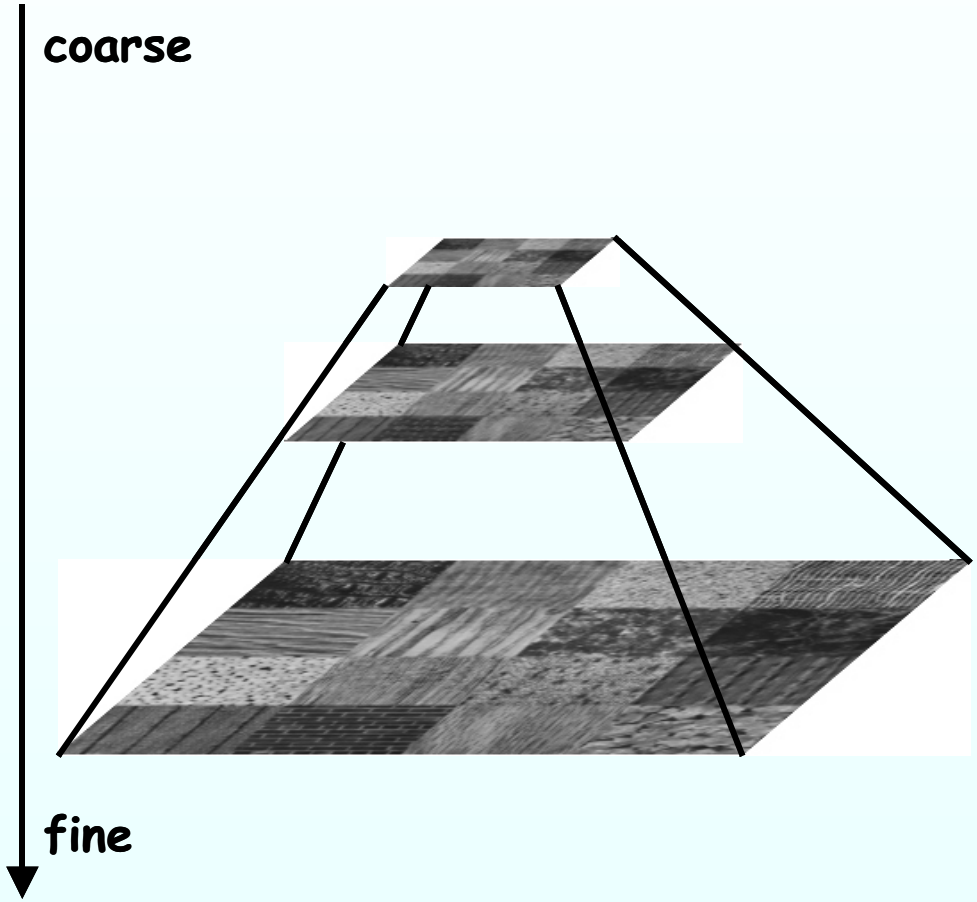


Phase Transitions in Segmentation



Scales in Vision and their Coupling

Coarsening
Variable Space

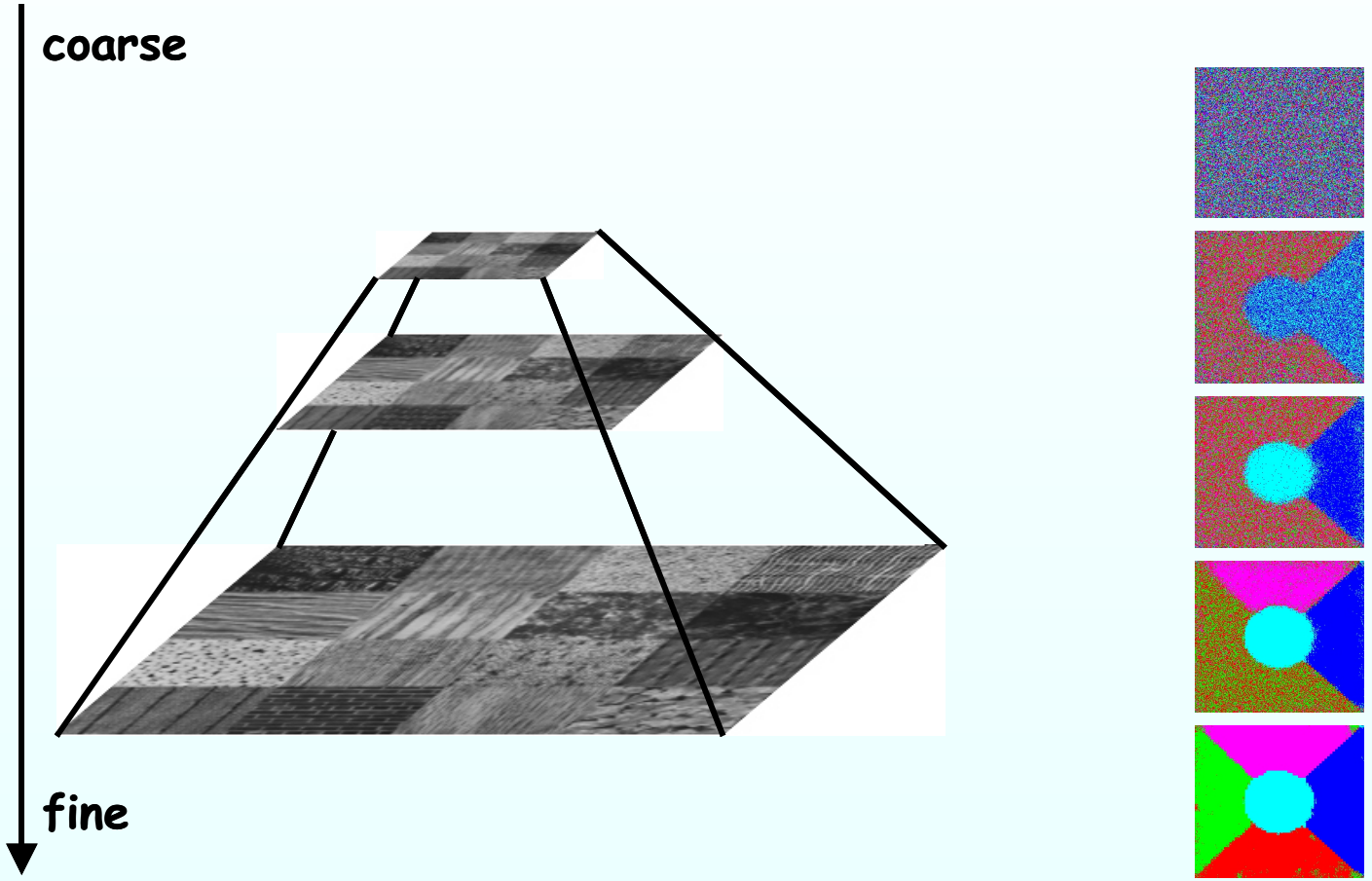


Resolution Pyramid

Scales in Vision and their Coupling

Coarsening
Variable Space

Coarsening
Optimization Criterion



Resolution Pyramid

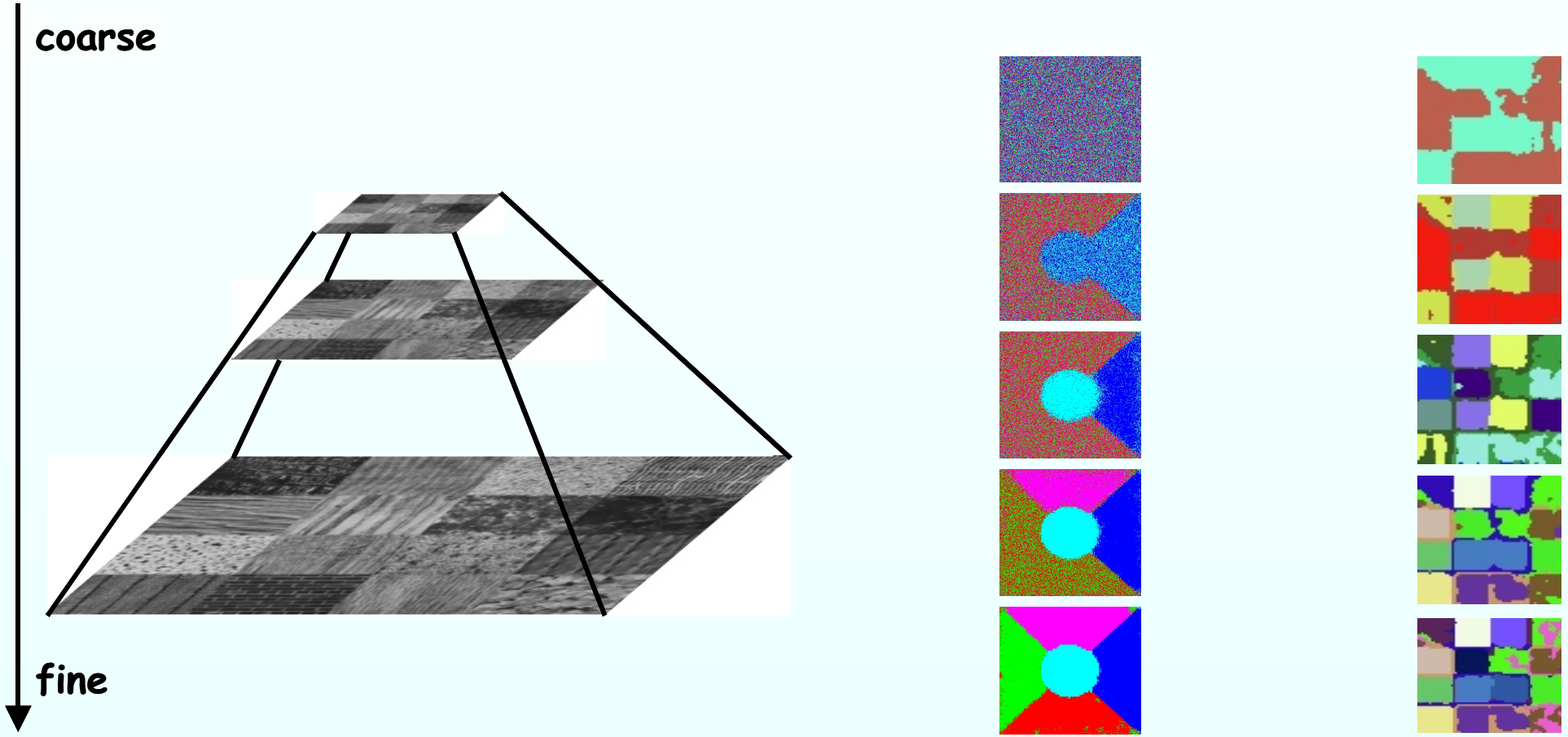
Raising Temperature

Scales in Vision and their Coupling

Coarsening
Variable Space

Coarsening
Optimization Criterion

Coarsening
Model Order



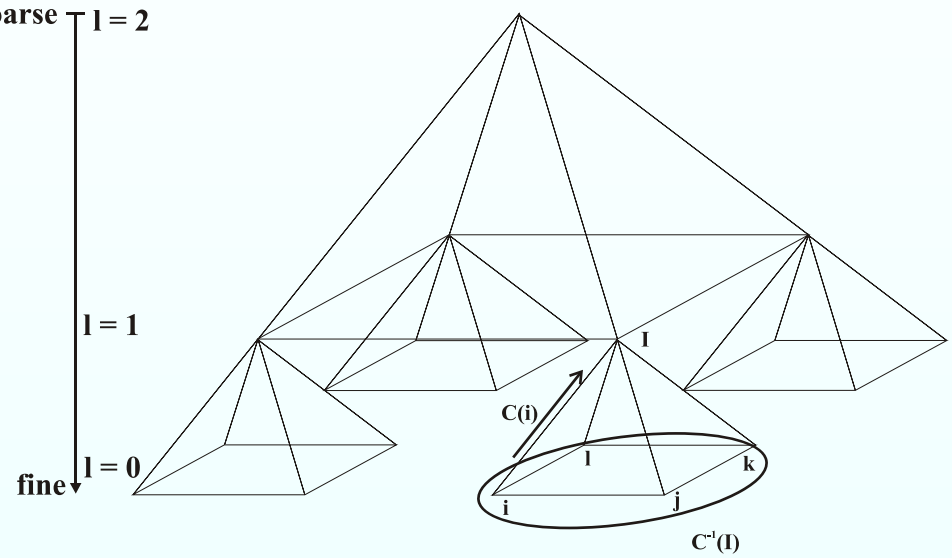
Resolution Pyramid

Raising Temperature

Decr. Number of Segments

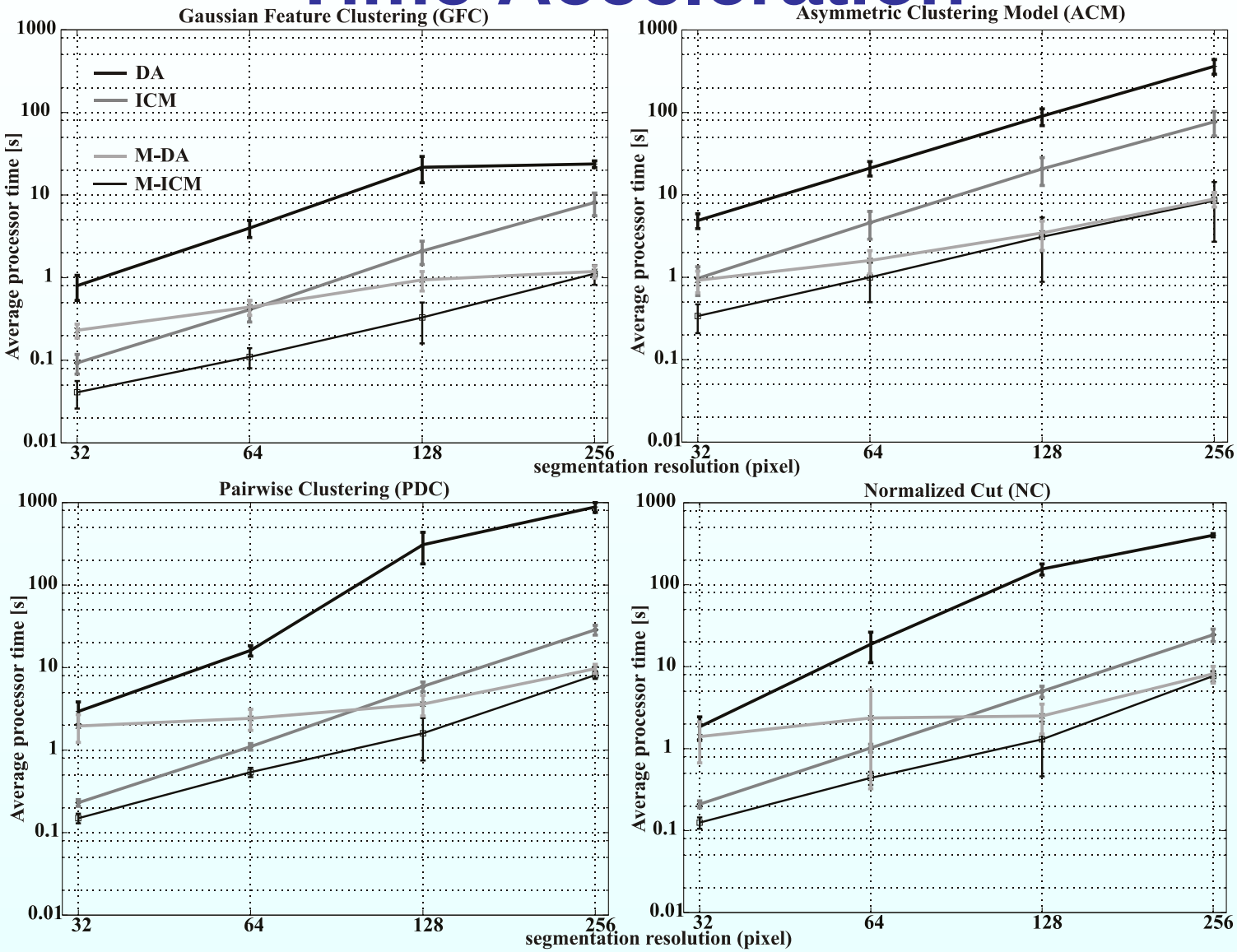
Multi-Scale Optimization

- Coarser set of image sites $S^\ell = \{s_1^\ell, \ldots, s_{n_\ell}^\ell\}$
- Prolongation operator P_ℓ : $S^{\ell+1} \rightarrow S^\ell$ defines map between two resolution levels

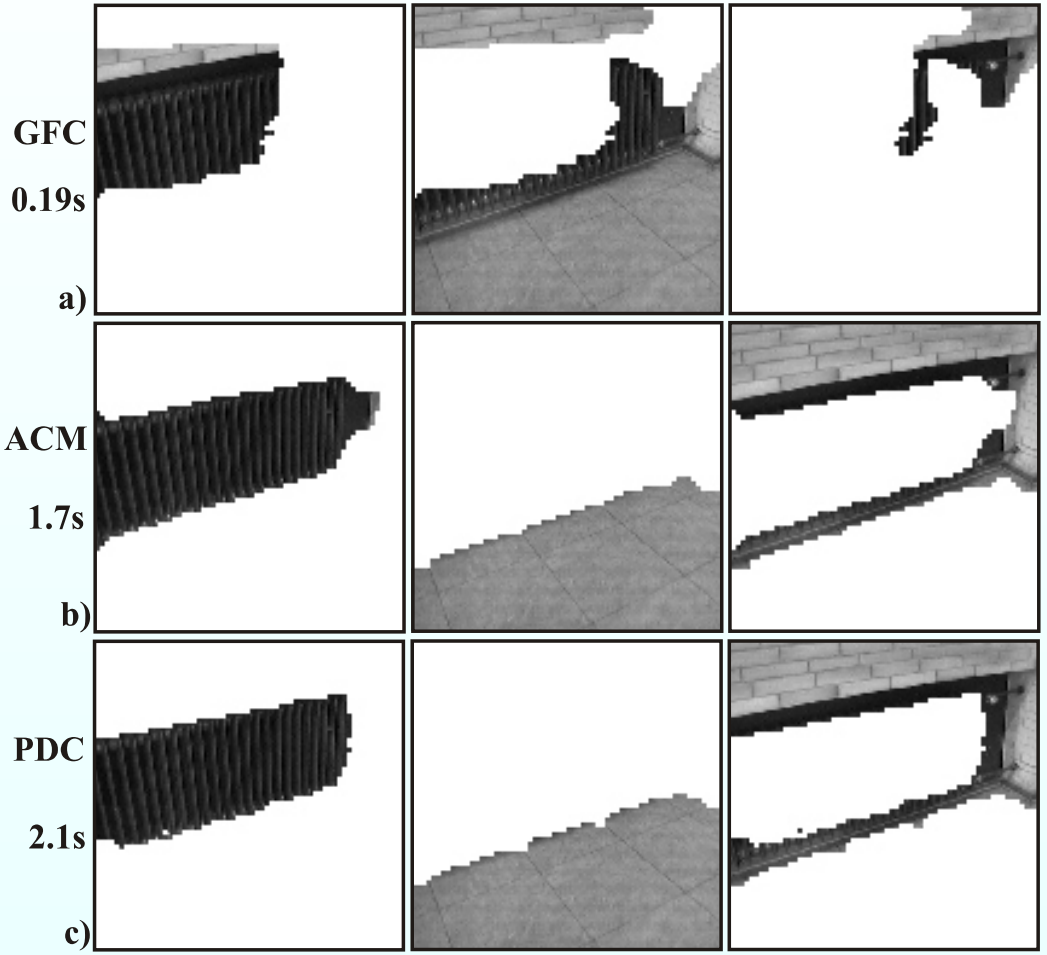


- Multiscale Operator $H^{\ell+1}(S^{l+1}) = \Gamma(H^\ell) = H^\ell(P(S^{\ell+1}))$ maps the value of the objective function

Time Acceleration



Example Multiscale optimization



Part III: Cluster Validation

- The Problem of Cluster Validity

Part III: Cluster Validation

- The Problem of Cluster Validity
- Complexity-based Validation *(Rissanen, 1978; Schwarz, 1978)*.

Part III: Cluster Validation

- The Problem of Cluster Validity
- Complexity-based Validation *(Rissanen, 1978; Schwarz, 1978)*.
- Cross-Validated Likelihood *(e.g. P. Smyth, 1998)*

Part III: Cluster Validation

- The Problem of Cluster Validity
- Complexity-based Validation *(Rissanen, 1978; Schwarz, 1978)*.
- Cross-Validated Likelihood *(e.g. P. Smyth, 1998)*
- The Gap Statistic *(Tibshirani, Walther, Hastie, 2001)*

Part III: Cluster Validation

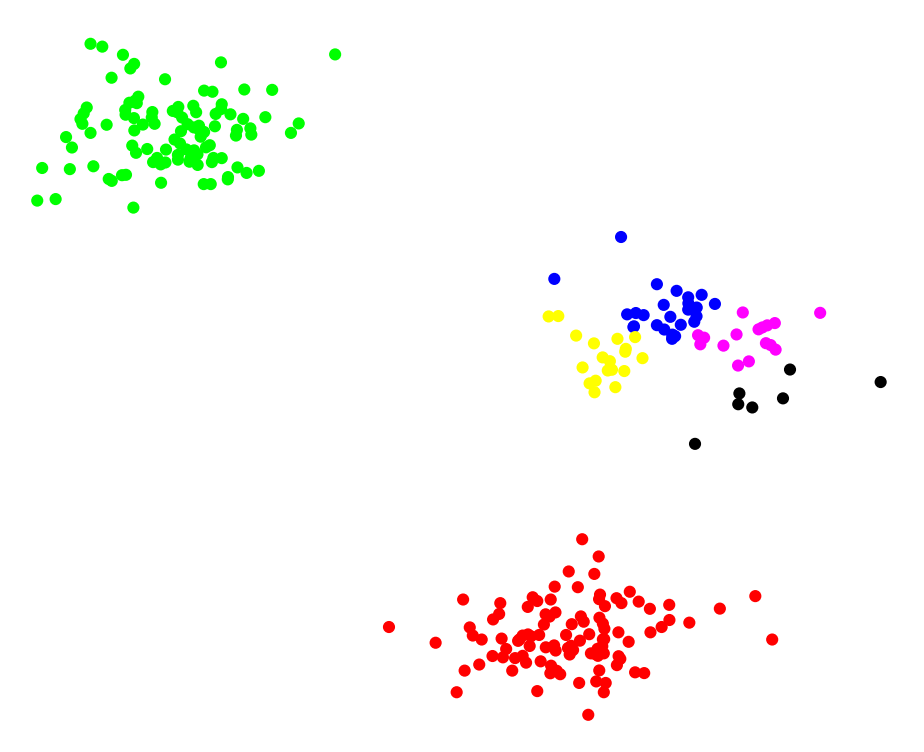
- The Problem of Cluster Validity
- Complexity-based Validation *(Rissanen, 1978; Schwarz, 1978)*.
- Cross-Validated Likelihood *(e.g. P. Smyth, 1998)*
- The Gap Statistic *(Tibshirani, Walther, Hastie, 2001)*
- Stability-based Validation *(Lange, Braun, Roth, Buhmann, 2002)*

The Problem of Cluster Validity

Clustering algorithms always impose structure on data.

The Problem of Cluster Validity

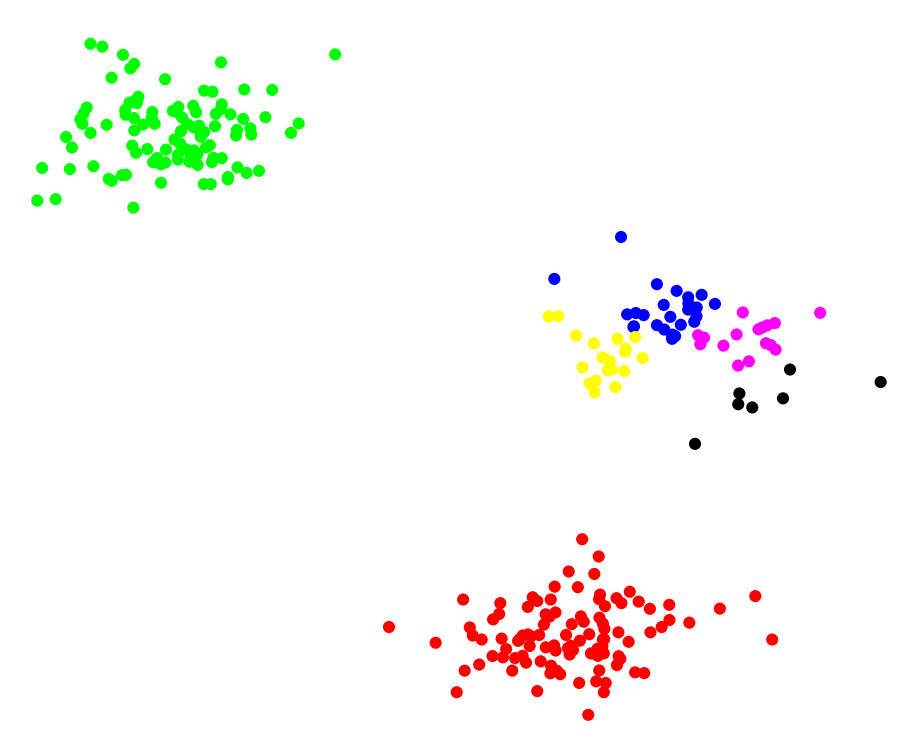
Clustering algorithms always impose structure on data.



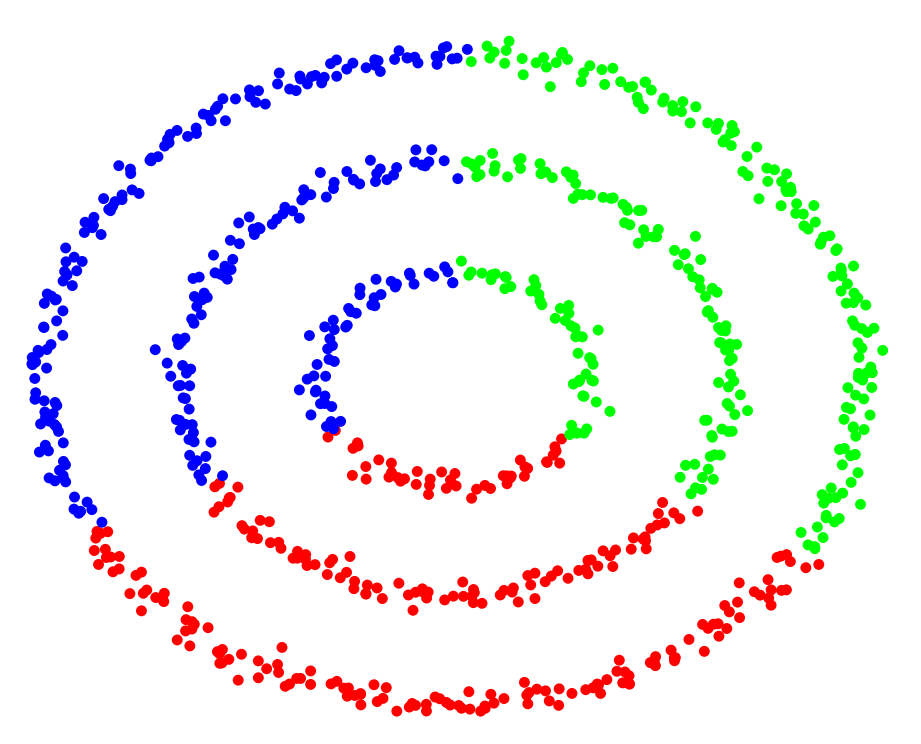
(a) Inappropriate model order.

The Problem of Cluster Validity

Clustering algorithms always impose structure on data.



(a) Inappropriate model order.



(b) Inappropriate model type.

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.
- ... evaluate a specific quality measure.

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.
- ... evaluate a specific quality measure.
- ... can be *external* (= cmp. with ground-truth) or *internal*.

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.
- ... evaluate a specific quality measure.
- ... can be *external* (= cmp. with ground-truth) or *internal*.
- ... can be used for model selection.

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.
- ... evaluate a specific quality measure.
- ... can be *external* (= cmp. with ground-truth) or *internal*.
- ... can be used for model selection.

Important Question:
What is the appropriate number of clusters k for my data?

Validation Methods ...

- ... are procedures and concepts for the quantitative and objective assessment of clustering solutions.
- ... evaluate a specific quality measure.
- ... can be *external* (= cmp. with ground-truth) or *internal*.
- ... can be used for model selection.

Important Question:
What is the appropriate number of clusters k for my data?

General approach: Measure quality for different k!

Complexity-based Validation

Add a complexity term to the neg. log-likelihood in model-based clustering!

Complexity-based Validation

Add a complexity term to the neg. log-likelihood in model-based clustering!

Occam's razor:

> Choose the model that provides the shortest description of the data

Complexity-based Validation

Add a complexity term to the neg. log-likelihood in model-based clustering!

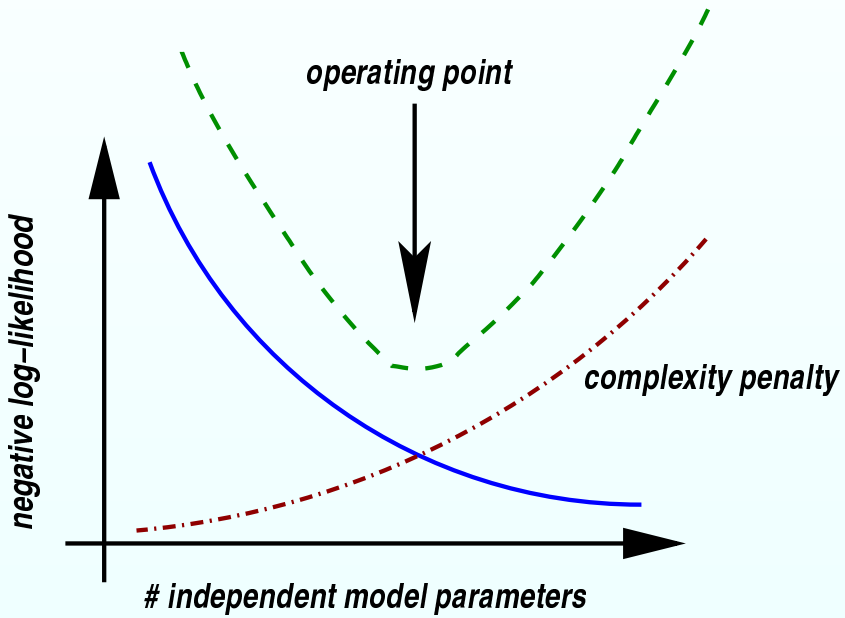
Occam's razor:

> Choose the model that provides the shortest description of the data

Idea formalized e.g. by

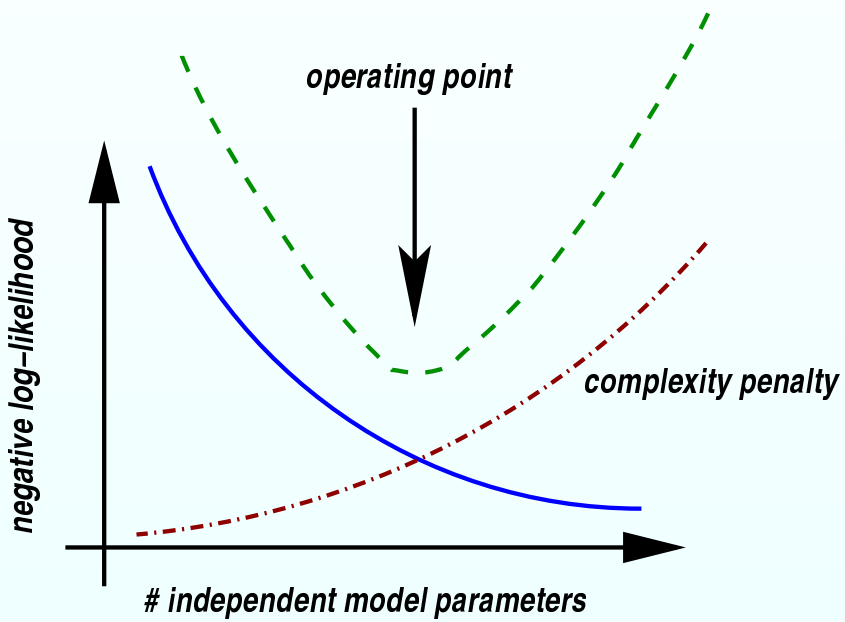
- Rissanen's **Minimum Description Length**.
- Schwartz's **Bayesian Information Criterion**.

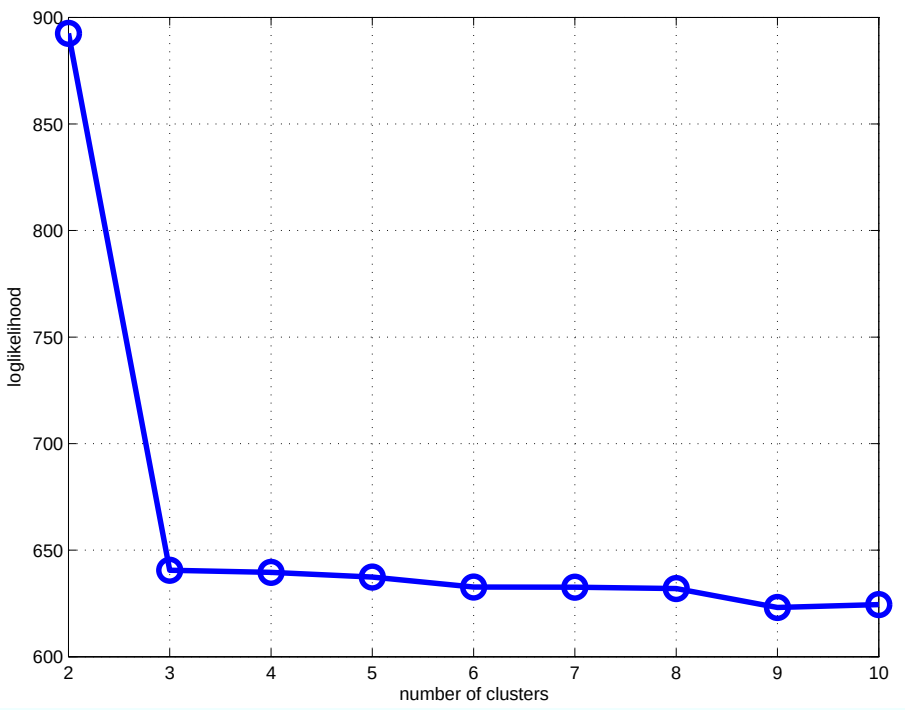
Underlying Principle



The (neg.) log-likelihood decreases with increasing model complexity. Correct this with a complexity penalty.

Underlying Principle





The (neg.) log-likelihood decreases with increasing model complexity. Correct this with a complexity penalty.

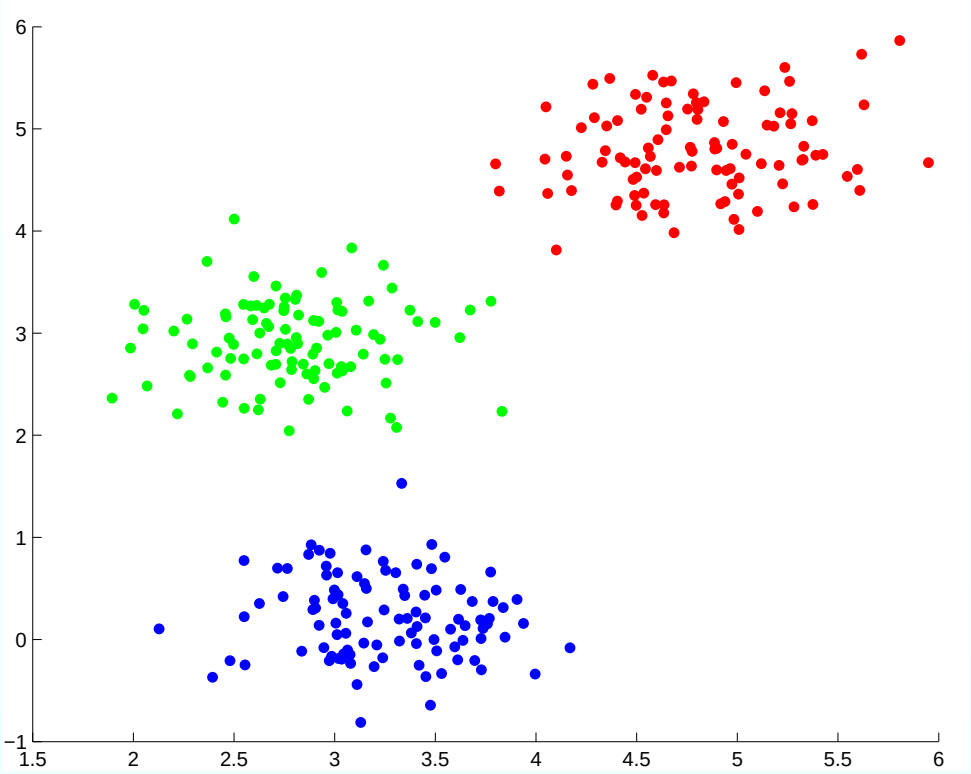
Minimum Description Length *(Rissanen,1978)*

MDL minimizes the overall description length of the data where the description consists of the data and the model parameters.

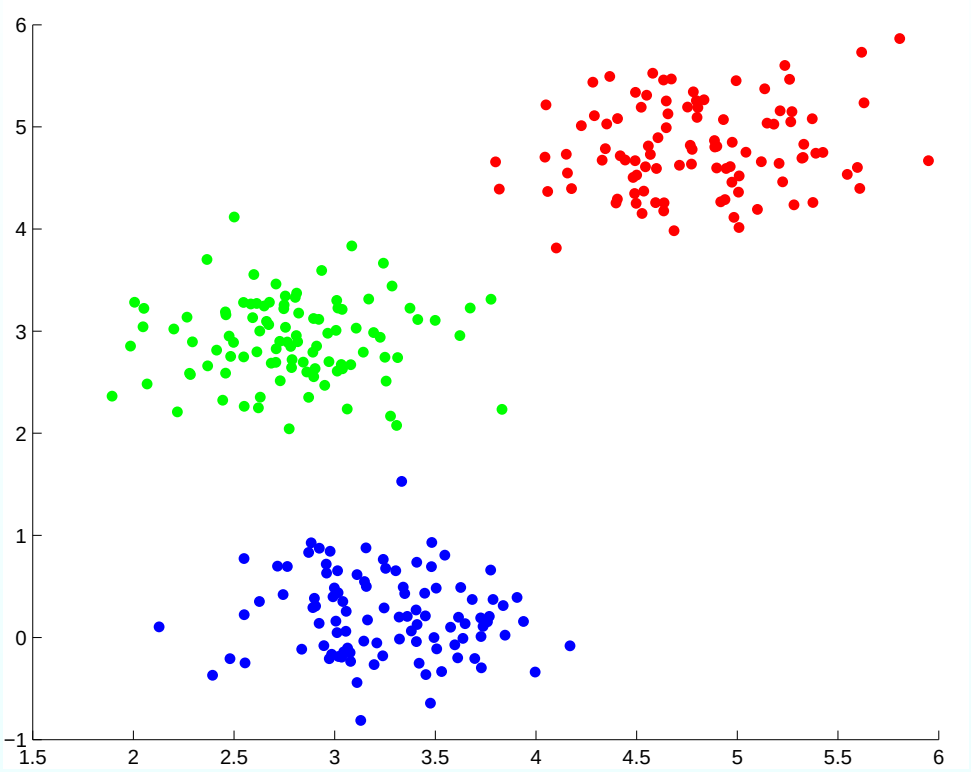
$$\hat{k} := \underset{1 \le k \le K_{\max}}{\operatorname{argmin}} \left(\underbrace{-\log(\hat{\mathrm{p}}(\mathbf{X} \mid \hat{\Phi}_k))}_{\text{negative loglikelihood}} + \underbrace{\frac{1}{2} k' \log n}_{\text{complexity penalty}} \right)$$

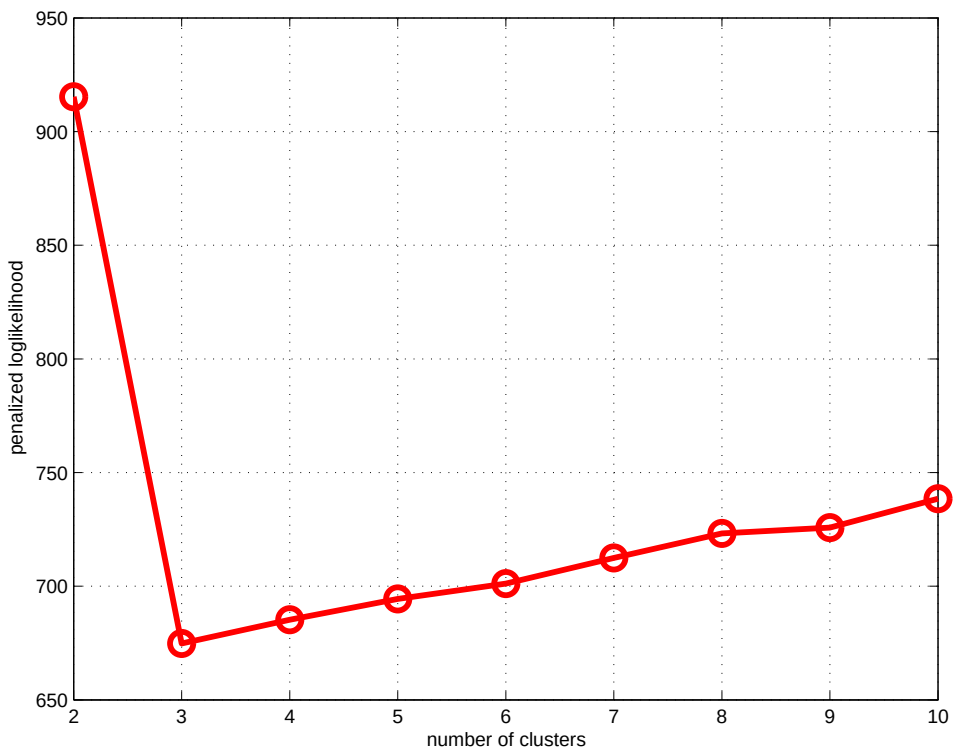
where k' is the number of independent parameters in the model Φ_k.

BIC Validation of a Mixture



BIC Validation of a Mixture





Stability-based Validation *(Lange, Braun, Roth, JB,2002)*

- Many validation methods incorporate a structural bias!

Stability-based Validation *(Lange, Braun, Roth, JB,2002)*

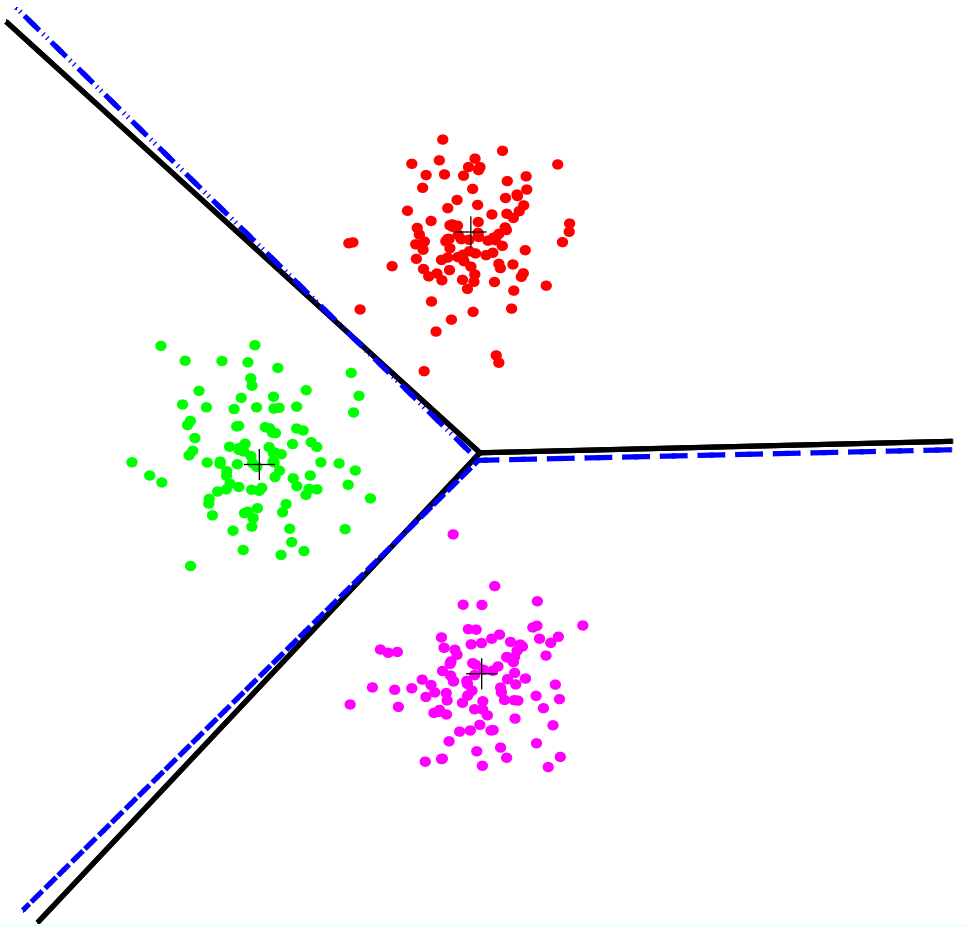
- Many validation methods incorporate a structural bias!
- What to do if no additional a priori knowledge available?

Stability-based Validation *(Lange, Braun, Roth, JB,2002)*

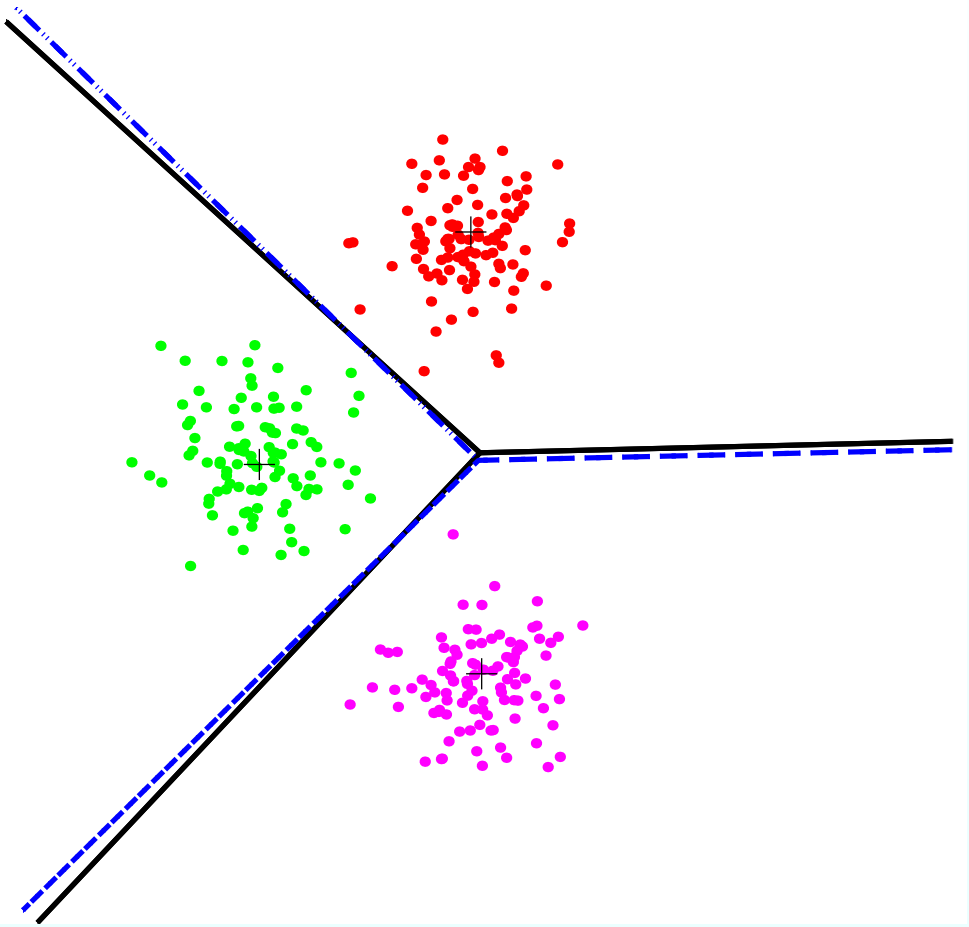
- Many validation methods incorporate a structural bias!
- What to do if no additional a priori knowledge available?
- **Main idea:**

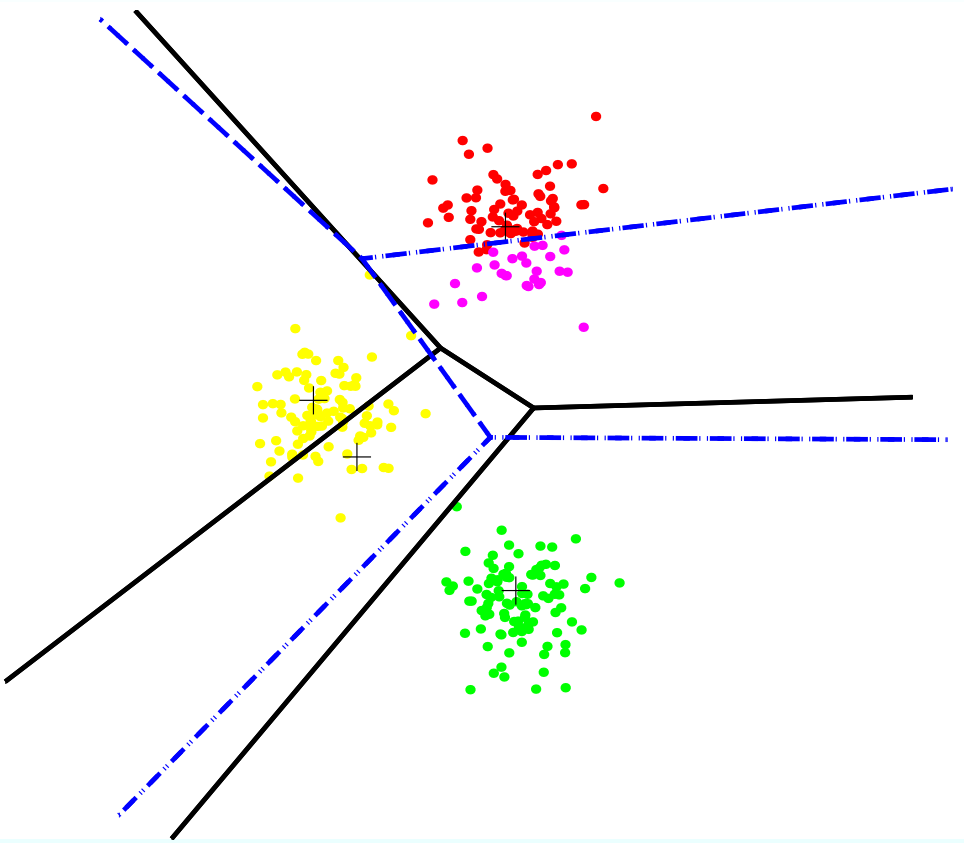
Stability:
Solutions on two data sets from the same source should be similar.

Stability



Stability





Two Sample Scenario

- **General procedure:**

  (i) Draw two data sets from the same source.

  (ii) Cluster both data sets.

  (iii) Compute agreement

Stability := expected agreement of the solutions.

Two Sample Scenario

- **General procedure:**
  - (i) Draw two data sets from the same source.
  - (ii) Cluster both data sets.
  - (iii) Compute agreement

Stability $:=$ expected agreement of the solutions.

- **In practical applications:** only one data set available.
  - (i) Estimate expected agreement by resampling.
  - (ii) Cluster entire data set with optimal k.

Measuring disagreement

Two labelings on one data set:

disagreement $:=$ Fraction of differently labeled objects.

Measuring disagreement

Two labelings on one data set:

disagreement $:=$ Fraction of differently labeled objects.

3 problems:

1. Clustering solutions are labelings of disjoint sets.

Measuring disagreement

Two labelings on one data set:

disagreement $:=$ Fraction of differently labeled objects.

3 problems:

1. Clustering solutions are labelings of disjoint sets.
2. Labeling is unique only up to permutation $\pi \in \mathfrak{S}_k$.

Measuring disagreement

Two labelings on one data set:

disagreement $:=$ Fraction of differently labeled objects.

3 problems:

1. Clustering solutions are labelings of disjoint sets.
2. Labeling is unique only up to permutation $\pi \in \mathfrak{S}_k$.
3. Fraction of differently labeled points is sensitive to model complexity:

   50% @ $k = 2 \rightarrow$ totally random,

   50% @ $k = 10 \rightarrow$ often acceptable.

Stability Measure: Labelings on disjoint sets

Extend solution from set A to B

Stability Measure: Labelings on disjoint sets

Extend solution from set A to B by

(i) training a predictor on A

Stability Measure: Labelings on disjoint sets

Extend solution from set A to B by

(i) training a predictor on A

(ii) predicting labels on B

Stability Measure: Labelings on disjoint sets

Extend solution from set A to B by

(i) training a predictor on A

(ii) predicting labels on B

(iii) compare clustering solutions on B

Stability Measure: Labelings on disjoint sets

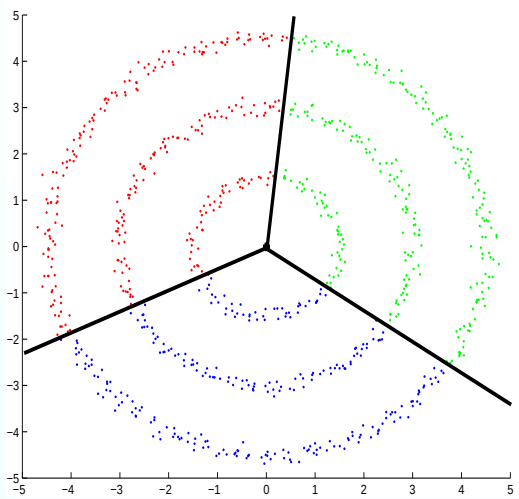
Extend solution from set A to B by

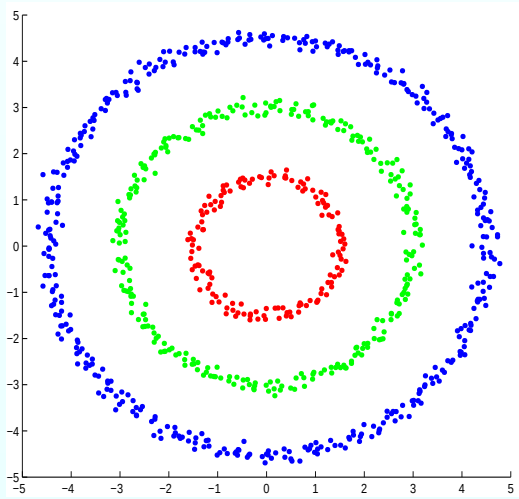
(i) training a predictor on A

(ii) predicting labels on B

(iii) compare clustering solutions on B

Choose predictor according to meta-principle.



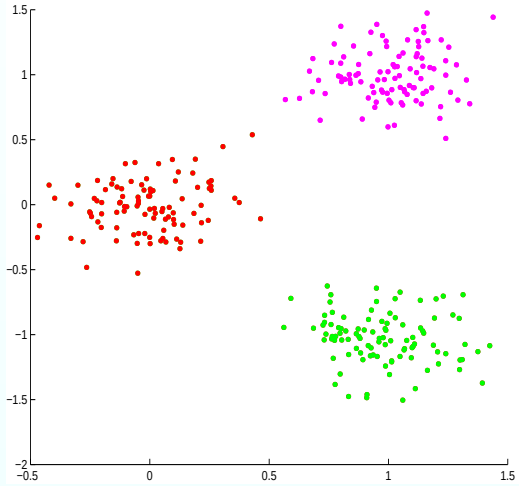


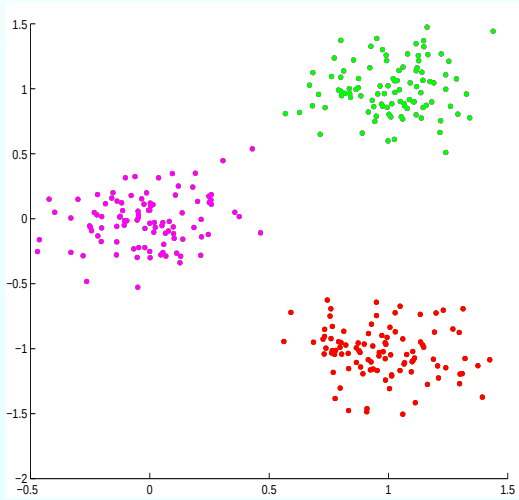
Stability Measure: Breaking Permutation Symmetry

- Labeling unique only up to $\pi \in \mathfrak{S}_k$.

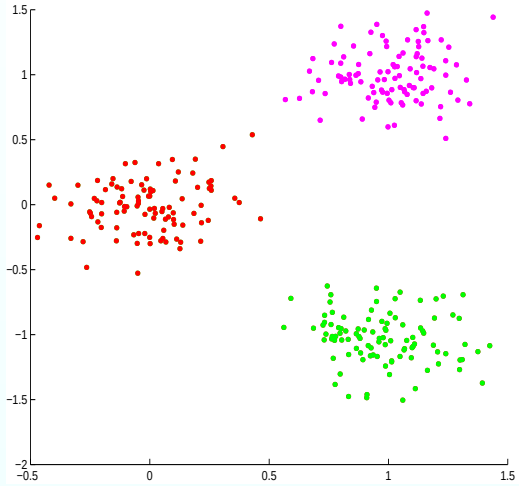
Stability Measure: Breaking Permutation Symmetry

- Labeling unique only up to $\pi \in \mathfrak{S}_k$.
- **Solution:** Stability index $S :=$ expected minimal disagreement over all $\pi \in \mathfrak{S}_k$.

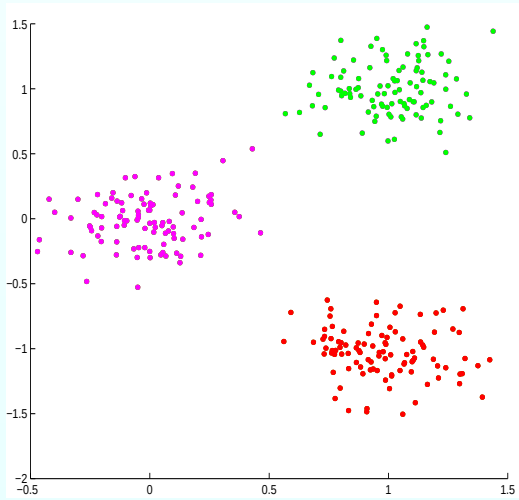




Stability Measure: Breaking Permutation Symmetry



- Labeling unique only up to $\pi \in \mathfrak{S}_k$.
- **Solution:** Stability index $S :=$ expected minimal disagreement over all $\pi \in \mathfrak{S}_k$.
- **Hungarian method** $O(k^3)$.



Stability Measure: Different Values of k

- Stability costs are scale-sensitive to k.

Stability Measure: Different Values of k

- Stability costs are scale-sensitive to k.
- **Solution:** Normalize by maximal stability costs:

  $$S_k(\alpha) \leq 1 - 1/k$$

  For the random predictor ϱ it holds:

  $$S_k(\varrho) \to 1 - 1/k \text{ as } n \to \infty.$$

Stability Measure: Different Values of k

- Stability costs are scale-sensitive to k.
- **Solution:** Normalize by maximal stability costs:

  $$S_k(\alpha) \leq 1 - 1/k$$

  For the random predictor ϱ it holds:

  $$S_k(\varrho) \to 1 - 1/k \text{ as } n \to \infty.$$

$$\rightsquigarrow \boxed{\text{Normalize } S \mapsto \frac{S_k(\alpha)}{S_k(\varrho)}.}$$

The Final Stability Measure

$$\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{\alpha(\mathbf{X})_i \neq \qquad g(X_i; \mathbf{X}', \alpha(\mathbf{X}'))\}$$

- Disagreement rate

The Final Stability Measure

$$\min_{\pi \in \mathfrak{S}_k} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{\alpha(\mathbf{X})_i \neq \pi \circ g(X_i; \mathbf{X}', \alpha(\mathbf{X}'))\}$$

- Disagreement rate
- Permutation symmetry breaking

The Final Stability Measure

$$\mathbb{E}_{\mathbf{X},\mathbf{X}'}\left(\min_{\pi \in \mathfrak{S}_k} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{\alpha(\mathbf{X})_i \neq \pi \circ g(X_i; \mathbf{X}', \alpha(\mathbf{X}'))\} \right)$$

- Disagreement rate
- Permutation symmetry breaking
- Expectation w.r.t. two samples from same source

The Final Stability Measure

$$\frac{1}{S(\varrho)}\mathbb{E}_{\mathbf{X},\mathbf{X}'}\left(\min_{\pi\in\mathfrak{S}_k}\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\{\alpha(\mathbf{X})_i \neq \pi\circ g(X_i;\mathbf{X}',\alpha(\mathbf{X}'))\}\right)$$

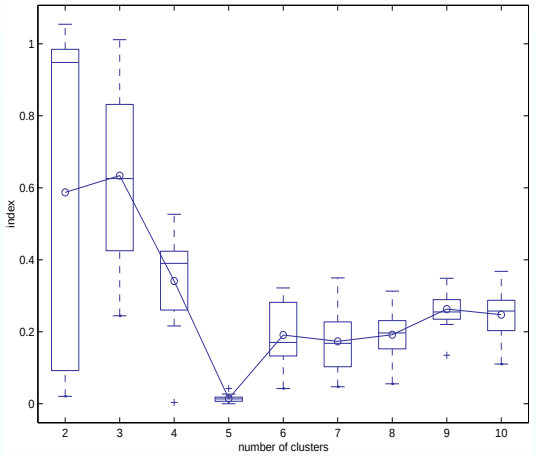
- Disagreement rate
- Permutation symmetry breaking
- Expectation w.r.t. two samples from same source
- Normalize by $S(\varrho)$.

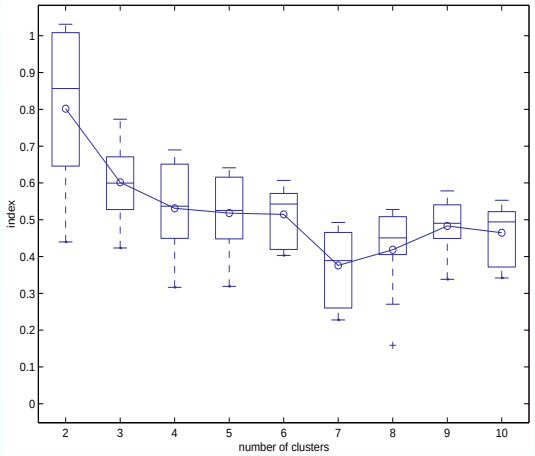
The Final Stability Measure

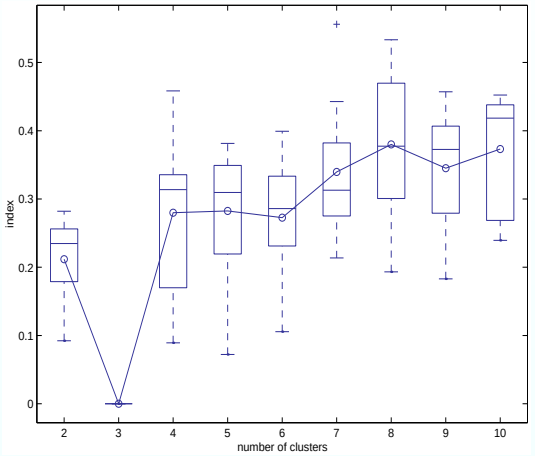
$$\frac{1}{S(\varrho)}\mathbb{E}_{\mathbf{X},\mathbf{X}'}\left(\min_{\pi\in\mathfrak{S}_k}\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\{\alpha(\mathbf{X})_i \neq \pi\circ g(X_i;\mathbf{X}',\alpha(\mathbf{X}'))\}\right)$$

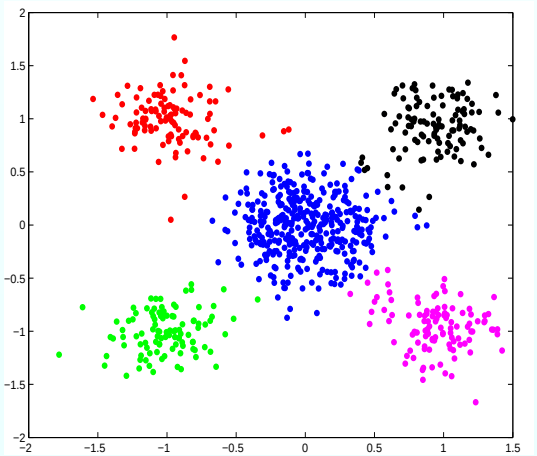
- Disagreement rate
- Permutation symmetry breaking
- Expectation w.r.t. two samples from same source
- Normalize by $S(\varrho)$.
- Estimate $\mathbb{E}_{\mathbf{X},\mathbf{X}'}$ by resampling

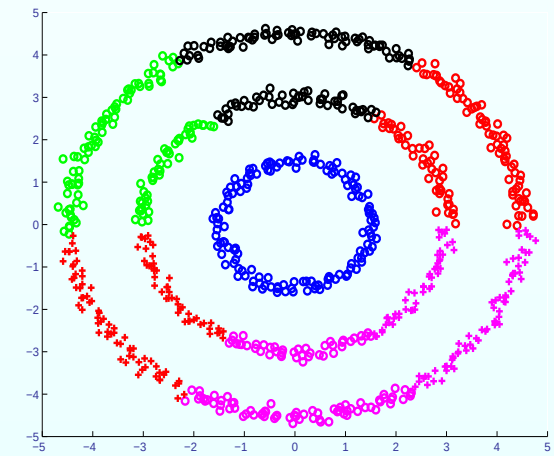
Results on Toy Data

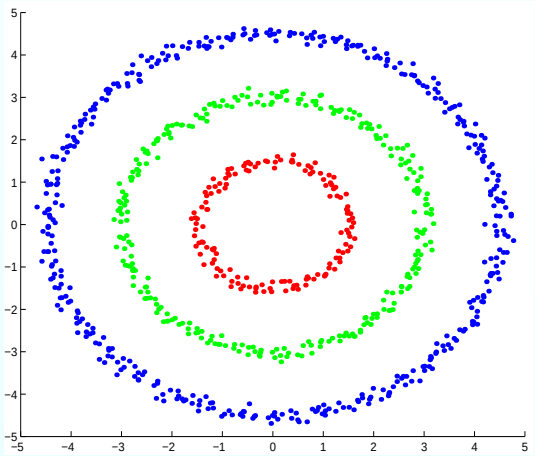












Biological Applications

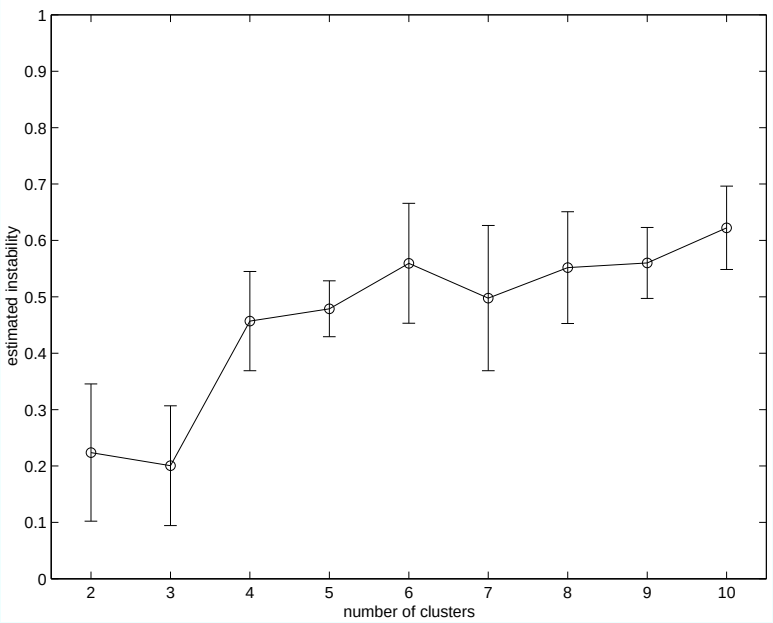
- Tumor Class Discovery from gene expression data:
  Identify different types of Leukemia
  72 Tumor-Samples, 100 selected genes,
  “ground truth”: 2 or 3 classes

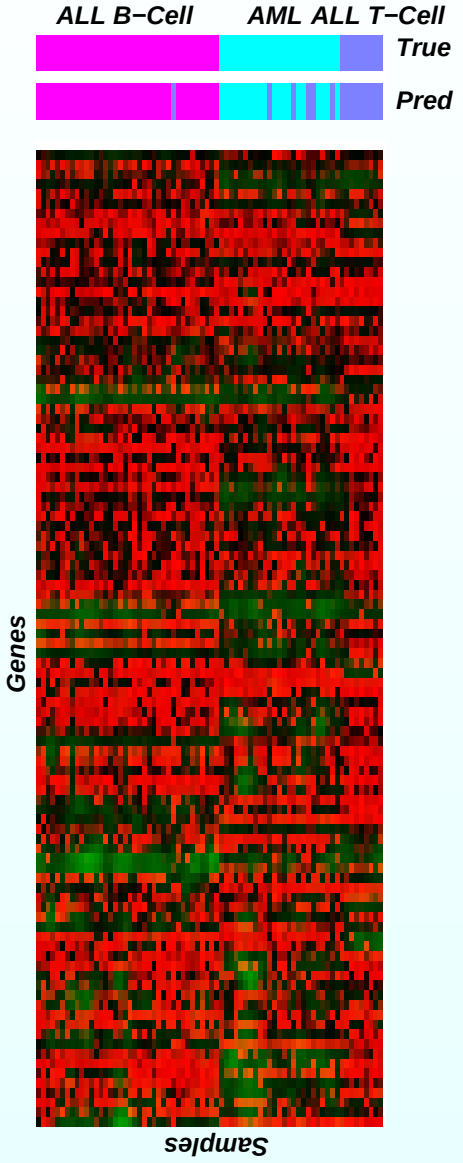
Biological Applications

- Tumor Class Discovery from gene expression data:
  Identify different types of Leukemia
  72 Tumor-Samples, 100 selected genes,
  “ground truth”: 2 or 3 classes

- Clustering of Proteins:
  Find groups of similar proteins
  Pairwise Data
  1200 Globins or Globin-like proteins
  “ground truth”: 5 classes

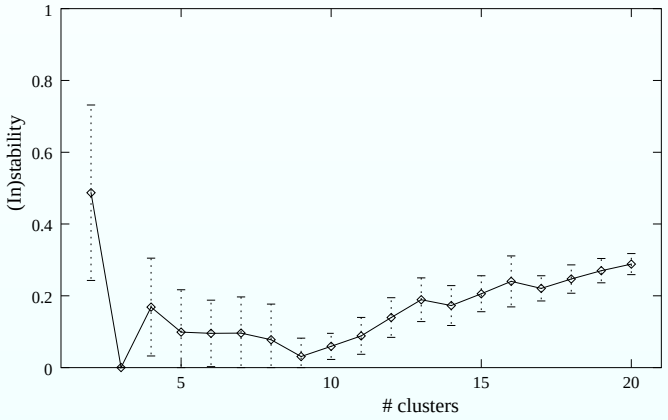
Class Discovery

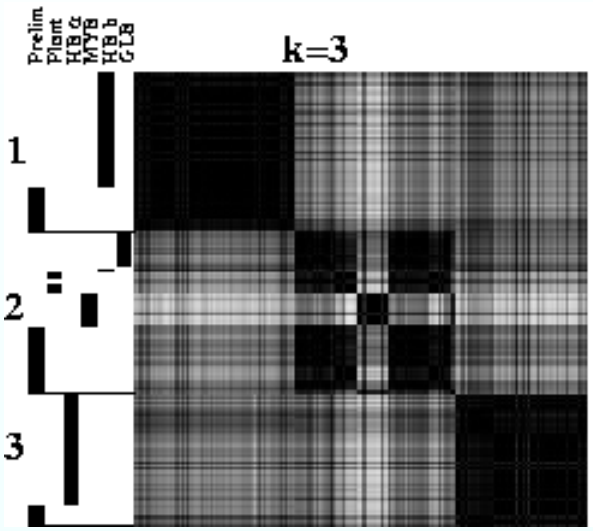


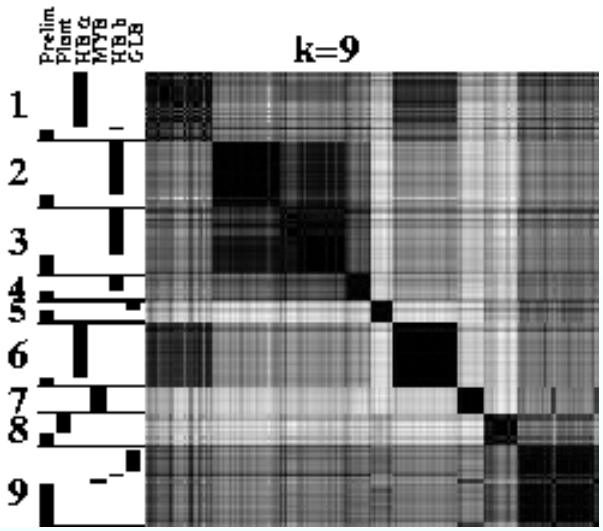


- Two candidates: $k = 3$ and $k = 2$
- Approx. 91 % of the known classes found.

Clustering of Globins







- Two candidates: $k = 3$ und $k = 9$
- Separation of hemoglobin-α and hemoglobin-β from the remaining globins.

Biologically plausible clustering!

Stability: Summary

- **Stability Principle:** Solutions for two data sets from the same source should be similar.
- **No additional assumptions** about the structure of solutions.
- **Good performance** on experimental data sets.

Empty Slide for Notes