OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

INF FAKULTÄT FÜR INFORMATIK

Fuzzy Systems

Fuzzy Data Analysis

Prof. Dr. Rudolf Kruse **Christoph Doell**

{kruse,doell}@ovgu.de

Otto-von-Guericke University of Magdeburg

Faculty of Computer Science

Institute for Intelligent Cooperating Systems

Two Interpretations of Fuzzy Data Analysis

FUZZY Data Analysis $\widehat{=}$ Fuzzy Techniques for the analysis of (crisp) data

in our course: Fuzzy Clustering

FUZZY DATA Analysis $\widehat{=}$ Analysis of Data in Form of Fuzzy Sets

in our course: Random Sets, Fuzzy Random Variables

Fuzzy Clustering

Clustering

This is an **unsupervised** learning task.

The goal is to divide the dataset such that both constraints hold:

- objects belonging to same cluster: as similar as possible
- objects belonging to different clusters: as dissimilar as possible

The **similarity** is measured in terms of a **distance** function.

The smaller the distance, the more similar two data tuples.

Definition

$d : \mathbb{R}^p \times \mathbb{R}^p \to [0, \infty)$ is a distance function if $\forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^p$:

(i) $d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}$ (identity),

(ii) $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ (symmetry),

(iii) $d(\boldsymbol{x}, \boldsymbol{z}) \leq d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z})$ (triangle inequality).

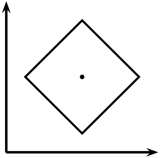
Illustration of Distance Functions

Minkowski family

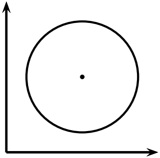
$$d_k(\mathbf{x}, \mathbf{y}) = \left(\sum_{d=1}^{p} |x_d - y_d|^k \right)^{\frac{1}{k}}$$

Well-known special cases from this family are

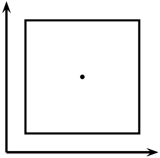
$k = 1$: Manhattan or city block distance,
$k = 2$: Euclidean distance,
$k \to \infty$: maximum distance, *i.e.* $d_\infty(\mathbf{x}, \mathbf{y}) = \max_{d=1}^{p} |x_d - y_d|$.



$k = 1$



$k = 2$



$k \to \infty$

Partitioning Algorithms

Here, we focus only on partitioning algorithms,

- *i.e.* given $c \in \mathbb{N}$, find the best partition of data into c groups.
- That is different from hierarchical techniques,
  i.e. organize data in a nested sequence of groups.

Usually the number of (true) clusters is unknown.

However, partitioning methods must specify a c-value.

Prototype-based Clustering

Another restriction of prototype-based clustering algorithms:

- Clusters are represented by cluster prototypes $C_i, i = 1, \ldots, c$.

Prototypes capture the structure (distribution) of data in each cluster.

The set of prototypes is $C = \{C_1, \ldots, C_c\}$.

Every prototype C_i is an *n*-tuple with

- the cluster center $\mathbf{c}_i$ and
- additional parameters about its size and shape.

Prototypes are constructed by clustering algorithms.

(hard) c-Means Clustering

1) Choose a number c of clusters to be found (user input).

2) Initialize the cluster centers randomly
(for instance, by randomly selecting c data points).

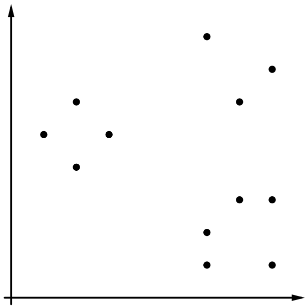
3) **Data point assignment**:
Assign each data point to the cluster center that is closest to it
(i.e. closer than any other cluster center).

4) **Cluster center update**:
Compute new cluster centers as mean of the assigned data points.
(Intuitively: center of gravity)

5) Repeat steps 3 and 4 until cluster centers remain constant.

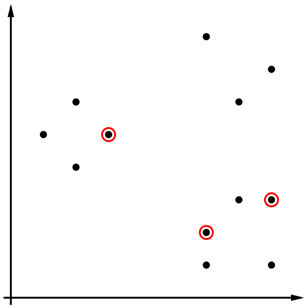
It can be shown that this scheme must converge

c-Means Clustering: Example



Data set to cluster.

Choose $c = 3$ clusters.
(From visual inspection, can be difficult to determine in general.)

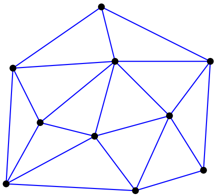


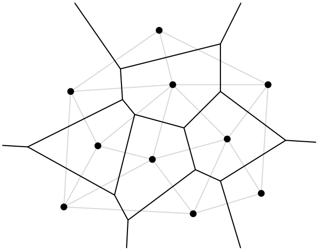
Initial position of cluster centers.

Randomly selected data points.
(Alternative methods include e.g. latin hypercube sampling)

Delaunay Triangulations and Voronoi Diagrams

Dots represent cluster centers (quantization vectors).





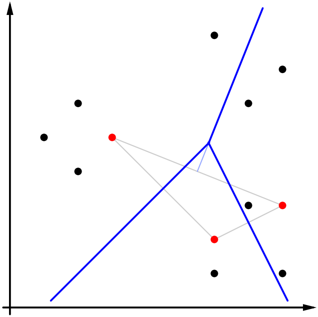
Delaunay Triangulation
The circle through the corners of a triangle does not contain another point.

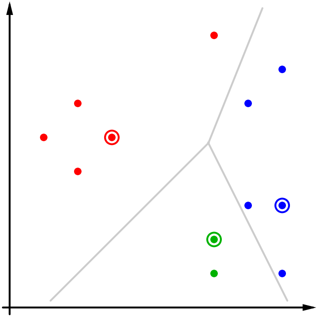
Voronoi Diagram
Midperpendiculars of the Delaunay triangulation: boundaries of the regions of points that are closest to the enclosed cluster center (Voronoi cells).

Delaunay Triangulations and Voronoi Diagrams

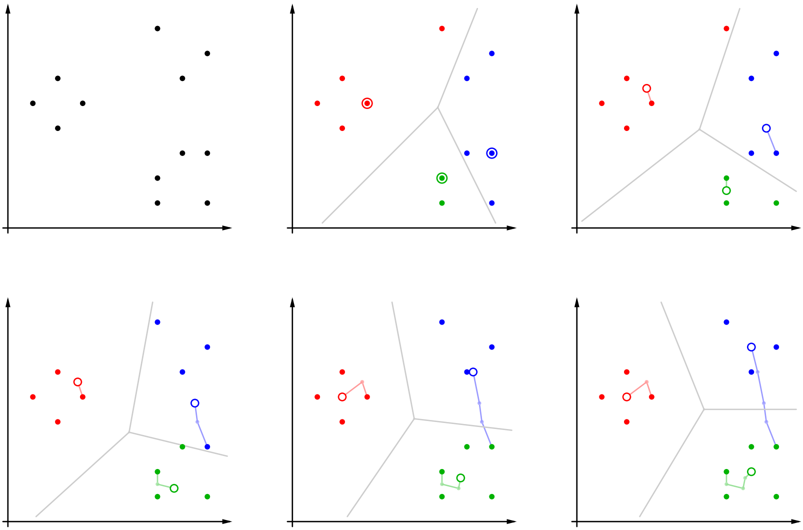
Delaunay Triangulation: simple triangle (shown in grey on the left)

Voronoi Diagram: midperpendiculars of the triangle's edges (shown in blue on the left, in grey on the right)





c-Means Clustering: Example



c-Means Clustering: Local Minima

In the example Clustering was successful and formed intuitive clusters

Convergence achieved after only 5 steps.
(This is typical: convergence is usually very fast.)

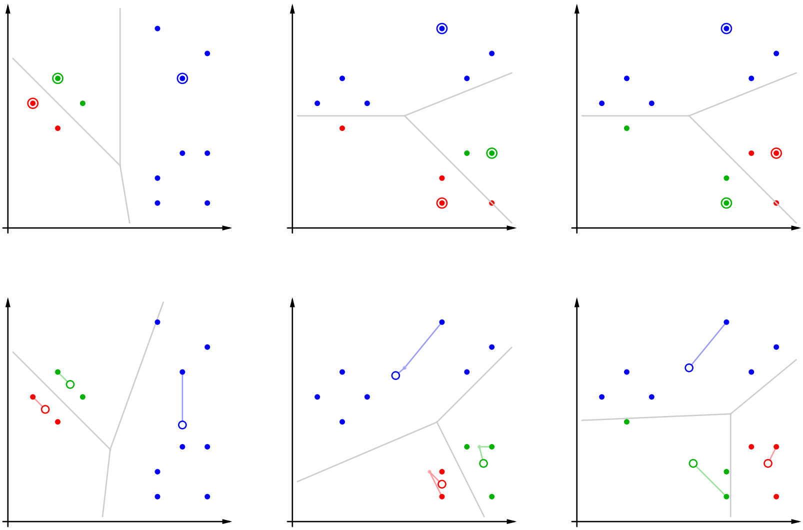
Nevertheless: Result is **sensitive to the initial positions** of cluster centers.

With a bad initialization clustering may fail
(the alternating update process gets stuck in a local minimum).

Fuzzy c-means clustering and the estimation of a mixture of Gaussians are much more robust (to be discussed later).

Research issue: How to determine the number of clusters automatically?
(Some approaches exists, but none of them is too successful.)

c-Means Clustering: Local Minima



Center Vectors and Objective Functions

Consider the simplest cluster prototypes, *i.e.* center vectors $C_i = (\boldsymbol{c}_i)$.

The distance d is based on an inner product, *e.g.* the Euclidean distance.

All algorithms are based on an objective functions J which

- quantifies the goodness of the cluster models and
- must be minimized to obtain optimal clusters.

Cluster algorithms determine the best decomposition by minimizing J.

Hard c-means

Each point x_j in the dataset $X = \{x_1, \ldots, x_n\}$, $X \subseteq \mathbb{R}^p$ is assigned to exactly 1 cluster.

That is, each cluster $\Gamma_i \subset X$.

The set of clusters $\Gamma = \{\Gamma_1, \ldots, \Gamma_c\}$ must be an exhaustive partition of X into c non-empty and pairwise disjoint subsets $\Gamma_i, 1 < c < n$.

The data partition is optimal when the sum of squared distances between cluster centers and data points assigned to them is minimal.

Clusters should be as homogeneous as possible.

Hard c-means

The objective function of the hard c-means is

$$J_h(X, U_h, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} d_{ij}^2$$

where d_{ij} is the distance between $\boldsymbol{c}_i$ and $\boldsymbol{x}_j$, *i.e.* $d_{ij} = d(\boldsymbol{c}_i, \boldsymbol{x}_j)$, and $U = (u_{ij})_{c \times n}$ is the the **partition matrix** with

$$u_{ij} = \begin{cases} 1, & \text{if } \boldsymbol{x}_j \in \Gamma_i \\ 0, & \text{otherwise.} \end{cases}$$

Each data point is assigned exactly to one cluster and every cluster must contain at least one data point:

$$\forall j \in \{1, \ldots, n\} : \quad \sum_{i=1}^{c} u_{ij} = 1 \quad \text{and} \quad \forall i \in \{1, \ldots, c\} : \quad \sum_{j=1}^{n} u_{ij} > 0.$$

Alternating Optimization Scheme

J_h depends on c, and U on the data points to the clusters.

Finding the parameters that minimize J_h is NP-hard.

Hard c-means minimizes J_h by **alternating optimization (AO)**:

- The parameters to optimize are split into 2 groups.
- One group is optimized holding other one fixed (and vice versa).
- This is an iterative update scheme: repeated until convergence.

There is no guarantee that the global optimum will be reached.

AO may get stuck in a local minimum.

AO Scheme for Hard c-means

(i) Chose an initial $\boldsymbol{c}_i$, *e.g.* randomly picking c data points $\in X$.

(ii) Hold C fixed and determine U that minimize J_h:
Each data point is assigned to its closest cluster center:

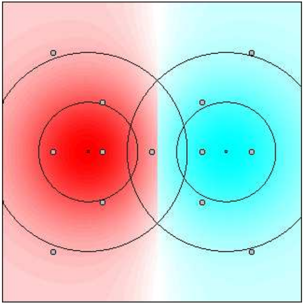
$$u_{ij} = \begin{cases} 1, & \text{if } i = \arg\min_{k=1}^{c} d_{kj} \\ 0, & \text{otherwise.} \end{cases}$$

(iii) Hold U fixed, update $\boldsymbol{c}_i$ as mean of all x_j assigned to them:
The mean minimizes the sum of square distances in J_h:

$$\boldsymbol{c}_i = \frac{\sum_{j=1}^{n} u_{ij} \boldsymbol{x}_j}{\sum_{j=1}^{n} u_{ij}}.$$

(iv) Repeat steps (ii)+(iii) until no changes in C or U are observable.

Example



Given a symmetric dataset with two clusters.

Hard c-means assigns a crisp label to the data point in the middle.

Is that very intuitive?

Discussion: Hard c-means

It tends to get stuck in a local minimum.

Thus necessary are several runs with different initializations

There are sophisticated initialization methods available, *e.g.* Latin hypercube sampling.

The best result of many clusterings is chosen based on J_h.

Crisp memberships $\{0, 1\}$ prohibit ambiguous assignments.

For adly delineated or overlapping clusters, one should relax the requirement $u_{ij} \in \{0, 1\}$.

Fuzzy Clustering

Fuzzy Clustering

It allows gradual memberships of data points to clusters in $[0, 1]$.

It offers the flexibility to express whether a data point belongs to more than 1 cluster.

Thus, membership degrees

- offer a finer degree of detail of the data model,
- express how ambiguously/definitely $\boldsymbol{x}_j$ should belong to Γ_i.

The solution spaces equal fuzzy partitions of $X = \{x_1, \ldots, x_n\}$.

Fuzzy Clustering

The clusters Γ_i have been classical subsets so far.

Now, they are represented by fuzzy sets μ_{Γ_i} of X.

Thus, u_{ij} is a membership degree of $\boldsymbol{x}_j$ to Γ_i such that $u_{ij} = \mu_{\Gamma_i}(\boldsymbol{x}_j) \in [0, 1]$.

The fuzzy label vector $\boldsymbol{u} = (u_{1j}, \ldots, u_{cj})^T$ is linked to each $\boldsymbol{x}_j$.

$U = (u_{ij}) = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n)$ is called **fuzzy partition matrix**.

There are 2 types of fuzzy cluster partitions:

- **probabilistic and possibilistic**
- They differ in constraints they place on the membership degrees.

Probabilistic Cluster Partition

Definition

Let $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be the set of given examples and let c be the number of clusters $(1 < c < n)$ represented by the fuzzy sets $\mu_{\Gamma_i}, (i = 1, \ldots, c)$. Then we call $U_f = (u_{ij}) = (\mu_{\Gamma_i}(\mathbf{x}_j))$ a *probabilistic cluster partition* of X if

$$\sum_{j=1}^{n} u_{ij} > 0, \quad \forall i \in \{1, \ldots, c\}, \quad \text{and}$$

$$\sum_{i=1}^{n} u_{ij} = 1, \quad \forall j \in \{1, \ldots, n\}$$

hold. The $u_{ij} \in [0, 1]$ are interpreted as the membership degree of datum $\mathbf{x}_j$ to cluster Γ_i relative to all other clusters.

Probabilistic Cluster Partition

The 1st constraint guarantees that there aren't any empty clusters.

- This is a requirement in classical cluster analysis.
- Thus, no cluster, represented as (classical) subset of X, is empty.

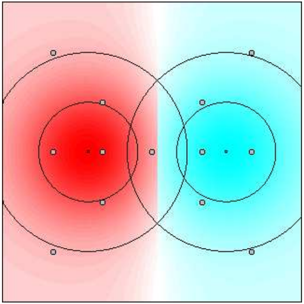
The 2nd condition says that sum of membership degrees must be 1 for each $\mathbf{x}_j$.

- Each datum gets the same weight compared to other data points.
- So, all data are (equally) included into the cluster partition.
- This relates to classical clustering where partitions are exhaustive.

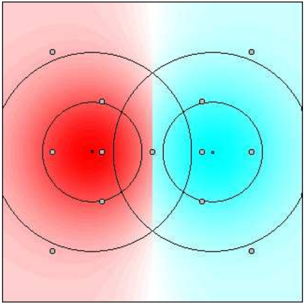
The consequence of both constraints are as follows:

- No cluster can contain the full membership of all data points.
- The membership degrees *resemble* probabilities of being member of corresponding cluster.

Example



hard c-means



fuzzy c-means

There is no arbitrary assignment for the equidistant data point in middle anymore.

In the fuzzy partition it is associated with the membership vector $(0.5, 0.5)^T$ (which expresses the ambiguity of the assignment).

Objective Function

Minimize the objective function

$$J_f(X, U_h, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2$$

subject to

$$\sum_{i=1}^{c} u_{ij} = 1, \quad \forall j \in \{1, \ldots, n\}$$

and

$$\sum_{j=1}^{n} u_{ij} > 0, \quad \forall i \in \{1, \ldots, c\}$$

where parameter $m \in \mathbb{R}$ with $m > 1$ is called the **fuzzifier** and $d_{ij} = d(\boldsymbol{c}_i, \boldsymbol{x}_j)$.

Fuzzifier

The actual value of m determines the "fuzziness" of the classification.

For $m = 1$ (*i.e.* $J_h = J_f$), the assignments remain hard.

Fuzzifiers of $m > 1$ lead to fuzzy memberships

Clusters become softer/harder with a higher/lower value of m.

Usually $m = 2$.

Reminder: Function Optimization

Task: find $\boldsymbol{x} = (x_1, \ldots, x_m)$ such that $f(\boldsymbol{x}) = f(x_1, \ldots, x_m)$ is optimal.

Often a feasible approach is to

- define the necessary condition for (local) optimum (max./min.): partial derivatives *w.r.t.* parameters vanish.
- Thus we (try to) solve an equation system coming from setting all partial derivatives *w.r.t.* the parameters equal to zero.

Example task: minimize $f(x, y) = x^2 + y^2 + xy - 4x - 5y$

Solution procedure:

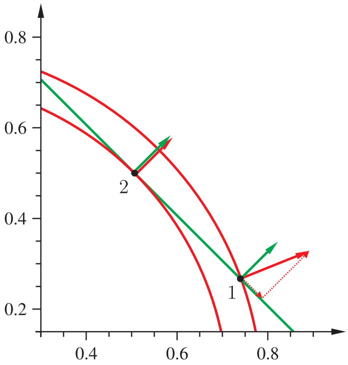
- Take the partial derivatives of f and set them to zero:

$$\frac{\partial f}{\partial x} = 2x + y - 4 = 0, \qquad \frac{\partial f}{\partial y} = 2y + x - 5 = 0.$$

- Solve the resulting (here linear) equation system: $x = 1,\ y = 2.$

Example

Task: Minimize $f(x_1, x_2) = x_1^2 + x_2^2$ subject to $g : x_1 + x_2 = 1$.



Crossing a contour line: Point 1 cannot be a constrained minimum because ∇f has a non-zero component in the constrained space. Walking in opposite direction to this component can further decrease f.

Touching a contour line: Point 2 is a constrained minimum: both gradients are parallel, hence there is no component of ∇f in the constrained space that might lead us to a lower value of f.

Function Optimization: Lagrange Theory

We can use the **Method of Lagrange Multipliers:**

Given: $f(\boldsymbol{x})$ to be optimized, k equality constraints $C_j(\boldsymbol{x}) = 0,\ 1 \leq j \leq k$

procedure:

1. Construct the so-called **Lagrange function** by incorporating $C_i,\ i = 1, \ldots, k$, with (unknown) **Lagrange multipliers** λ_i:

$$L(\boldsymbol{x}, \lambda_1, \ldots, \lambda_k) = f(\boldsymbol{x}) + \sum_{i=1}^{k} \lambda_i C_i(\boldsymbol{x}).$$

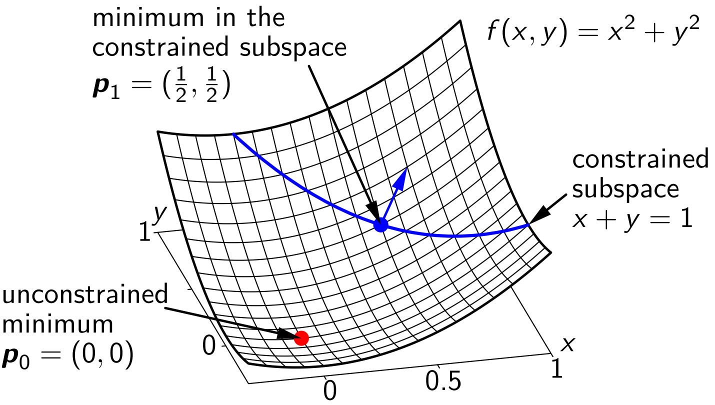
2. Set the partial derivatives of Lagrange function equal to zero:

$$\frac{\partial L}{\partial x_1} = 0, \quad \ldots, \quad \frac{\partial L}{\partial x_m} = 0, \qquad \frac{\partial L}{\partial \lambda_1} = 0, \quad \ldots, \quad \frac{\partial L}{\partial \lambda_k} = 0.$$

3. (Try to) solve the resulting equation system.

Lagrange Theory: Revisited Example 1

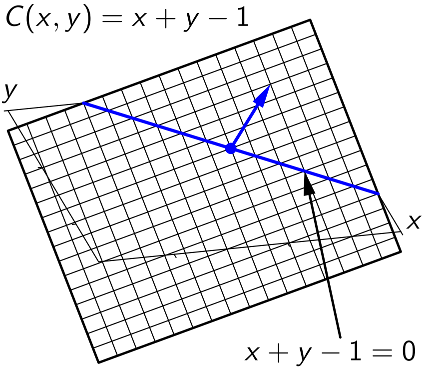
Example task: Minimize $f(x, y) = x^2 + y^2$ subject to $x + y = 1$.

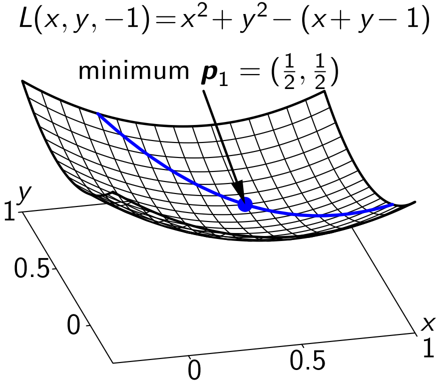


The unconstrained minimum is not in the constrained subspace. At the minimum in the constrained subspace the gradient does not vanish.

Lagrange Theory: Revisited Example 1

Example task: Minimize $f(x, y) = x^2 + y^2$ subject to $x + y = 1$.





The gradient of the constraint is perpendicular to the constrained subspace. The (unconstrained) minimum of the $L(x, y, \lambda)$ is the minimum of $f(x, y)$ in the constrained subspace.

Lagrange Theory: Example 2

Example task: find the side lengths x, y, z of a box with maximum volume for a given area S of the surface.

Formally: max. $f(x, y, z) = xyz$ such that $2xy + 2xz + 2yz = S$

Solution procedure:

- The constraint is $C(x, y, z) = 2xy + 2xz + 2yz - S = 0$.
- The Lagrange function is

$$L(x, y, z, \lambda) = xyz + \lambda(2xy + 2xz + 2yz - S).$$

- Taking the partial derivatives yields (in addition to constraint):

$$\frac{\partial L}{\partial x} = yz+2\lambda(y+z) = 0, \quad \frac{\partial L}{\partial y} = xz+2\lambda(x+z) = 0, \quad \frac{\partial L}{\partial y} = xy+2\lambda(x+y) = 0.$$

- The solution is $\lambda = -\frac{1}{4}\sqrt{\frac{S}{6}}$, $\quad x = y = z = \sqrt{\frac{S}{6}}$ $\quad$ (*i.e.* box is a cube).

Optimizing the Membership Degrees

J_f is alternately optimized, *i.e.*

- optimize U for a fixed cluster parameters $U_\tau = j_U(C_{\tau-1})$,
- optimize C for a fixed membership degrees $C_\tau = j_C(U_\tau)$.

The update formulas are obtained by setting the derivative J_f *w.r.t.* parameters U, C to zero. .

The resulting equations form the fuzzy c-means (FCM) algorithm [Bezdek, 1981]:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}^2}{d_{kj}^2} \right)^{\frac{1}{m-1}}} = \frac{d_{ij}^{-\frac{2}{m-1}}}{\sum_{k=1}^{c} d_{kj}^{-\frac{2}{m-1}}}.$$

That is independent of any distance measure.

First Step: Fix the cluster parameters

Introduce the Lagrange multipliers λ_j, $0 \leq j \leq n$, to incorporate the constraints $\forall j; 1 \leq j \leq n : \sum_{i=1}^{c} u_{ij} = 1$.

Then, the Lagrange function (to be minimized) is

$$L(X, U_f, C, \Lambda) = \underbrace{\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d_{ij}^2}_{=J(X, U_f, C)} + \sum_{j=1}^{n} \lambda_j \left(1 - \sum_{i=1}^{c} u_{ij}\right).$$

The necessary condition for a minimum is that the partial derivatives of the Lagrange function *w.r.t.* membership degrees vanish, *i.e.*

$$\frac{\partial}{\partial u_{kl}} L(X, U_f, C, \Lambda) = m u_{kl}^{m-1} d_{kl}^2 - \lambda_l \stackrel{!}{=} 0$$

which leads to

$$\forall i; 1 \leq i \leq c : \forall j; 1 \leq j \leq n : \qquad u_{ij} = \left(\frac{\lambda_j}{m d_{ij}^2}\right)^{\frac{1}{m-1}}.$$

Optimizing the Membership Degrees

Summing these equations over clusters leads

$$1 = \sum_{i=1}^{c} u_{ij} = \sum_{i=1}^{c} \left(\frac{\lambda_j}{m d_{ij}^2} \right)^{\frac{1}{m-1}} .$$

Thus the λ_j, $1 \leq j \leq n$ are

$$\lambda_j = \left(\sum_{i=1}^{c} \left(m d_{ij}^2 \right)^{\frac{1}{1-m}} \right)^{1-m} .$$

Inserting this into the equation for the membership degrees yields

$$\forall i; 1 \leq i \leq c : \forall j; 1 \leq j \leq n : \qquad u_{ij} = \frac{d_{ij}^{\frac{2}{1-m}}}{\sum_{k=1}^{c} d_{kj}^{\frac{2}{1-m}}}.$$

This update formula is independent of any distance measure.

Optimizing the Cluster Prototypes

The update formula j_C depend on both

- cluster parameters (location, shape, size) and
- the distance measure.

Thus the general update formula cannot be given.

For the basic fuzzy c-means model,

- the cluster centers serve as prototypes, and
- the distance measure is an induced metric by the inner product.

Thus the **second step** (*i.e.* the derivations of J_f *w.r.t.* the centers) yields [Bezdek, 1981]

$$\mathbf{c}_i = \frac{\sum_{j=1}^{n} u_{ij}^m \mathbf{x}_j}{\sum_{j=1}^{n} u_{ij}^m}.$$

Discussion: Fuzzy c-means

It is initialized with randomly placed cluster centers.

The updating in AO scheme stops if

- the number of iterations exceeds some predefined limit
- or the changes in the prototypes $\leq$ some termination accuracy.

FCM is stable and robust.

Compared to hard c-means, it's

- quite insensitive to initialization and
- not likely to get stuck in local minimum.

FCM converges in a saddle point or minimum (but not in a maximum) [Bezdek, 1981].

Cluster Validity

Problems with Fuzzy Clustering

What is optimal number of clusters c?

Shape and location of cluster prototypes: not known a priori $\Rightarrow$ initial guesses needed

Must be handled: different data characteristics, *e.g.* variabilities in shape, density and number of points in different clusters

Cluster Validity for Fuzzy Clustering

Idea: each data point has c memberships

Desirable: summarize information by single criterion indicating how well data point is classified by clustering

Cluster validity: average of any criteria over entire data set

"good" clusters are actually not very fuzzy!

Criteria for definition of "optimal partition" based on:

- clear separation between resulting clusters
- minimal volume of clusters
- maximal number of points concentrated close to cluster centroid

Judgment of Classification by Validity Measures

Validity measures can be based on several criteria, *e.g.*

membership degrees should be $\approx 0/1$, *e.g.* **partition coefficient**

$$\mathrm{PC} = \frac{1}{n} \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^2$$

Compactness of clusters, *e.g.* **average partition density**

$$\mathrm{APD} = \frac{1}{c} \sum_{i=1}^{c} \frac{\sum_{j \in Y_i} u_{ij}}{\sqrt{|\Sigma_i|}}$$

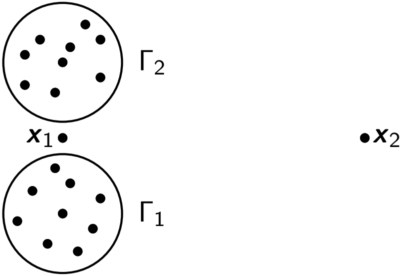
where $Y_i = \{j \in \mathbb{N},\ j \le n \mid (\mathbf{x}_j - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) < 1\}$

especially for FCM: **partition entropy**

$$\mathrm{PE} = -\sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} \log u_{ij}$$

Extensions of Fuzzy Clustering

Problems with Probabilistic c-means



$\boldsymbol{x}_1$ has the same distance to Γ_1 and $\Gamma_2 \Rightarrow \mu_{\Gamma_1}(\boldsymbol{x}_1) = \mu_{\Gamma_2}(\boldsymbol{x}_1) = 0.5$.

The same degrees of membership are assigned to $\boldsymbol{x}_2$.

This problem is due to the normalization.

A better reading of memberships is "If $\boldsymbol{x}_j$ must be assigned to a cluster, then with probability u_{ij} to Γ_i".

Problems with Probabilistic c-means

The normalization of memberships is a problem for noise and outliers.

A fixed data point weight causes a high membership of noisy data, although there is a large distance from the bulk of the data.

This has a bad effect on the clustering result.

Dropping the normalization constraint

$$\sum_{i=1}^{c} u_{ij} = 1, \quad \forall j \in \{1, \ldots, n\},$$

we obtain more intuitive membership assignments.

Possibilistic Cluster Partition

Definition

Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be the set of given examples and let c be the number of clusters $(1 < c < n)$ represented by the fuzzy sets $\mu_{\Gamma_i}, (i = 1, \ldots, c)$. Then we call $U_p = (u_{ij}) = (\mu_{\Gamma_i}(\boldsymbol{x}_j))$ a *possibilistic cluster partition* of X if

$$\sum_{j=1}^{n} u_{ij} > 0, \quad \forall i \in \{1, \ldots, c\}$$

holds. The $u_{ij} \in [0, 1]$ are interpreted as degree of representativity or typicality of the datum $\boldsymbol{x}_j$ to cluster Γ_i.

now, u_{ij} for $\boldsymbol{x}_j$ resemble possibility of being member of corresponding cluster

Possibilistic Fuzzy Clustering

J_f is not appropriate for possibilistic fuzzy clustering.

Dropping the normalization constraint leads to a minimum for all $u_{ij} = 0$.

Thus is, data points are not assigned to any Γ_i. Thus all Γ_i are empty.

Hence a penalty term is introduced which forces all u_{ij} away from zero.

The objective function J_f is modified to

$$J_p(X, U_p, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} (1 - u_{ij})^m$$

where $\eta_i > 0 (1 \le i \le c)$.

The values η_i balance the contrary objectives expressed in J_p.

Optimizing the Membership Degrees

The update formula for membership degrees is

$$u_{ij} = \frac{1}{1 + \left(\frac{d_{ij}^2}{\eta_i}\right)^{\frac{1}{m-1}}}.$$

The membership of $\boldsymbol{x}_j$ to cluster i depends only on d_{ij} to this cluster.

A small distance corresponds to a high degree of membership.

Larger distances result in low membership degrees.

So, u_{ij}'s share a typicality interpretation.

Interpretation of η_i

The update equation helps to explain the parameters η_i.

Consider $m = 2$ and substitute η_i for d_{ij}^2 yields $u_{ij} = 0.5$.

Thus η_i determines the distance to Γ_i at which u_{ij} should be 0.5.

η_i can have a different geometrical interpretation:

- the hyperspherical clusters (*e.g.* PCM), thus $\sqrt{\eta_i}$ is the mean diameter.

Estimating η_i

If such properties are known, η_i can be set a priori.

If all clusters have the same properties, the same value for all clusters should be used.

However, information on the actual shape is often unknown a priori.

- So, the parameters must be estimated, *e.g.* by FCM.
- One can use the fuzzy intra-cluster distance, *i.e.* for all $\Gamma_i,\ 1 \leq i \leq n$

$$\eta_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} d_{ij}^{2}}{\sum_{j=1}^{n} u_{ij}^{m}}.$$

Optimizing the Cluster Centers

The update equations j_C are derived by setting the derivative of J_p *w.r.t.* the prototype parameters to zero (holding U_p fixed).

The update equations for the cluster prototypes are identical.

Then the cluster centers in the PCM algorithm are re-estimated as

$$\boldsymbol{c}_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} \boldsymbol{x}_j}{\sum_{j=1}^{n} u_{ij}^{m}}.$$

Revisited Example: The Iris Data

© **Iris Species Database** `http://www.badbear.com/signa/`

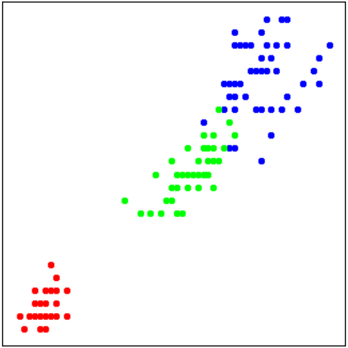
Iris setosa

Iris versicolor

Iris virginica

Collected by Ronald Aylmer Fischer (famous statistician).
150 cases in total, 50 cases per Iris flower type.
Measurements: sepal length/width, petal length/width (in cm).
Most famous dataset in pattern recognition and data analysis.

Example: The Iris Data



Shown: sepal length and petal length.

Iris setosa (red), Iris versicolor (green), Iris virginica (blue)

Cluster Coincidence

characteristic	FCM	PCM
data partition	exhaustively forced to	not forced to
membership degr.	distributed	determined by data
cluster interaction	covers whole data	non
intra-cluster dist.	high	low
cluster number c	exhaustively used	upper bound

Clusters can coincide and might not even cover data.

PCM tends to interpret low membership data as outliers.

A better coverage obtained by

- using FCM to initialize PCM (*i.e.* prototypes, η_i, c),
- after 1st PCM run, re-estimate η_i again,
- then use improved estimates for 2nd PCM run as final solution.

Cluster Repulsion I

J_p is truly minimized only if all cluster centers are identical.

Other results are achieved when PCM gets stuck in a local minimum.

PCM can be improved by modifying J_p:

$$J_{rp}(X, U_p, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} (1 - u_{ij})^m + \sum_{i=1}^{c} \gamma_i \sum_{k=1, k \neq i}^{c} \frac{1}{\eta d(\boldsymbol{c}_i, \boldsymbol{c}_k)^2}.$$

γ_i controls the strength of the cluster repulsion.

η makes the repulsion independent of normalization of data attributes.

Cluster Repulsion II

The minimization conditions lead to the update equation

$$\boldsymbol{c}_i = \frac{\sum_{j=1}^{n} u_{ij}^m \boldsymbol{x}_j - \gamma_i \sum_{k=1, k \neq i}^{c} \frac{1}{d(\boldsymbol{c}_i, \boldsymbol{c}_k)^4} \boldsymbol{c}_k}{\sum_{j=1}^{n} u_{ij}^m - \gamma_i \sum_{k=1, k \neq i}^{c} \frac{1}{d(\boldsymbol{c}_i, \boldsymbol{c}_k)^4}}.$$

This equation shows an effect of the repulsion between clusters:

- A cluster is attracted by data assigned to it.
- It is simultaneously repelled by other clusters.

The update equation of PCM for membership degrees is not modified.

It yields a better detection of shape of very close or overlapping clusters.

Recognition of Positions and Shapes

Possibilistic models do not only carry problematic properties.

The cluster prototypes are more intuitive:

- The memberships depend only on the distance to one cluster.

Shape & size of clusters better fit data clouds than with FCM.

- They are less sensitive to outliers and noise.
- This is an attractive tool in image processing.

Distance Function Variants

Distance Function Variants

So far, only Euclidean distance leading to standard FCM and PCM

Euclidean distance only allows spherical clusters

Several variants have been proposed to relax this constraint

- fuzzy Gustafson-Kessel algorithm
- fuzzy shell clustering algorithms
- kernel-based variants

Can be applied to FCM and PCM

Gustafson-Kessel Algorithm

Replacement of the Euclidean distance by cluster-specific Mahalanobis distance

For cluster Γ_i, its associated Mahalanobis distance is defined as

$$d^2(\boldsymbol{x}_j, C_j) = (\boldsymbol{x}_j - \boldsymbol{c}_i)^T \Sigma_i^{-1} (\boldsymbol{x}_j - \boldsymbol{c}_i)$$

where Σ_i is covariance matrix of cluster

Euclidean distance leads to $\forall i : \Sigma_i = I$, *i.e.* identity matrix

Gustafson-Kessel (GK) algorithm leads to prototypes $C_i = (\boldsymbol{c}_i, \Sigma_i)$

Gustafson-Kessel Algorithm

Specific constraints can be taken into account, *e.g.*

- restricting to axis-parallel cluster shapes
- by considering only diagonal matrices
- usually preferred when clustering is applied for fuzzy rule generation

Cluster sizes can be controlled by $\varrho_i > 0$ demanding $\det(\Sigma_i) = \varrho_i$

Usually clusters are equally sized by $\det(\Sigma_i) = 1$

Objective Function

Identical to FCM and PCM: J, update equations for c_i and U

Update equations for covariance matrices are

$$\Sigma_i = \frac{\Sigma_i^*}{\sqrt[p]{\det(\Sigma_i^*)}}$$

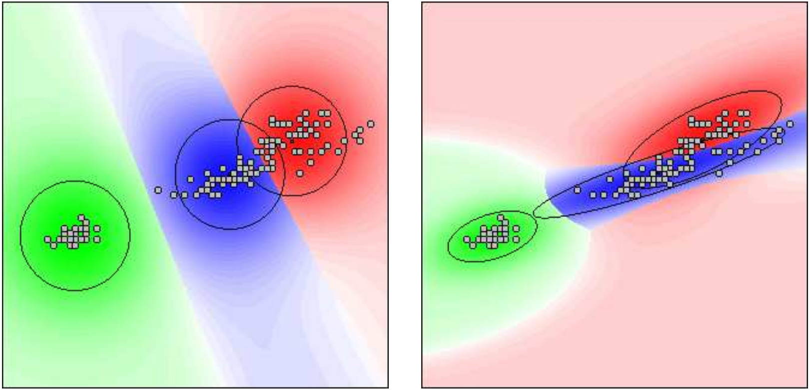
where

$$\Sigma_i^* = \frac{\sum_{j=1}^{n} u_{ij}(\mathbf{x}_j - \mathbf{c}_i)(\mathbf{x}_j - \mathbf{c}_i)^T}{\sum_{j=1}^{n} u_{ij}}$$

Covariance of data assigned to cluster i

Σ_i are modified to incorporate fuzzy assignment

Fuzzy Clustering of the Iris Data



Fuzzy c-Means

Gustafson-Kessel

Summary: Gustafson-Kessel

Extracts more information than standard FCM and PCM

More sensitive to initialization

Recommended initializing: few runs of FCM or PCM

Compared to FCM or PCM: due to matrix inversions GK is

- computationally costly
- hard to apply to huge datasets

Restriction to axis-parallel clusters reduces computational costs

Fuzzy Shell Clustering

Up to now: searched for convex "cloud-like" clusters

Corresponding algorithms = **solid clustering** algorithms

Especially useful in data analysis

For image recognition and analysis:
variants of FCM and PCM to detect lines, circles or ellipses

shell clustering algorithms

replace Euclidean by other distances

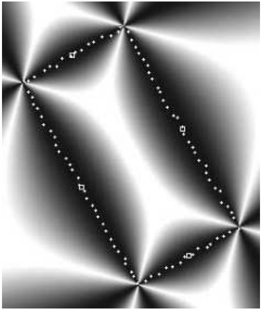
Fuzzy c-varieties Algorithm

Fuzzy c-varieties (FCV) algorithm recognizes lines, planes, or hyperplanes

Each cluster is affine subspace characterized by point and set of orthogonal unit vectors, $C_i = (\mathbf{c}_i, \mathbf{e}_{i1}, \ldots, \mathbf{e}_{iq})$ where q is dimension of affine subspace

Distance between data point $\mathbf{x}_j$ and cluster i

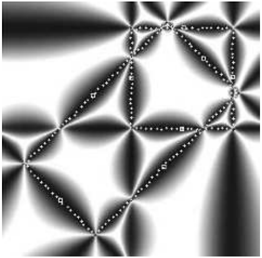
$$d^2(\mathbf{x}_j, \mathbf{c}_i) = \|\mathbf{x}_j - \mathbf{c}_i\|^2 - \sum_{l=1}^{q} (\mathbf{x}_j - \mathbf{c}_i)^T \mathbf{e}_{il}$$



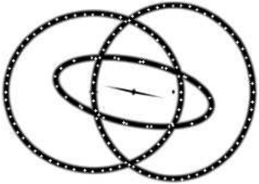
Also used for locally linear models of data with underlying functional interrelations

Other Shell Clustering Algorithms

Name	Prototypes
adaptive fuzzy *c*-elliptotypes (AFCE)	line segments
fuzzy *c*-shells	circles
fuzzy *c*-ellipsoidal shells	ellipses
fuzzy *c*-quadric shells (FCQS)	hyperbolas, parabolas
fuzzy *c*-rectangular shells (FCRS)	rectangles



AFCE



FCQS

FCRS

Kernel-based Fuzzy Clustering

Kernel variants modify distance function to handle non-vectorial data, *e.g.* sequences, trees, graphs

Kernel methods [Schölkopf and Smola, 2001] extend classic linear algorithms to non-linear ones without changing algorithms

Data points can be vectorial or not $\Rightarrow x_j$ instead of $\boldsymbol{x}_j$

Kernel methods: based on mapping $\phi : \mathcal{X} \to \mathcal{H}$

Input space $\mathcal{X}$, feature space $\mathcal{H}$ (higher or infinite dimensions)

$\mathcal{H}$ must be Hilbert space, *i.e.* dot product is defined

Principle

Data are not handled directly in $\mathcal{H}$, only handled by dot products

Kernel function

$$k : \mathcal{X} \times \mathcal{X} \to \mathrm{I\!R}, \forall x, x' \in \mathcal{X} : \langle \phi(x), \phi(x') \rangle = k(x, x')$$

No need to known ϕ explicitly

Scalar products in $\mathcal{H}$ only depend on k and data $\Rightarrow$ **kernel trick**

Kernel methods $=$ algorithms with scalar products between data

Kernel Fuzzy Clustering

Kernel framework has been applied to fuzzy clustering

Fuzzy shell clustering extracts prototypes, kernel methods do not

They compute similarity between $x, x' \in \mathcal{X}$

Clusters: no explicit representation

Kernel variant of FCM [Wu et al., 2003] transposes J_f to $\mathcal{H}$

Centers $c_i^\phi \in \mathcal{H}$ are linear combinations of transformed data

$$c_i^\phi = \sum_{r=1}^{n} a_{ir}\phi(x_r)$$

Kernel Fuzzy Clustering

Euclidean distance between points and centers in $\mathcal{H}$ is

$$d^2_{\phi_{ir}} = \left\|\phi(x_r) - c_i^\phi\right\|^2 = k_{rr} - 2\sum_{s=1}^{n} a_{is} k_{rs} + \sum_{s,t=1}^{n} a_{is} a_{it} k_{st}$$

whereas $k_{rs} \equiv k(x_r, x_s)$

Objective function becomes

$$J_\phi(X, U_\phi, C) = \sum_{i=1}^{c}\sum_{r=1}^{n} u_{ir}^m d^2_{\phi_{ir}}$$

Minimization leads to update equations:

$$u_{ir} = \frac{1}{\sum_{l=1}^{c}\left(\frac{d^2_{\phi_{ir}}}{d^2_{\phi_{lr}}}\right)^{\frac{1}{m-1}}}, \quad a_{ir} = \frac{u_{ir}^m}{\sum_{s=1}^{n} u_{is}^m}, \quad c_i^\phi = \frac{\sum_{r=1}^{n} u_{ir}^m \phi(x_r)}{\sum_{s=1}^{n} u_{is}^m}$$

Summary: Kernel Fuzzy Clustering

Update equations (and J_ϕ) are expressed by k

For Euclidean distance, membership degrees are identical to FCM

Cluster centers: weighted mean of data (comparable to FCM)

Disadvantage of kernel methods:

- choice of proper kernel and its parameters
- similar to feature selection and data representation
- cluster centers belong to $\mathcal{H}$ (no explicit representation)
- only weighting coefficients a_{ir} are known

Objective Function Variants

Objective Function Variants

So far, variants of FCM with different distance functions

Now, other variants based on modifications of J

Aim: improving clustering results, *e.g.* noisy data

Many different variants:

- explicitly handling noisy data
- modifying fuzzifier m in objective function
- new terms in objective function (*e.g.* optimize cluster number)
- improving PCM *w.r.t.* coinciding cluster problem

Noise Clustering

Noise clustering (NC) adds to c clusters one noise cluster

- shall group noisy data points or outliers
- not explicitly associated to any prototype
- directly associated to distance between implicit prototype and data

Center of noise cluster has constant distance δ to all data points

- all points have same "probability" of belonging to noise cluster
- during optimization, "probability" is adapted

Noise Clustering

Noise cluster: added to objective function as any other cluster

$$J_{nc}(X, U, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} d_{ij}^{2} + \sum_{k=1}^{n} \delta^{2} \left(1 - \sum_{i=1}^{c} u_{ik}\right)^{m}$$

Added term: similar to terms in first sum

- distance to cluster prototype is replaced by δ
- outliers can have low membership degrees to standard clusters

J_{nc} requires setting of parameter δ, *e.g.*

$$\delta = \lambda \frac{1}{c \cdot n} \sum_{i=1}^{c} \sum_{j=1}^{n} d_{ij}^{2}$$

λ user-defined parameter: if low λ, then high number of outliers

Fuzzifier Variants

Fuzzifier m introduces problem:

$$u_{ij} = \begin{cases} \{0,1\} & \text{if } m = 1, \\ ]0,1[& \text{if } m > 1 \end{cases}$$

Disadvantage for noisy datasets (to be discussed in the exercise)

Possible solution: convex combination of hard and fuzzy c-means

$$J_{hf}(X, U, C) = \sum_{i=1}^{c} \sum_{j=1}^{n} \left[\alpha u_{ij} + (1-\alpha) u_{ij}^2\right] d_{ij}^2$$

where $\alpha \in [0, 1]$ is user-defined threshold

FUZZY DATA Analysis

Random Sets

Standard statistical data analysis is based on random variables

$X : \Omega \to U$

A measurable mapping from a probability space to a set U, i.e. $U = \mathbb{R}$

A random set $\Gamma : \Omega \to 2^U$

is a generalization where the outcome is a subset of U

Example: Languages

$U = \{$ English, German, French, Spanish $\}$ Languages

Ω Employees of a working group, P uniform distribution on Ω

$\Gamma : \Omega \to 2^L$ collection of languages ω can speak

Typical questions and answers in this context:

- What is the proportion P_1 of employees that can speak German and English and cannot speak any other language?
  $P_1 = P\left(\{\omega \in \Omega : \Gamma(\omega) = \{\text{English}, \text{German}\}\}\right)$
- What is the proportion P_2 of employees that can speak German or English but no other language?
  $P_2 = P\left(\{\omega \in \Omega : \Gamma(\omega) \subseteq \{\text{English}, \text{German}\}\}\right)$
- What is the proportion P_3 of employees that can speak German or English?
  $P_3 = P\left(\{\omega \in \Omega : \Gamma(\omega) \cap \{\text{English}, \text{German}\} \neq \emptyset\}\right)$
- What is the proportion P_4 of employees that can speak at least three languages?
  $P_4 = P\left(\{\omega \in \Omega : |\Gamma(\omega)| \geq 3\}\right)$

Upper and Lower Probability

$(\Omega, 2^\Omega, P)$ finite, $\Gamma : \Omega \to 2^U$

Proportion of elements whose images “touch” a given subset

upper probability of A: $P^\star(A) = P\left(\{\omega \in \Omega | \Gamma(\omega) \cap A \neq \emptyset\}\right)$

Proportion of elements whose image is fully contained in a given subset

lower probability of A: $P_\star(A) = P\left(\{\omega \in \Omega | \Gamma(\omega) \subseteq A, \Gamma(\omega) \neq \emptyset\}\right)$

Example: Mean Temperature

Ω Days in 1984, P uniform distribution on Ω

U = Temperature, only $T_{min}(\omega)$, $T_{max}(\omega)$ the max-min temperature in Branschweig are recorded

$\Gamma : \Omega \to 2^{\mathbb{R}}, \Gamma(\omega) = [T_{min}(\omega), T_{max}(\omega)]$,

What is the mean temperature at 18:00h in 1984?

$X : \Omega \to \mathbb{R}$, true (but unknown) temperature at 18:00h in 1984 on day ω.

$T_{min}(\omega) \leq X_0(\omega) \leq T_{max}(\omega)$ hold for all $\omega \in \Omega$

EX expected value of X, we only know $ET_{min} \leq EX \leq ET_{max}$

Descriptive Analysis of Imprecise Data

$(\Omega, 2^{\Omega}, P)$ finite, $\Gamma : \Omega \to 2^{U}$

$$E(\Gamma) = \{E(X) | X(\omega) \in \Gamma(\omega), \\ X \text{ is random variable such that } E(X) \text{ exists and } \forall \omega \in \Omega\}$$

This method can be used for other quantities such as the variance

Ontic and epistemic view

Ontic view of A: Several elements of A may be true (several languages)

Epistemic view of A: Only one element of A is true (one temperature)

Possibility Theory

Possibility Theory

Epistemic view of fuzzy sets

Possibility disribution π quantifies the state of knowledge

$\pi : X \to [0, 1]$, with an $x_0 \in X$ such that $\pi(x_0) = 1$.

$\pi(u) = 0$: u is rejected as impossible

$\pi(u) = 1$: u is totally possible

Specificity of possibility distributions

π is at least as specific as π' iff

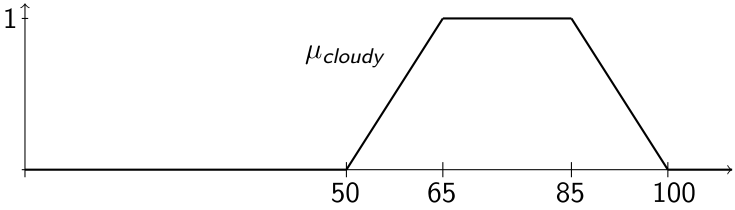
for each x: $\pi(x) \leq \pi'(x)$ holds.

Example: How Cloudy is Brunswick?

Given: remark that weather was 'cloudy'

Fuzzy set $\mu_{cloudy} : X \to [0, 1]$, where $X = [0, 100]$, the imprecise concept cloudy.

$x \in X$ clouding degree in percent, $\mu_{cloudy}(x)$ membership degree of x to μ_{cloudy}.



Estimate $\pi(x) := \mu(x)$, for all $x \in \mathbb{R}$.

40 rejected impossible, 70 totally possible, 60 possible with degree 0.66

Possibility and Necessity

Let $\pi : X \to [0,1]$ a possibility distribution.

Possibility degree of $A \subseteq X$: $\Pi(A) := \sup\{\pi(x) : x \in A\}$

Necessity degree of $A \subseteq X$: $N(A) := \inf\{1 - \pi(x) : x \in \overline{A}\}$.

$\Pi(A)$ evaluates to what extent A is consistent with π

$N(A)$ evaluates to what extent A is certainly implied.

Duality expressed by: $N(A) = 1 - \Pi(\overline{A})$ for all A.

It holds:

$$\begin{aligned}
\Pi(X) &= N(X) = 1 \\
\Pi(\emptyset) &= N(\emptyset) = 0, \text{ and} \\
\Pi(A \cup B) &= \max\{\Pi(A), \Pi(B)\} \text{ for all } A \text{ and } B \\
\Pi(A \cap B) &\leq \min\{\Pi(A), \Pi(B)\} \text{ for all } A \text{ and } B
\end{aligned}$$

Fuzzy Random Variables

Random Fuzzy Sets

$X : \Omega \to U$ random variable

$\Gamma : \Omega \to 2^U$ random set

$\Gamma : \Omega \to \mathcal{F}(U)$ random fuzzy set / fuzzy random variable

Example: Languages

$U = \{$ English, German, French, Spanish $\}$ Languages

Ω Employees of a working group, p uniform distribution on Ω

To each person ω and each language u we assign the result of the European Language Test on a [0,1] scale $\Gamma_\omega(u)$

$\Gamma_\omega : U \to [0, 1]$ in fuzzy set describing the language competence of ω

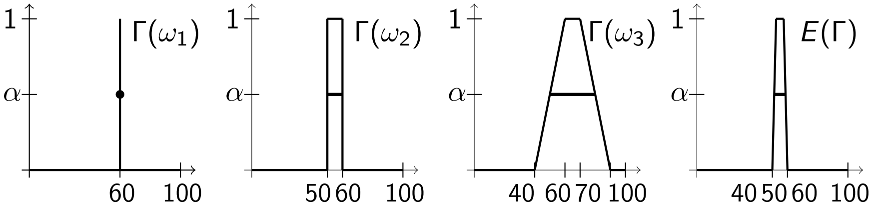
What is the probability that the people in the group speak both English and Spanish to a degree of at least 0.8?

Result can be found by analysis: $\Gamma : \Omega \to 2^U, \omega \mapsto \Gamma_\omega$

$P\left(\{\omega \in \Omega | \Gamma(\omega) \geq \mu\}\right)$, $\mu : U \to [0, 1]$,
$\mu(English) = 0.8, \mu(Spanish) = 0.8, \mu(German) = \mu(French) = 0$

Example: Clouding degrees

Analyze observations of clouding degrees for three days given



For one day precise, for one interval-valued, one subjective by possibility distribution

How to determine location and range parameters like mean value and variance?

Expected Value

(1) Define possibility distribution on the set of all random variables, describing possibility of random set being the original

(2) Apply extension principle to mapping that assigns to each random variable its expected value.

$U : \Omega \to X$ a random variable

Possibility degree that $U(\omega)$ is the original of $\Gamma(\omega)$ is $(\Gamma(\omega))(U(\omega))$

Possibility that U is the original on Γ is

$$\pi_\Gamma(U) := \inf_{\omega \in \Omega} \{\Gamma(\omega)(U(\omega))\}$$

Expected Value

$\Gamma : \Omega \to F(X)$ fuzzy random variable

Expected value $E(\Gamma) : X \to [0,1]$ fuzzy set of X:

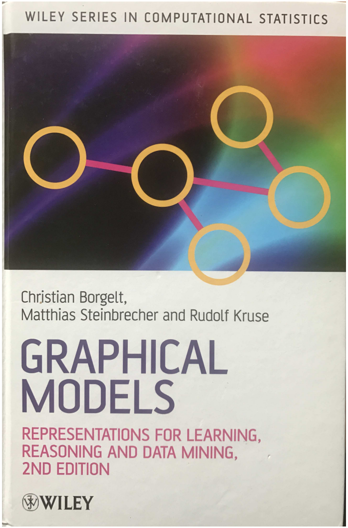
$$x \mapsto \sup_{U:E(U)=x} \left\{ \min_{\omega\in\Omega} \{(\Gamma(\omega))(U(\omega))\} \right\}$$

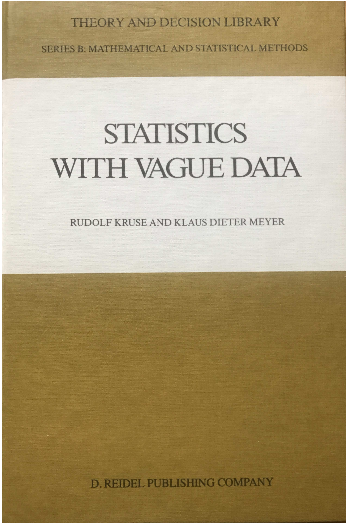
Variance can be defined similar

If probability space finite $\Omega = \{\omega_1, \ldots, \omega_n\}$ and possibility distributions on $\mathbb{R}$: calculation simplifies to

$$[E(\Gamma)]_\alpha = \sum_{\omega\in\Omega} P\{\omega\} \cdot [\Gamma(\omega)]_\alpha \text{ for } \alpha > 0$$

References I





References II

Bandemer, H. and Näther, W. (1992).
Fuzzy data analysis. series: Mathematical and statistical methods.

Bezdek, J. (1981).
Pattern Recognition With Fuzzy Objective Function Algorithms.
Plenum Press, New York, NY, USA.

Bezdek, J. C., of Electrical, I., Council, E. E. N. Y. N. N., and Pal, S. K. (1992).
Fuzzy models for pattern recognition: methods that search for structures in data.
IEEE press.

Borgelt, C., Steinbrecher, M., and Kruse, R. R. (2009).
Graphical models: representations for learning, reasoning and data mining.
John Wiley & Sons.

Couso, I., Dubois, D., and Sánchez, L. (2014).
Random sets and random fuzzy sets as ill-perceived random variables.

Dubois, D. and Prade, H. (2015).
Possibility theory and its applications: Where do we stand?
In *Springer Handbook of Computational Intelligence*, pages 31–60. Springer.

References III

Höppner, F., Klawonn, F., Kruse, R., and Runkler, T. (1999).
Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition.
John Wiley & Sons Ltd, New York, NY, USA.

Hüllermeier, E. (2005).
Fuzzy methods in machine learning and data mining: Status and prospects.
Fuzzy sets and Systems, 156(3):387–406.

Keller, A. and Kruse, R. (2002).
Fuzzy rule generation for transfer passenger analysis.
In Wang, L., Halgamuge, S. K., and Yao, X., editors, *Proceedings of the 1st International Conference on Fuzzy Systems and Knowledge Discovery (FSDK'02)*, pages 667–671, Orchid Country Club, Singapore.

Kruse, R. (1987).
On the variance of random sets.
Journal of Mathematical Analysis and Applications, 122(2):469–473.

Kruse, R. and Meyer, K. D. (1987).
Statistics with Vague Data.
D. Reidel Publishing Company, Dordrecht.

References IV

Schölkopf, B. and Smola, A. J. (2001).
Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond.
MIT Press, Cambridge, MA, USA.

Viertl, R. (2011).
Statistical methods for fuzzy data.
John Wiley & Sons.

Walley, P. (1991).
Statistical reasoning with imprecise probabilities.

Wu, Z., Xie, W., and Yu, J. (2003).
Fuzzy c-means clustering algorithm based on kernel method.
In *Proceedings of the Fifth International Conference on Computational Intelligence and Multimedia Applications (ICCIMA)*, pages 1–6.