Intelligent Data Mining Techniques

(tutorial presented at ANNIE´2003)

Iveta Mrázová

Department of Software Engineering

Charles University Prague

Content outline

- Intelligent Data Mining: introduction and overview of Intelligent Data Mining Techniques (20 min)
- Selected Data Mining Techniques: principles and examples
  - undirected DM-techniques:
    - Market Basket Analysis (MBA) - (20 min)
    - Link Analysis and Scale-Free Networks (10 min)
    - Automatic Cluster Detection and Fuzzy Systems: Clustering the World Bank Data (20 min)
  - directed DM-techniques:
    - Internal Knowledge Representation in BP-Networks (20 min)
    - Modular Networks, Sensitivity Analysis and Feature Selection (20 min)
    - Neural Networks and Decision Trees: Students´Questionnaire (20 min)
    - Genetic Algorithms and BP-networks: Generating Melodies (10 min)
- Conclusions, Questions + Answers (10 min)

Intelligent Data Mining: References

- M. J. A. Berry, G. Linoff: *Data Mining Techniques for Marketing, Sales, and Customer Support*, John Wiley & Sons, 1997
- M. J. A. Berry, G. Linoff: *Mastering Data Mining*, John Wiley & Sons, 2000
- J. Han, M. Kamber: *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001
- D. Hand, H. Mannila, P. Smyth: *Principles of Data Mining*, The MIT Press, 2001
- D. Pyle: *Data Preparation for Data Mining*, Morgan Kaufmann Press, 1999
- I. H. Witten, E. Frank: *Data Mining: practical machine learning tools and techniques with Java implementations*, Morgan Kaufmann Publishers, 2000

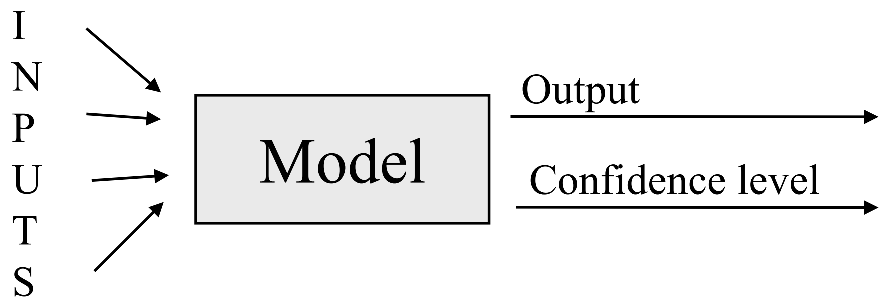
  http://www.mkp.com/datamining

  http://www.cs.waikato.ac.nz/ml/weka

What is Data Mining?

- **discovering patterns in data**
  - discovered patterns should be **meaningful**
  - should lead to some **advantage**, e.g. economic, …
  - allows to make **non-trivial predictions** on new data
- the data is present in substantial **quantities**
- **automatic** or semi-automatic process
- two extremes for the form of discovered patterns:
  - **<u>black box</u>** - e.g. neural networks
  - **<u>transparent box</u>** - more structured, capture the decision structure in an explicit way

Building models for the data



- Classification model:
  - assigns an existing classification to new records
- Predictive model
  - Time-series model
- Clustering model

Data Analysis: Influence of other disciplines

- statistics → interpret observations
- sampling → reduce the size of data
- regression analysis → inter- and extrapolate observations
  - linear regression → fit a line to observed data
- correlation analysis → mutual occurrence of observations
- memory-based reasoning → directly from AI
- link analysis → graph theory
- genetic algorithms and neural networks → model biological processes

Intelligent DM-Techniques: an overview

- Market Basket Analysis (MBA)
- Memory-Based Reasoning (MBR)
- Automatic Cluster Detection
- Fuzzy Systems (FS)
- Link Analysis
- Decision Trees
- Artificial Neural Networks (ANN)
- Genetic Algorithms (GA)

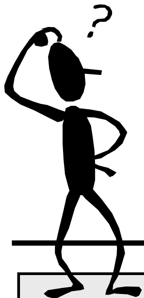
Market Basket Analysis (MBA)

- Analyses in the retail industry:

What items occur together in a "basket"?

- Results:
  - expressed as rules
  - highly actionable

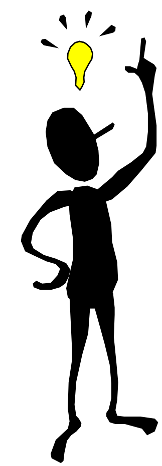
- Applications:
  - planning store layouts
  - offering coupons, limiting specials
  - bundling products



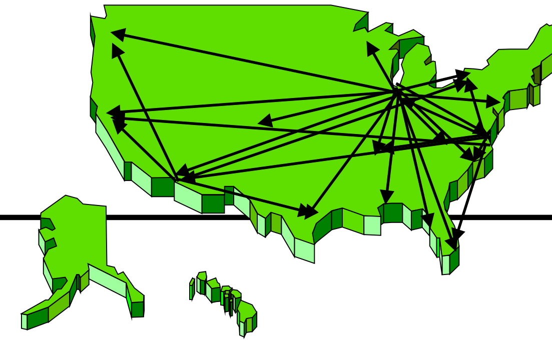
Memory-Based Reasoning (MBR)

Look for the nearest "known" neighbor to classify or predict value!

- applicable to virtually any data
- new instances learned by adding them to the data set
- distance to neighbors estimates the correctness of the results
- Key elements in MBR:
  - ***distance function*** - to find nearest neighbors
  - ***combination function*** - combine values at nearest neighbors to classify or predict

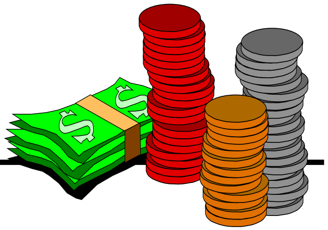


Link Analysis



- Goals:
  - ***find patterns in relationships between records***
  - ***visualize the links***
- Application areas:
  - telecommunications
  - law enforcement - clues about crimes are linked together to solve them
  - marketing - relationships between customers

Automatic Cluster Detection

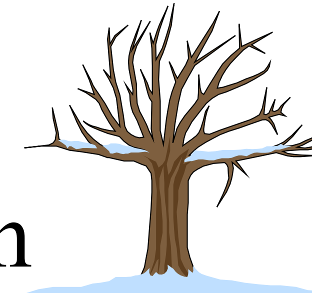


- <u>Goal:</u>

Find previously unknown similarities in the data!

- Build models that find data records similar to each other
- Good as an initial analysis of the data
- Undirected data mining

Decision Trees and Rule Induction



Divide the data into disjoint subsets characterized by simple rules!

- Directed data mining (classification)
- Explainable rules applicable directly to new records
- Techniques:
  - Classification And Regression Trees (CART)
  - Chi-squared Automatic Induction (CHAID)
  - C4.5

Artificial Neural Networks (ANN)

Detect patterns in the data in a way "similar" to human thinking!

- Directed data mining (classification and prediction)
- Applicable also to undirected data mining (SOMs)
- <u>Two major drawbacks:</u>
  - difficulty in understanding the models they produce
  - sensitivity to the format of incoming data

Genetic Algorithms (GA)

Apply genetics and natural selection to find optimal parameters of a predictive function!

- GA use "genetic" operators to evolve successive generations of solutions:
  - selection
  - crossover
  - mutation
- Best candidates "survive" to further generations until convergence is achieved
- Directed data mining

On-Line Analytic Processing (OLAP)

- an important tool for extracting and presenting information
- facilitates understanding of the data and important patterns inside it
- a way of presenting relational data to users
- multi-dimensional databases (MDDs):
  - a representation of data
  - allows users to drill down into the data and understand various important summarizations

Market Basket Analysis (MBA)

- Analyses in the retail industry:

What items occur together in a "basket"?

- Results:
  - expressed as rules
  - highly actionable

- Applications:
  - planning store layouts
  - offering coupons, limiting specials
  - bundling products

Association rules

How do the products relate one to each other?

- <u>Association rules should be:</u>
  - ***easy to understand:*** once the pattern is found, it is easy to justify it
  - ***useful:*** contain actionable information leading to other interventions
- <u>Association rules should not be:</u>
  - ***trivial:*** results are already known by anyone familiar with the business
  - ***inexplicable:*** seem to have no explanation and do not suggest any action

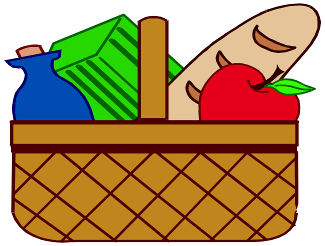
MBA to compare stores

- Virtual items:
  - specify which group the transaction comes from
  - do not correspond to a product or service
- Comparison between new and existing stores:
  1. Gather data for a specific period from store openings
  2. Gather about the same amount of data from existing stores
  3. Apply MBA to find association rules in each set
  4. Consider especially association rules containing the virtual items

MBA - how does it work?

- ***Items*** - products or service offerings
- ***Transactions*** contain one or more ***items***
- ***Co-occurrence table***
  - indicates the number of times that any two ***items co-occur*** in a ***transaction*** (i.e. these products were purchased together)
  - values along the diagonal represent the ***number of transactions*** containing just that one item

MBA - example



- Grocery transactions:

Customer	Items
1	bread, butter
2	milk, bread, butter
3	bread, coffee
4	bread, butter, coffee
5	coffee, butter

- Co-occurrence of products:

	bread	butter	milk	coffee
bread	4	3	1	2
butter	3	4	1	2
milk	1	1	1	0
coffee	2	2	0	3

Sales patterns apparent from the co-occurrence table:

Bread and butter are likely to be purchased together.
Milk is never purchased with coffee.

MBA - Association rules

- Rule:

  IF *Condition* THEN *Result.*

  (*Rule_r* : IF *Item_i* THEN *Item_j* .)

- **Questions:**
  - How good are the found association rules?
    - support
    - confidence
    - improvement
  - How to find association rules automatically?

Support and confidence

Support: ***How frequently can the rule be applied?***

$$Support(Rule\_r) = \frac{Nr\_of\_Transactions\_containing\_i\_and\_j}{Number\_of\_all\_Transactions} \bullet 100\,\%$$

Confidence: ***How much can we rely on the result of the rule?***

$$Confidence(Rule\_r) = \frac{Nr\_of\_Transactions\_containing\_i\_and\_j}{Nr\_of\_Transactions\_containing\_i} \bullet 100\,\%$$

Support and confidence - example

Rule 1: ***If* a customer purchases *bread* *then* the customer also purchases *butter*.**

Rule 2: ***If* a customer purchases *coffee* *then* the customer also purchases *butter*.**

Support (Rule_1) = 3 / 5 • 100 % = 60 %

Support (Rule_2) = 2 / 5 • 100 % = 40 %

Confidence (Rule_1) = 3 / 4 • 100 % = 75 %

Confidence (Rule_2) = 2 / 3 • 100 % = 66 %

Improvement of a rule

<u>Improvement:</u> ***How much is a rule better at predicting the result than just assuming it?***

$$Improvement(Rule\_r) = \frac{p(i\_and\_j)}{p(i) \bullet p(j)}$$

<u>If Improvement < 1:</u>

- rule is worse at predicting the result than random chance
- <u>NEGATING</u> the result might produce a better rule

IF *Condition* THEN NOT *Result.*

Improvement of a rule - example

Rule: ***If* a customer purchases *milk* *then* the customer also purchases *butter*.**

$$\mathit{Support}\,(\,\mathit{Rule\_1}\,) = 1/5 \bullet 100\,\% = 20\,\%$$

$$\mathit{Confidence}\,(\,\mathit{Rule\_1}\,) = 1/1 \bullet 100\,\% = 100\,\%$$

$$\mathit{Improvement}\,(\,\mathit{Rule\_1}\,) = (\,1/5\,)\,/\,(\,(\,1/5\,) \bullet (\,4/5\,)\,) = 5/4 = 1.25$$

Basic steps of MBA

- **Choose** the right set of **items** and the right level
- **Generate rules** by deciphering the co-occurrence matrix
  - calculate the probabilities and joint probabilities of items and their combinations in transactions
  - limit the search with thresholds set on support
- Analyze probabilities to **determine best rules**
  - overcome limits imposed by the number of items and their combinations in “interesting” transactions

MBA - the choice of right items

<u>Gathering transaction data:</u>

- often bad quality requiring extensive pre-processing
- items of interest may change over time
- the right level of detail:
  - a growing number of item combinations
  - actionable results (specific items)
  - rules with sufficient support (frequent occurrence in the data set)

Taxonomies: hierarchical categories

MBA - Complexity of generated rules:

- Use more general items initially
- Then, generate rules for more specific items using only transactions containing these items

MBA - Actionable results:

Items should occur in roughly the same number of transactions:

- roll up rare items to higher levels in the taxonomy (to become more frequent)
- keep more common items at lower levels (to prevents rules from being dominated by the most common items)

Virtual items: go beyond the taxonomy

- cross product boundaries of original items
  - e.g. designer labels - Calvin Klein
- may include information about the transaction itself
  - ***anonymous*** (day of week, time, etc.)
  - ***signed*** (info about customers and their behavior over time)
- might be a cause of redundant rules
  - items from the taxonomy are associated with just one virtual item ("*If Coke product then Coke.*")
  - virtual and generalized items appear together in a rule ("*If Coke product and diet soda then pretzels*" instead of "*If diet coke then pretzels*")

MBA - generating rules

- <u>Compute the co-occurrence table:</u>
  - provides the information about which combinations of items occur most commonly in the transactions
  - applicable for evaluating basic probabilities necessary to evaluate the importance of generated rules
- <u>Provide useful rules:</u>
  - improvement should be greater than 1
    - low improvement can be increased by negating the rules
    - negated rules might be less actionable than original rules
  - reduce the number of generated rules - <u>**PRUNING**</u>

Minimum support pruning

<u>Eliminate less frequent items</u>

- actions should ***affect enough transactions***
- two possibilities:
  - eliminate rare items from consideration (then, eliminate their respective associative rules)
  - use taxonomy to generalize items (then, resulting generalized items should meet the threshold criteria)
- ***variable minimum support*** - a cascading effect

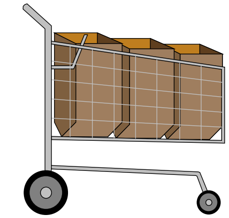
MBA - Dissociation rules

- <u>Rule:</u> **IF *A* <u>AND NOT</u> *B* THEN *C*.**
  - Introduce new items inverse to original ones
  - Each transaction will contain an inverse item if it does not contain the original one
- <u>Drawbacks:</u>
  - doubled number of items
  - growing size of transactions
  - inverse items tend to occur more frequently than original (leading to less actionable rules with all items inverted:
    "IF <u>NOT</u> *A* AND <u>NOT</u> *B* THEN <u>NOT</u> *C*.")

Time-series analysis with MBA

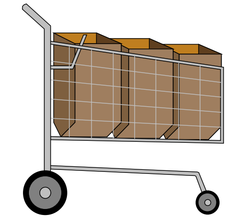
- **<u>Analyze cause and effects:</u>**
  - time- or sequencing information to determine when transactions occurred relative to each other
  - usually requires some way of identifying the customer
- **<u>Conversions to an MBA-problem:</u>**
  - include in transactions items before the event of interest (for *causes*) or after the event of interest (for *effects*); then, remove duplicate items from the transaction
  - ***time-window***: a "snapshot" of all items that occur within a certain period (e.g. all transactions within a month)
    - trends for rare items

Strengths of MBA



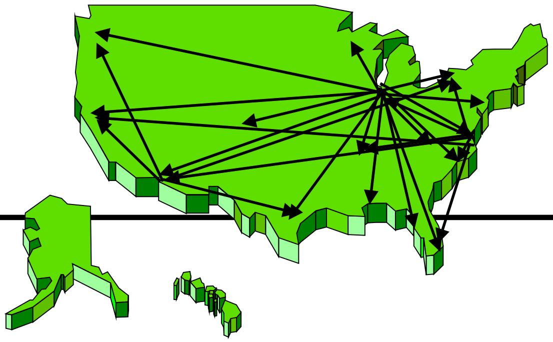
- Produces clear and understandable results
  - ***actionable IF - THEN - rules***
- Supports ***undirected data mining***
  - important when approaching large data sets with no prior knowledge
- Works on ***variable-length data***
- Computations are ***easy to understand***
  - Computational costs grow exponentially with the number of items!

Weaknesses of MBA



- Exponentially growing computational costs
  - necessity for item taxonomies and virtual items
- Limited support for attributes on the data
  - pruning of less actionable general items
- Difficult to determine the right number of items
  - items should have approximately the same frequency
- Discounts rare items
  - variable thresholds for minimum support pruning
  - higher levels in item taxonomies

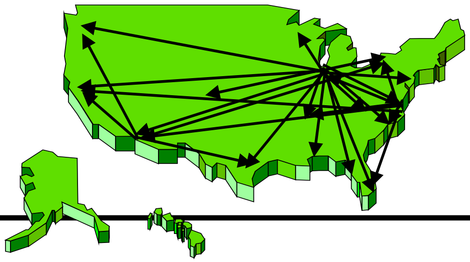
Link Analysis



- Goals:
  - ***find patterns in relationships between records***
  - ***visualize the links***

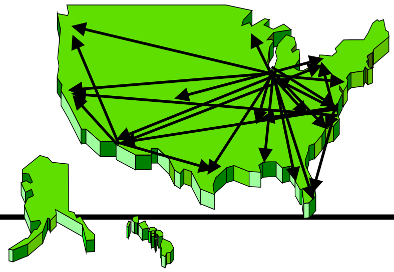
- Application areas:
  - telecommunications
  - law enforcement - clues about crimes are linked together to solve them
  - marketing - relationships between customers

Scale-Free Networks

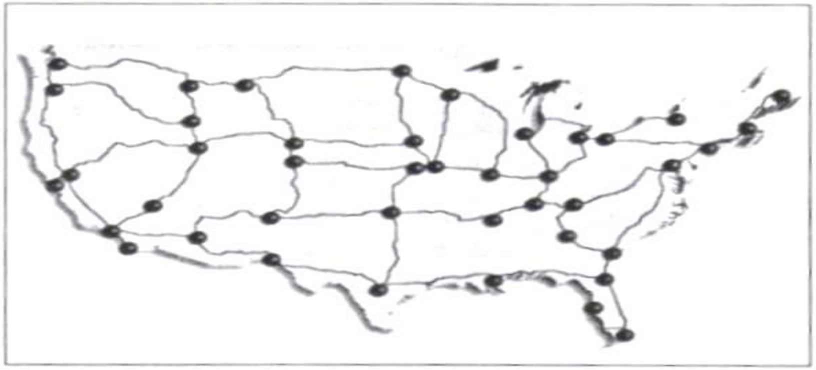


- Some nodes have an extremely large number of links (edges) to other nodes - **hubs**
- Most nodes have just a few links to other nodes
- Robust against accidental failures
- Vulnerable to coordinated attacks
- New application areas
  - preventing computer viruses spreading through the Internet
  - medicine (vaccinations)
  - business (marketing)

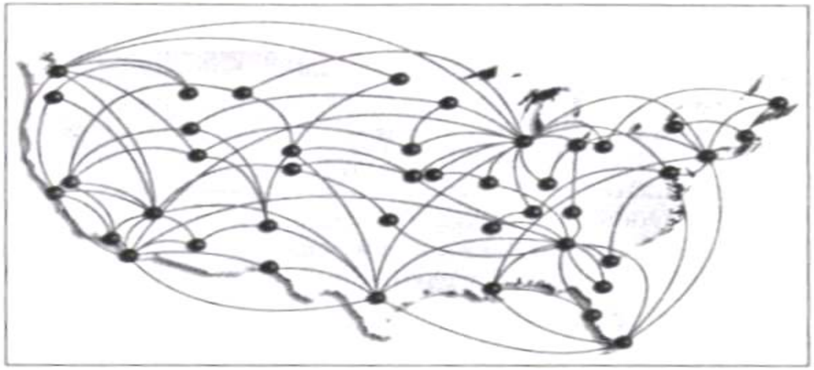
Scale-Free Networks



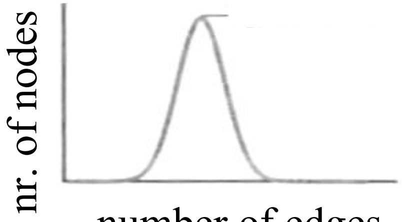
A random graph



A scale-free network



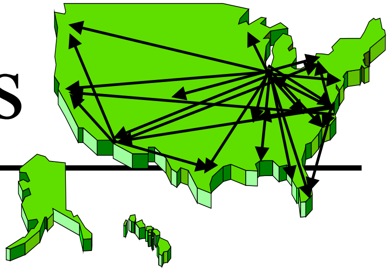
Distribution of edges



Distribution of edges

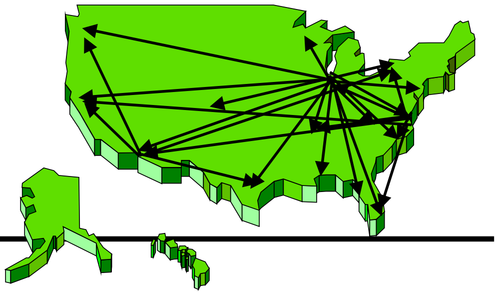
adapted from "A. L. Barabasi and E. Bonabeau: *Scale-Free Networks*, Scientific American, May 2003"

Examples of Scale-Free Networks

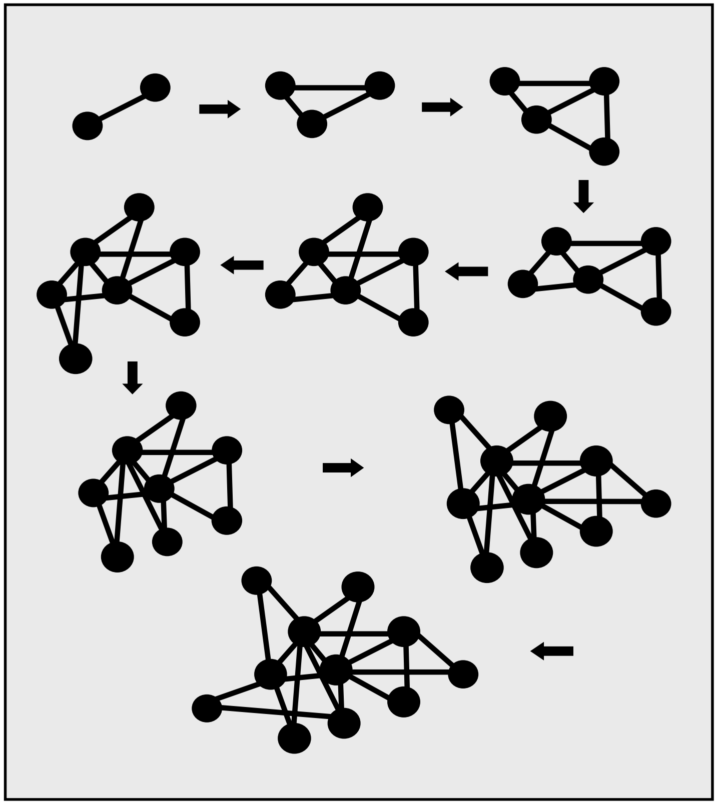


- Social networks
  - research collaboration (scientists, co-authorship of papers)
  - Hollywood (actors, appearance in the same movie)
- Biological networks
  - cellular metabolism (molecules involved in energy production, participation in the same biological reaction)
  - protein regulatory network (proteins controlling cell activity, interactions among proteins)
- Socio-technical networks
  - Internet (routers, optical or other connections)
  - World Wide Web (Web-pages and URLs)

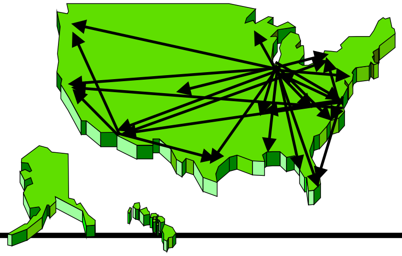
Scale-Free Networks: basic characteristics

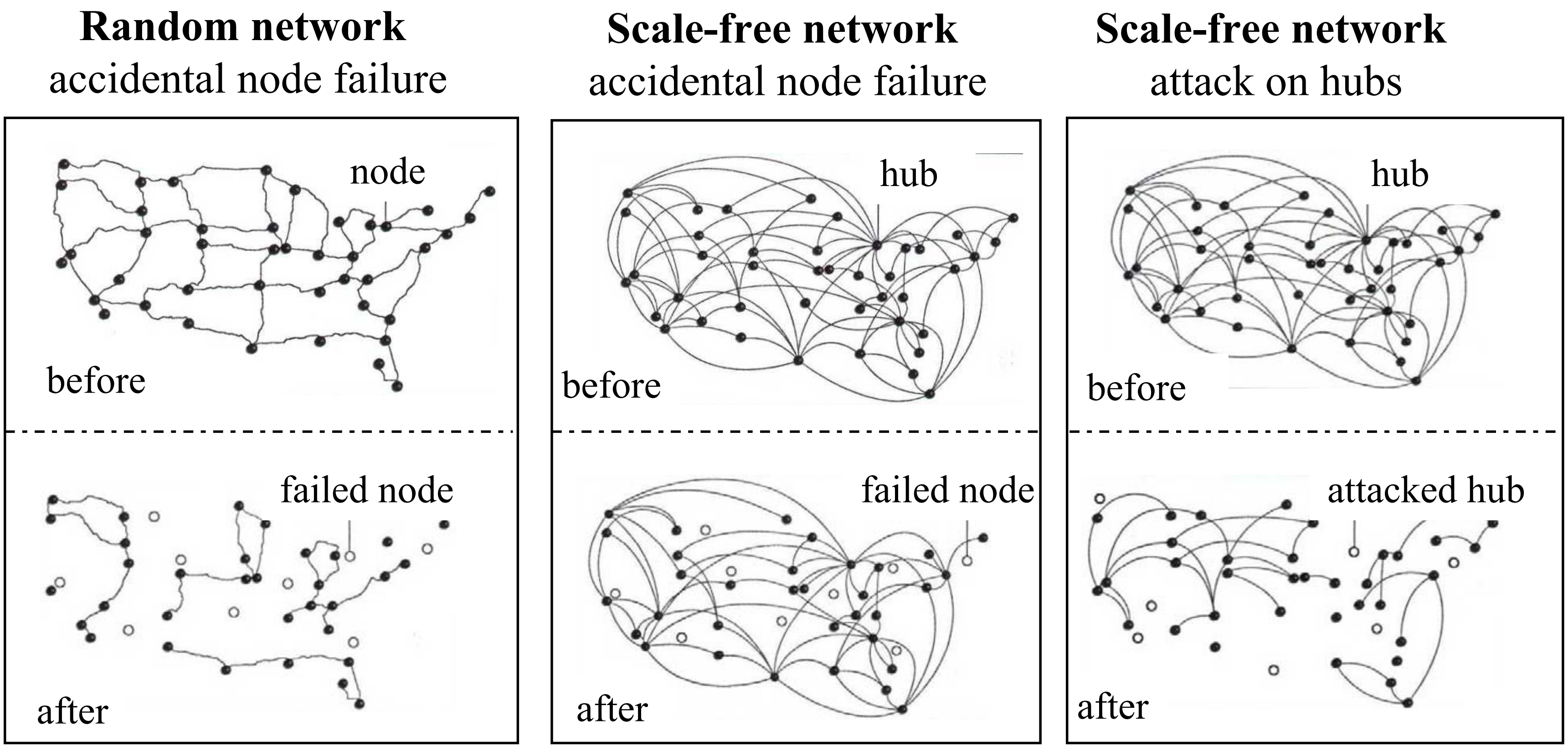


- Two basic mechanisms:
  - **growth**
  - **preferential attachment**
- "The rich get richer" (hubs):
  - new nodes tend to connect to the more connected sites
  - "popular locations" acquire more links over time than less connected neighbors
- Reliability
  - ***accidental failures*** (80% of *randomly selected* nodes can fail without fragmenting the cluster)
  - ***coordinated attacks*** (eliminating 5-15% of *all hubs* can crash the system)



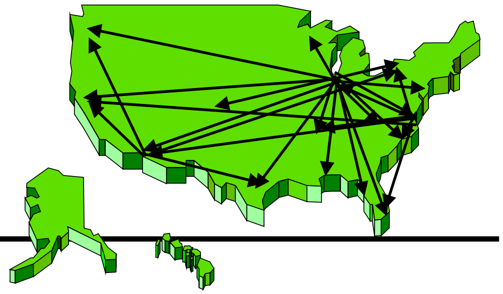
Scale-Free Networks





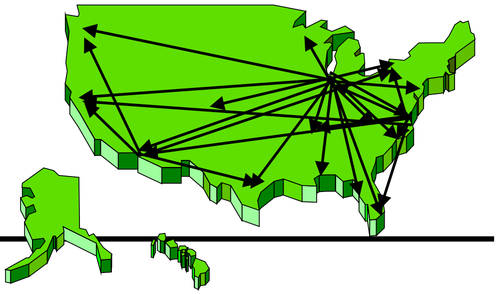
adapted from “A. L. Barabasi and E. Bonabeau: *Scale-Free Networks*, Scientific American, May 2003”

Implications of Scale-Free Networks



- **<u>Computing</u>**
  - networks with scale-free architectures
- **<u>Medicine</u>**
  - vaccination campaigns and new drugs
- **<u>Business</u>**
  - cascading financial failures
  - marketing

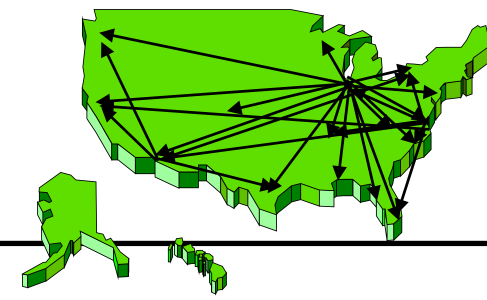
Implications of Scale-Free Networks



Computing

- computer networks with scale-free architectures (e.g. WWW)
  - highly resistant to accidental failures
  - very vulnerable to deliberate attacks and sabotage
- eradicating viruses from the Internet will be effectively impossible

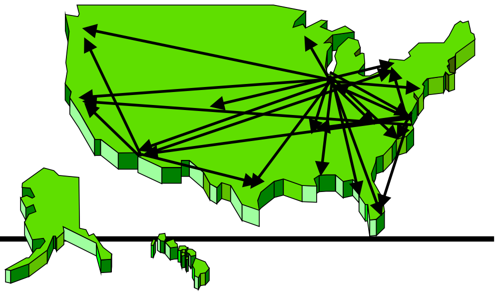
Implications of Scale-Free Networks



Medicine

- vaccination campaigns against serious viruses focused on hubs
  - people with many connections to others
  - difficult to identify such people
- new drugs targeting the hub molecules involved in certain diseases
- control the side-effects of drugs with maps of networks within cells

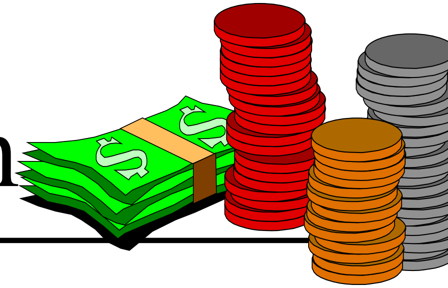
Implications of Scale-Free Networks



Business

- financial failures
  - understand how companies, industries and economies are inter-linked
  - monitor and avoid cascading financial failures
- marketing
  - study the spread of a contagion on a scale-free network
  - more efficient ways of propagating consumer buzz about new products

Automatic Cluster Detection

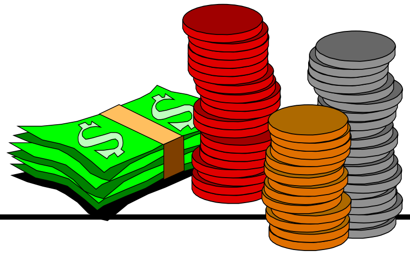


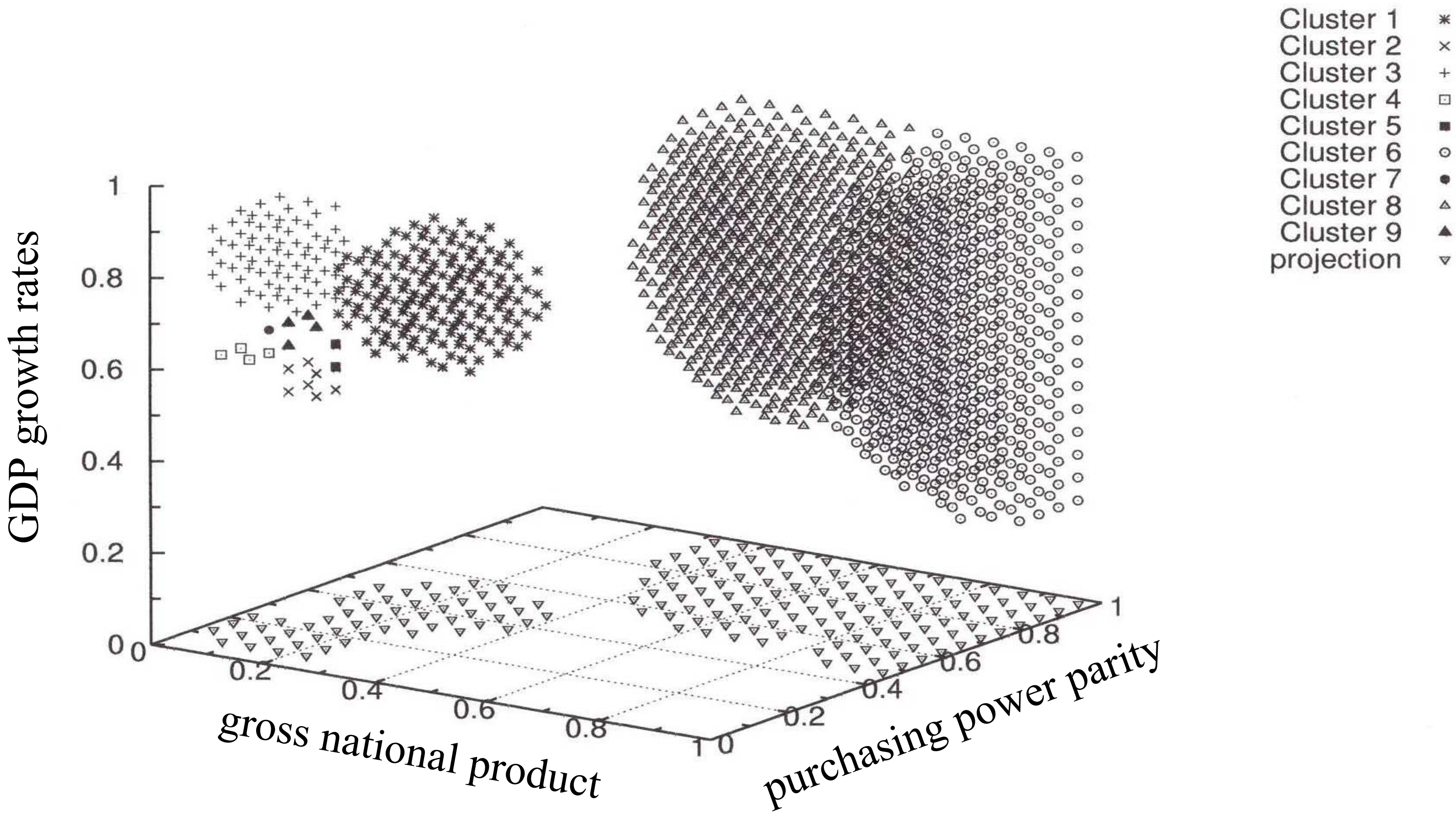
- Goal:

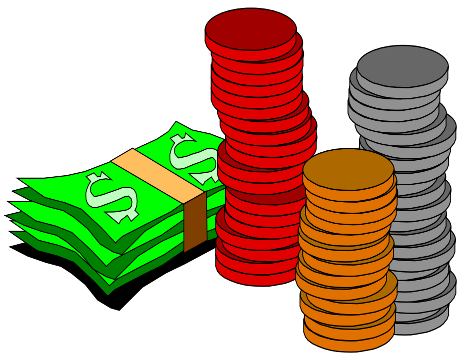
Find previously unknown similarities in the data!

- Build models that find data records similar to each other
- Good as an initial analysis of the data
- Undirected data mining

Economies grouped according to their results





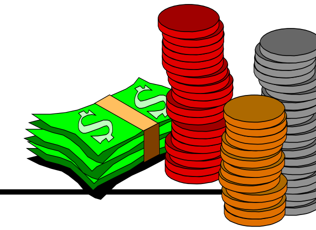


Mining the World Bank Data:
the Fuzzy c-means Clustering Approach

with Cihan H. Dagli,

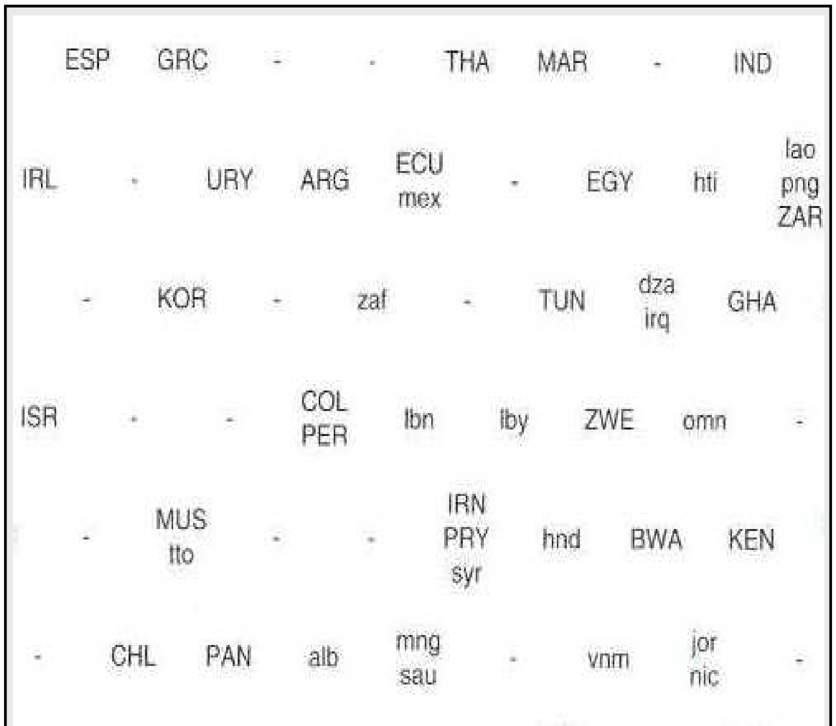
Engineering Management Department, University of Missouri - Rolla

FCM-clustering: introduction



- **World Development Indicators (WDI)**
  - published annually by the World Bank
  - reflect development process in the countries
  - incomplete and imprecise data
- **Previously applied techniques**
  - regression analysis - linear relationships
  - US-based grouping of countries (G. Ip, Wall Street Journal)
  - GDP-based grouping of economies (World Bank)
  - self-organizing feature maps (T. Kohonen, S. Kaski, G. Deboeck)

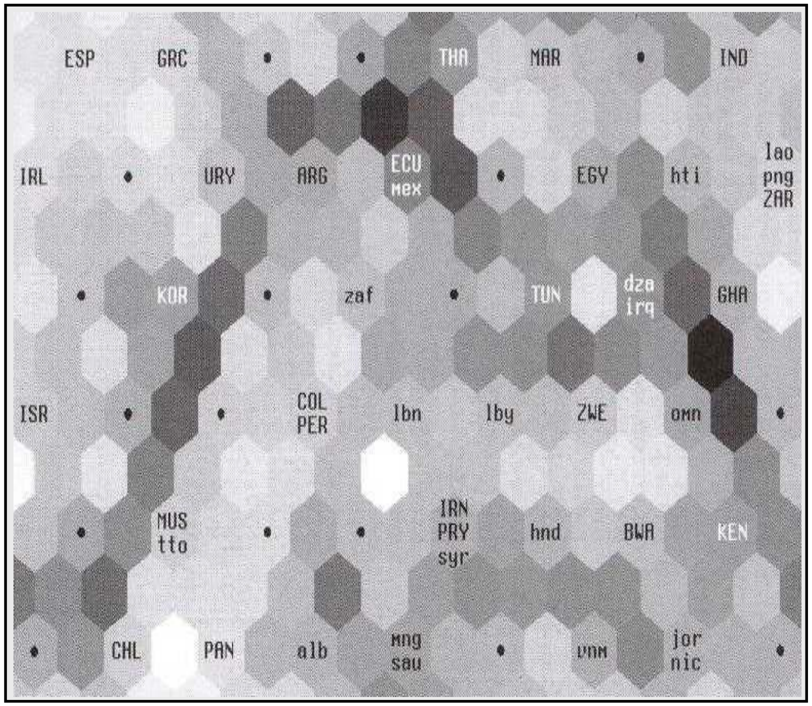
Poverty maps - T. Kohonen



adapted from "T. Kohonen: *Self-Organizing Maps*, 3-rd Edition, Springer-Verlag, 2001"

- more neurons than countries
- only local geometric relations are important
- countries mapped close to each other have a similar state of development

Poverty maps - T. Kohonen, S. Kaski

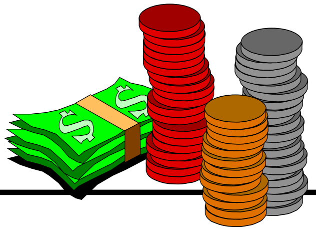


adapted from "T. Kohonen: *Self-Organizing Maps*, 3-rd Edition, Springer-Verlag, 2001"

U-matrix:

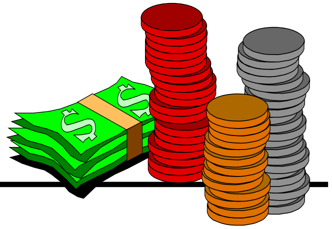
- illustrate "boundaries" between clusters
- represent average distances between neighboring neurons in a gray scale
  - small average distance
    $\Rightarrow$ ***light shade***
  - large average distance
    $\Rightarrow$ ***dark shade***

Our goal



- **Cluster efficiently imprecise data**
- **Estimate the number of clusters**
- **Visualize the results**
- **Interpret the results**

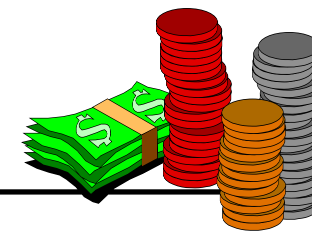
Our goal



- Cluster efficiently imprecise data
- Estimate the number of clusters
- Visualize the results
- Interpret the results

> - **fuzzy *c* - means clustering (FCM)**
> - **cluster validity indicators**
> - **spread-sheet-like form**
> - **find “landmarks”**

The objective function



- corresponds to the **weighted distance** between input patterns and cluster centers:

cluster center

$$J_s(\mathbf{U},\mathbf{v}) = \sum_{p=1}^{P}\sum_{i=1}^{c}(u_{ip})^s\left[\sum_{j=1}^{n}(x_{pj}-v_{ij})^2\right]$$

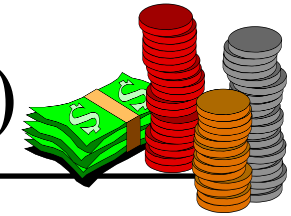
fuzziness parameter

membership degree

input pattern

- membership degrees between 0 and 1: $0 \le u_{ip} \le 1$
- total membership of a pattern equals to 1: $\forall p \mid \sum_{i=1}^{c} u_{ip} = 1$
- no empty or full clusters: $\forall i \mid 0 < \sum_{p=1}^{P} u_{ip} < P$

Fuzzy *c*-means Clustering (FCM)



- **Step 1:** Initialize $\boldsymbol{c}$, $\boldsymbol{s}$, ε and t. Choose randomly $U^{(0)}$.
- **Step 2:** Determine new fuzzy cluster centers:

$$\vec{v}_i^{\,(t)} = \frac{1}{\sum_p \left(u_{ip}^{(t)}\right)^s} \sum_p \left(u_{ip}^{(t)}\right)^s \vec{x}_p$$

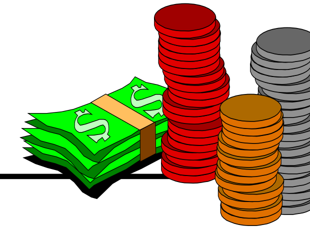
- **Step 3:** Calculate new partition matrix $U^{(t+1)}$:

$$u_{ip}^{(t+1)} = \frac{\left(1 / \left\| \vec{x}_p - \vec{v}_i^{\,(t)} \right\|^2\right)^{1/s-1}}{\sum_{k=1}^{c} \left(1 / \left\| \vec{x}_p - \vec{v}_k^{\,(t)} \right\|^2\right)^{1/s-1}}$$

- **Step 4:** Evaluate $\Delta = \left\| U^{(t+1)} - U^{(t)} \right\| = \max_{i,p} \left| u_{ip}^{(t+1)} - u_{ip}^{(t)} \right|$

  If $\Delta > \varepsilon$ then set $t = t + 1$ and go to **Step 2**. If $\Delta \le \varepsilon$ then **Stop**.
- **END of FCM**

Cluster validity criteria



- **Partition coefficient:**

clusters

$$F(U;c) = \frac{1}{P}\sum_{p=1}^{P}\sum_{i=1}^{c}(u_{ip})^2$$

membership degree

patterns

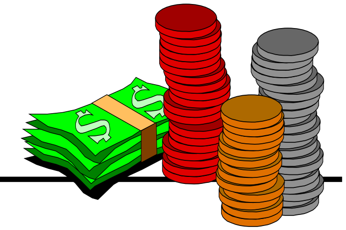
- **Partition entropy:**

$$H(U;c) = -\frac{1}{P}\sum_{p=1}^{P}\sum_{i=1}^{c} u_{ip}\ln(u_{ip})\,; \quad u_{ip}\ln(u_{ip}) = 0 \quad \text{for } u_{ip} = 0.$$

- **Windham´s proportion exponent:**

$$W(U;c) = -\sum_{p=1}^{P}\ln\left[\sum_{j=1}^{\lfloor \mu_p^{-1}\rfloor}(-1)^{j+1}\binom{c}{j}\left(1 - j\cdot\mu_p\right)^{c-1}\right]; \quad \mu_p = \max_{1\le i\le c}\left\{u_{ip}\right\}$$

How many clusters?



- **Partition coefficient:**

$$\max_{2\le c\le P-1}\left\{ \max_{U}\left[F(U;c) \right] \right\}$$

partitions clusters

- **Partition entropy:**

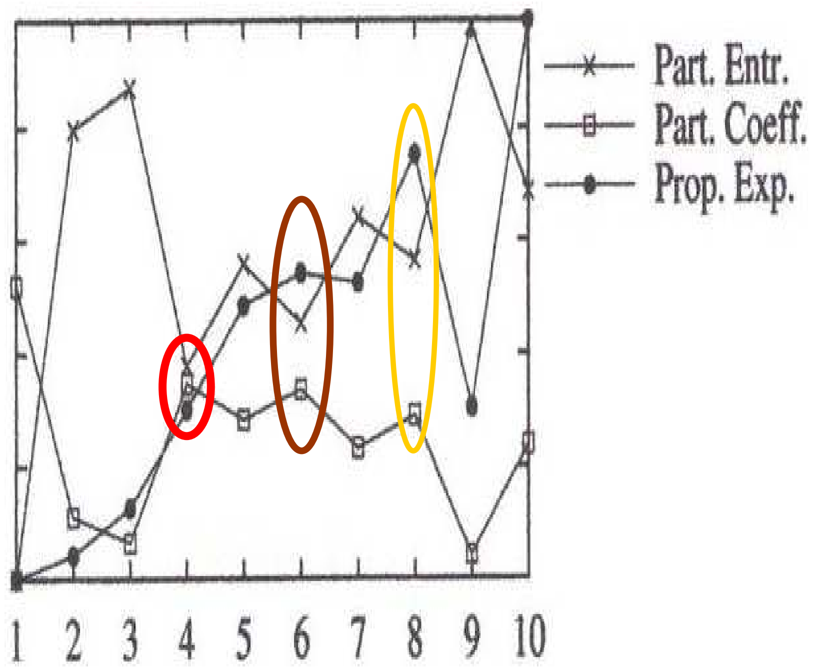
$$\min_{2\le c\le P-1}\left\{ \min_{U}\left[H(U;c) \right] \right\}$$

- **Windham´s proportion exponent:**

$$\max_{2\le c\le P-1}\left\{ \max_{U}\left[W(U;c) \right] \right\}$$

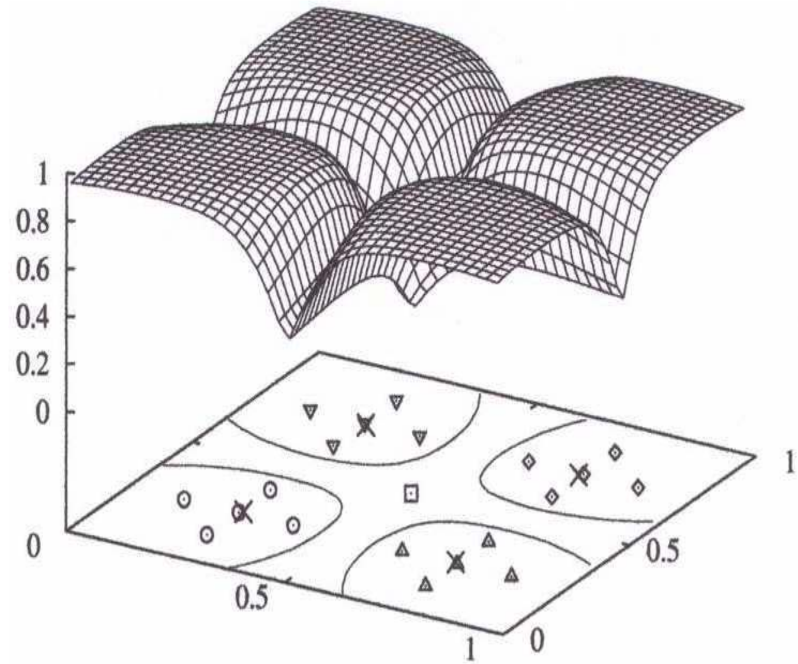
Supporting experiments - artificial data

Cluster validity indicators for artificial data



21 input patterns, $s = 1.4$, $\varepsilon = 0.05$

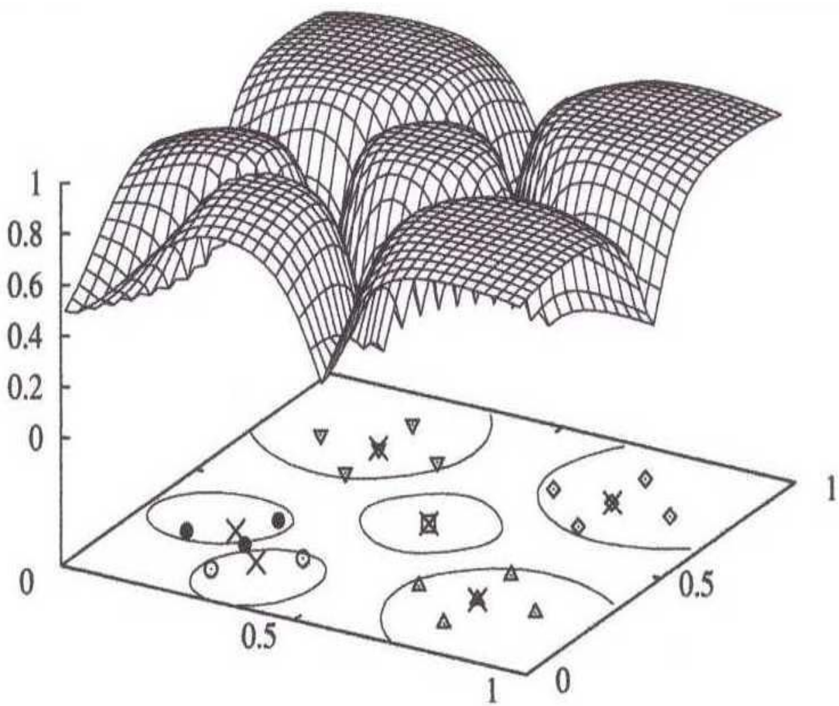
Fuzzy 4-partition of the data



´×´ indicates cluster centers, patterns from the same clusters are labeled identically

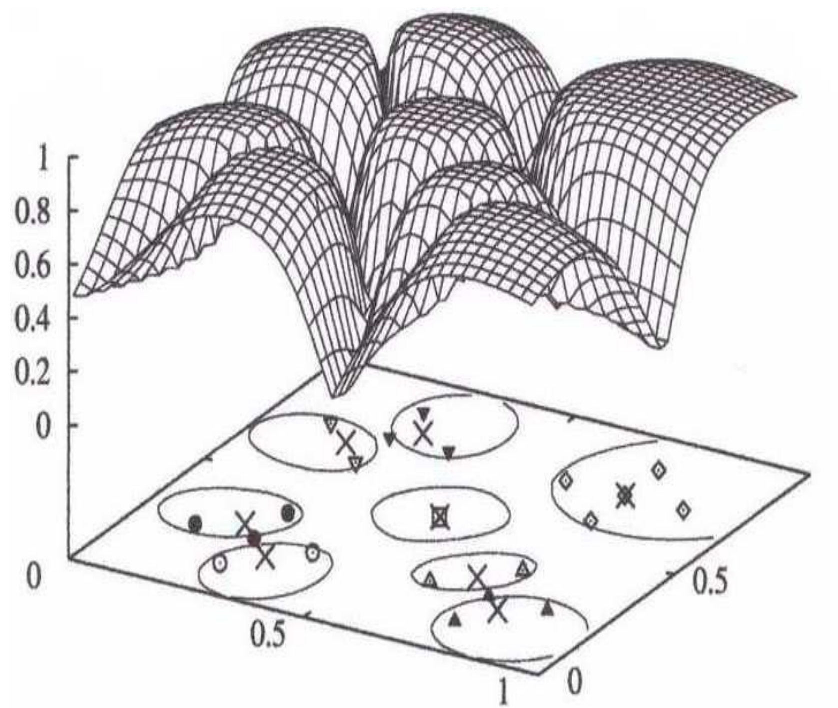
Supporting experiments - artificial data

Fuzzy 6-partition of the data



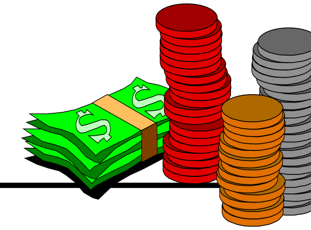
´×´ indicates cluster centers, patterns from the same clusters are labeled identically

Fuzzy 8-partition of the data



´×´ indicates cluster centers, patterns from the same clusters are labeled identically

Interpret the results!

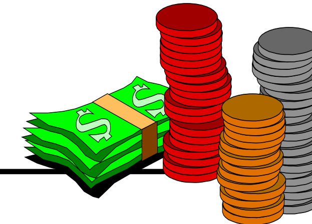


<u>Characteristic features for detected clusters:</u>

- cluster centers - *"fictive" patterns out of the data set*
- "calibrate" clusters with the "most representative" patterns from the data set - *based on just one pattern*
- **Determine outstanding properties for clusters:**
  - compared to other properties within the cluster
  - compared to properties of other clusters
  - exception: "border areas"

⇨ **fuzzy *c*-landmarks**

Automatic landmark selection



Fuzzy c-landmark for cluster i^*: $\left(j^*, v_{i^* j^*} \right)$

- "fuzzy distance" from the cluster center should be small
- "fuzzy distance" from all other cluster centers should be large

$$j^* = \arg\min_{1 \le j \le n} \frac{\dfrac{\sum_{p=1}^{P} u^{s}_{i^* p} \left| x_{pj} - v_{i^* j} \right|}{\sum_{p=1}^{P} u^{s}_{i^* p}}}{\min_{\substack{1 \le i \le c \\ i \ne i^*}} \dfrac{\sum_{p=1}^{P} u^{s}_{i^* p} \left| x_{pj} - v_{ij} \right|}{\sum_{p=1}^{P} u^{s}_{i^* p}}}$$

input patterns

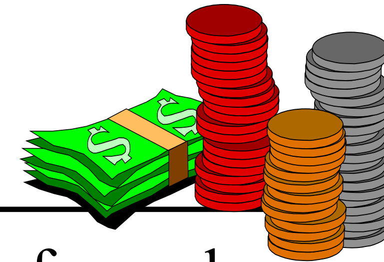
cluster centers

membership degrees

inputs

clusters

Supporting experiments: The World Bank Data



- 99 state economies with 16 (latest) indicators for each country
- economical and social potential of countries and their citizens
- all indicators are relative to population
- element-wise transformation to (0,1) with:

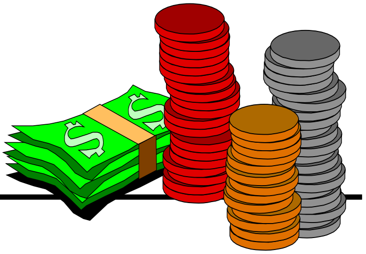
$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad \text{and} \quad x'' = \frac{1}{1 + e^{-k(x' - 1/2)}}$$

minimum over all patterns

maximum over all patterns

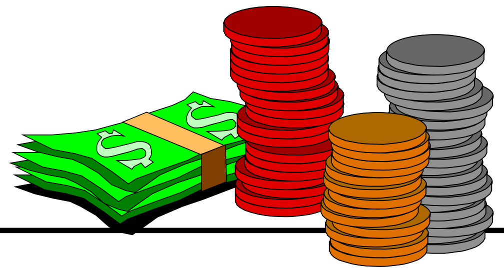
- the choice of other parameters (*k=4; s=1.4; ε=0.05*)

Used Development Indicators

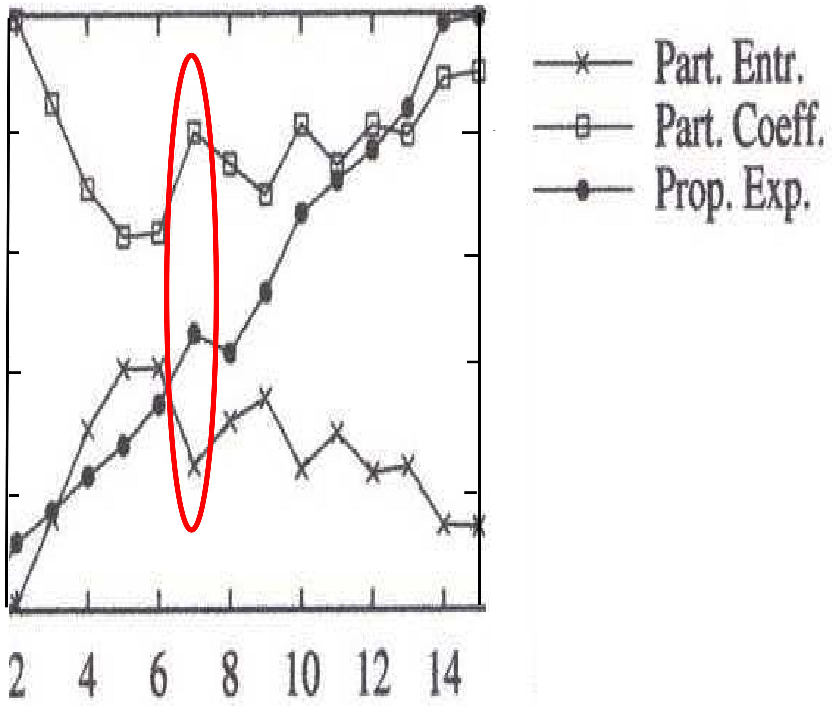


- GNP per capita
- Purchasing Power Parity
- Growth rate of GDP per capita
- GDP implicit deflator
- External debt (% of GNP)
- Total debt service (% of export of goods and services)
- High technology exports (% of manufactured exports)
- Military expenditures (% of GNP)
- Expenditures for R&D (% of GNP)
- Total expenditures on health (% of GDP)
- Public expenditures on education (% of GNP)
- Male life expectancy at birth
- Fertility rates
- GINI-index (distribution of income/consumption)
- Internet hosts per 10000 people
- Mobile phones per 1000 people

Supporting experiments: the World Bank data

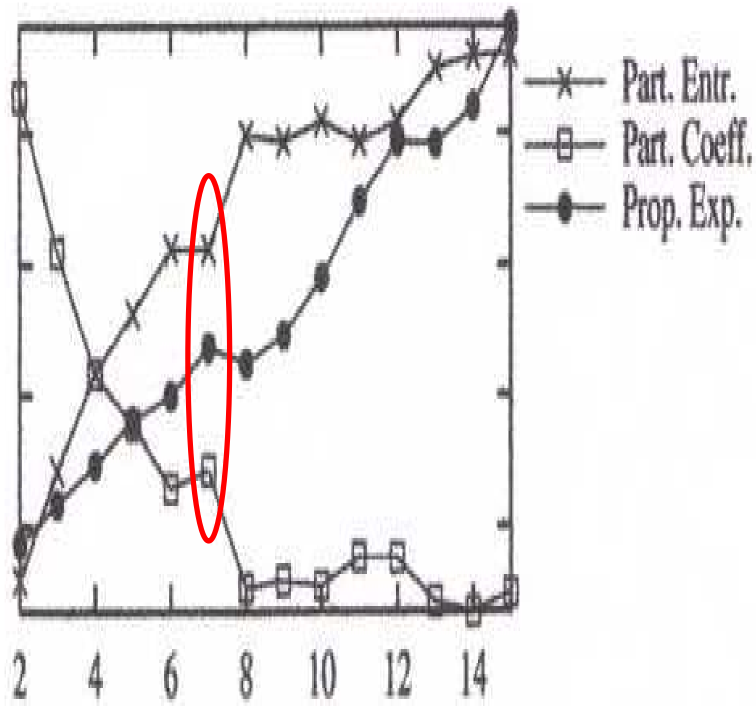


Cluster validity indicators for the WB-data



99 countries with 16 indicators
$s = 1.1$, $\varepsilon = 0.05$

Cluster validity indicators for the WB-data



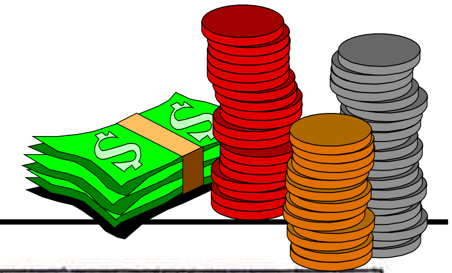
99 countries with 16 indicators
$s = 1.4$, $\varepsilon = 0.05$

Fuzzy 7-partition of the WB data

26	Egypt A.R.	0.02	0.90	0.01	0.05	0.01	0.00	0.01
27	El Salvador	0.01	0.08	0.01	0.11	0.01	0.00	0.78
28	Estonia	0.04	0.13	0.01	0.06	0.67	0.01	0.09
29	Ethiopia	0.00	0.00	0.97	0.02	0.00	0.00	0.00
30	Finland	0.00	0.00	0.00	0.00	0.02	0.98	0.00
31	France	0.00	0.00	0.00	0.00	0.01	0.99	0.00
32	Georgia	0.96	0.02	0.00	0.01	0.01	0.00	0.01
33	Germany	0.00	0.00	0.00	0.00	0.03	0.97	0.00
34	Ghana	0.00	0.07	0.08	0.82	0.00	0.00	0.02
35	Greece	0.01	0.05	0.00	0.02	0.85	0.04	0.03
36	Guatemala	0.01	0.09	0.18	0.37	0.01	0.00	0.34
37	Guinea	0.00	0.00	0.99	0.01	0.00	0.00	0.00
38	Honduras	0.01	0.03	0.02	0.09	0.01	0.00	0.86
39	Hungary	0.03	0.24	0.01	0.04	0.65	0.01	0.02
40	India	0.01	0.85	0.01	0.11	0.01	0.00	0.02
41	Indonesia	0.06	0.43	0.10	0.20	0.05	0.01	0.16
42	Ireland	0.01	0.02	0.01	0.01	0.13	0.79	0.02
43	Italy	0.00	0.00	0.00	0.00	0.01	0.99	0.00
44	Jamaica	0.07	0.47	0.01	0.14	0.10	0.00	0.22
45	Japan	0.00	0.00	0.00	0.00	0.01	0.98	0.00
46	Jordan	0.09	0.24	0.06	0.26	0.14	0.02	0.20
47	Kazakhstan	0.84	0.10	0.00	0.03	0.01	0.00	0.01
48	Kenya	0.01	0.04	0.19	0.67	0.01	0.00	0.07
49	Korea	0.04	0.09	0.02	0.05	0.38	0.38	0.05

A part of the fuzzy 7-partition of the World Bank data:
99 countries with 16 indicators; $s = 1.4$, $\varepsilon = 0.05$

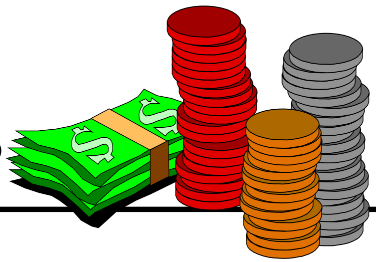
Landmarks for the WB data



No.	Representant	1. char. feature	2. char. feature	3. char. feature
1	Uzbekistan	GDP impl. defl. 330 % ann. growth	Hi-Tech exports 4 % of annual exp.	Gini-index 33.90
2	Vietnam	Fertility rate 2.57	Gini-index 36.73	Total exp. on health 4.94 % of GDP
3	Guinea	Internet hosts 0 per 10000 people	PPP per capita 1276 USD	GNP per capita 441.43 USD
4	Ghana	Fertility rate 3.94	Life exp. (males) 57.62 years	Gini-index 42.61
5	Slovenia	PPP per capita 13485 USD	Mobile phones 270 per 1000 people	Expend. on R&D 0.98 % of GNP
6	Netherlands	GDP impl. defl. 2.3 % of ann. growth	Ext. debt 1.1 % of GNP	Tot. debt serv. 0.47 % of export
7	Peru	Gini-index 48.98	GDP growth rate -1.92 % per capita	Life exp. (males) 66.95 years

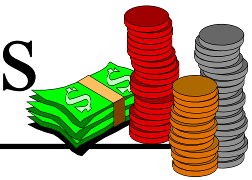
"Representative patterns" and fuzzy 7-landmarks for the World Bank data:
99 countries with 16 indicators; $s = 1.4$, $\varepsilon = 0.05$

FCM-Clustering: conclusions



- <u>FCM-clustering</u>
  - efficiency and cluster validity
  - choice of the fuzziness parameter
  - grouping of country economies (World Bank, Ip, Kohonen, Deboeck)
- <u>Visualization</u>
  - membership degree
  - topological relationships
- <u>Landmarks and interpretation of the results</u>
  - formulation of “class discriminating” criteria

From FCM towards Fuzzy Systems



- <u>Rule extraction:</u>

$$\boxed{\mathit{Characteristics} \Rightarrow \mathit{Economical\_results}}$$

  - fuzzy inference systems
  - (feed-forward) neural networks
    - back-propagation
    - RBF-networks
- <u>Neuro-fuzzy systems with adaptive inputs</u>
  - detection of significant input patterns
  - influence of internal knowledge representation
  - speed-up the training and recall process

Artificial Neural Networks (ANN)

Detect patterns in the data in a way "similar" to human thinking!

- Directed data mining (classification and prediction)
- Applicable also to undirected data mining (SOMs)
- <u>Two major drawbacks:</u>
  - difficulty in understanding the models they produce
  - sensitivity to the format of incoming data

Back-Propagation and GREN-networks

Introduction

- Multi-layer feed-forward networks (BP-networks)
  - one of the most often used models
  - relatively simple training algorithm
  - relatively good results
- Limits of the considered model
  - the speed of the training process
  - convergence and local minimums
  - generalization and “over-training”

⇨ **additional demands**
on the desired network behavior

The error function

- corresponds to the difference between the actual and the desired network output:

desired output

$$E = \frac{1}{2} \sum_{p} \sum_{j} \left(y_{j,p} - d_{j,p} \right)^2$$

patterns

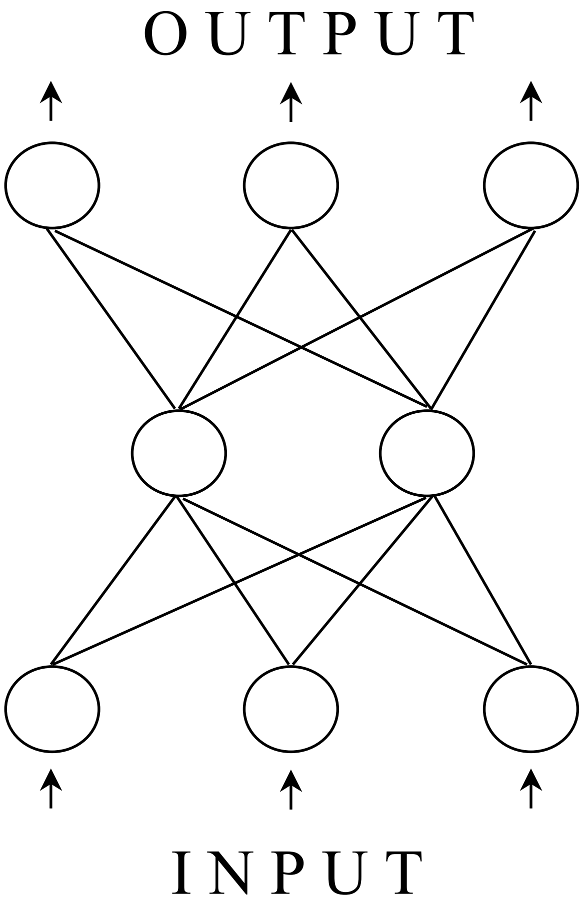
output neurons

actual output

- the goal of the training process is to minimize this difference on the given training set

⇨ **Back-Propagation training algorithm**

The Back-Propagation training algorithm



- computes the actual output for a given training pattern
- compares the desired and the actual output
- adapts the weights and the thresholds
  - against the gradient of the error function
  - backwards from the output layer towards the input layer

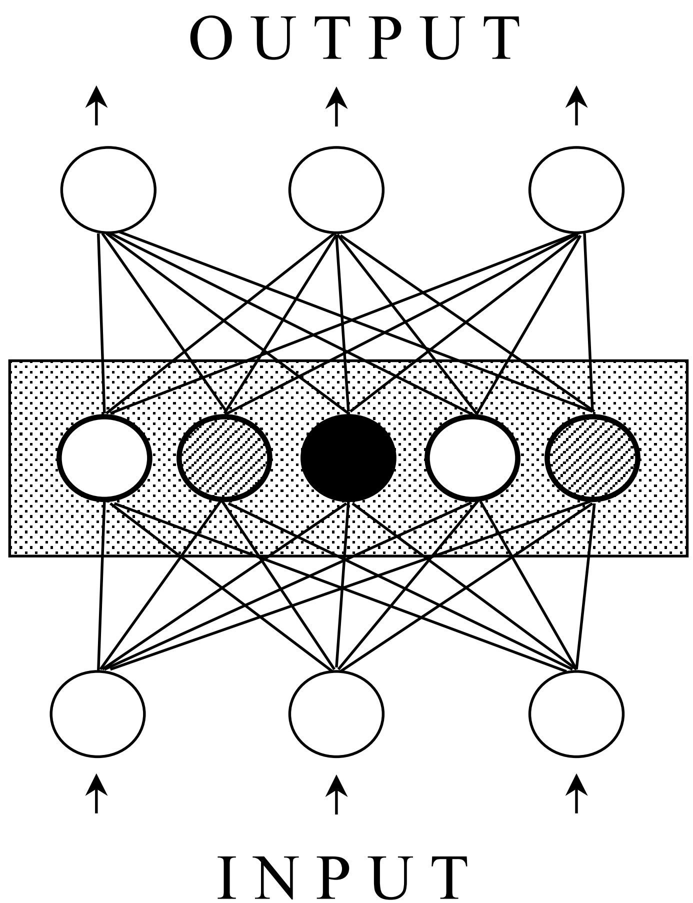
Drawbacks of the standard BP-model

- The error function
  - correspondence to the desired behavior
  - the form of the training set
    - requires the knowledge of desired network outputs
    - better performance for “larger” and “well-balanced” training sets
- Generalization abilities
  - ability to interpret and evaluate the “gained” experience
  - retraining for modified and/or developing task domains

Desired properties of trained networks

- Robustness against small deviations of those input patterns lying “close to the separating hyper-plane”
- Transparent network structure with a suitable internal knowledge representation
- A possible reuse of already trained networks under changed conditions

Condensed internal representation



- interpret the activity of hidden neurons:

  1 $\leftrightarrow$ active $\leftrightarrow$ YES

  0 $\leftrightarrow$ passive $\leftrightarrow$ NO

  $\frac{1}{2}$ $\leftrightarrow$ silent $\leftrightarrow$ $\leftrightarrow$ "no decision possible"

- "clear" the inner network structure
- detect superfluous neurons and prune

How to force the condensed internal representation?

- formulate "the desired properties" in the form of an objective function:

Standard error function

$$G = E + c_s F$$

Representation error function

the influence of F on G

- local minima of the representation error function correspond to active, passive and silent states:

$$F = \sum_p \sum_h y_{h,p}^s \left(1 - y_{h,p}\right)^s \left(y_{h,p} - 0.5\right)^2$$

patterns

hidden neurons

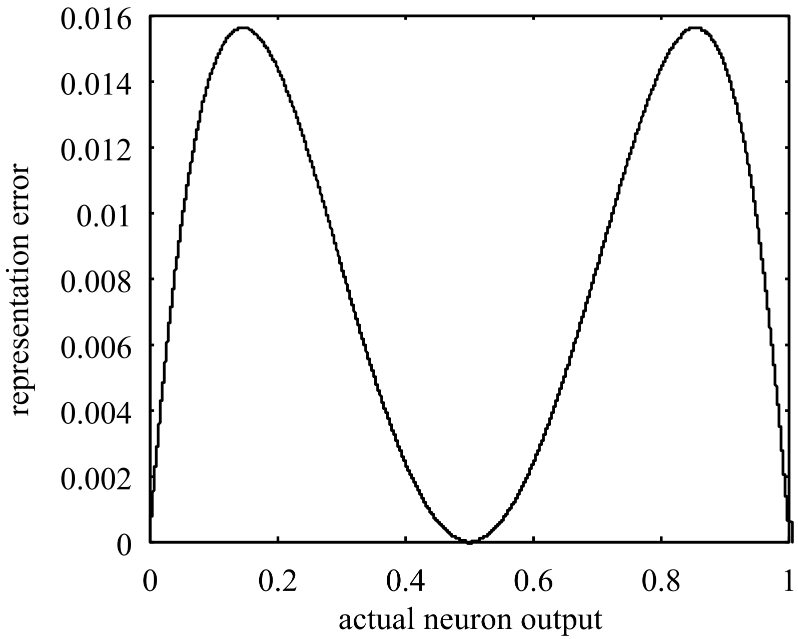
passive state

the shape of F

silent state

active state

Influence of parameters

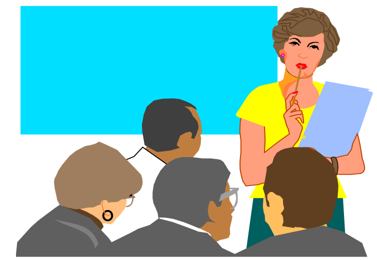


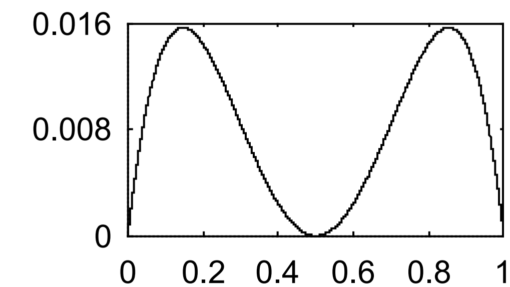
$$w_{ij}(t+1) = w_{ij}(t) + \alpha\delta_j y_i + \alpha_r \rho_j y_i + \\ + \alpha_m \big(w_{ij}(t) - w_{ij}(t-1)\big)$$

- slower forcing of the internal representation and the desired network function
- stability of the forced internal representation and an optimal network architecture
- the shape of the representation error function, the speed of the representation forcing process and its form
- the time-overhead of the weight adjustment

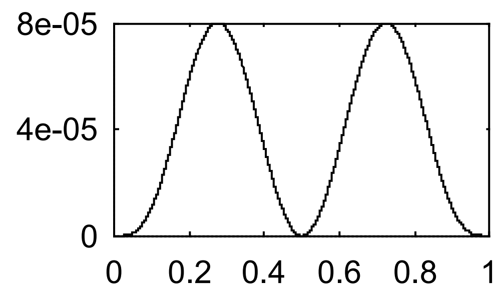
Shape of the representation function

$$F = y^s(1-y)^s(y-0.5)^t$$

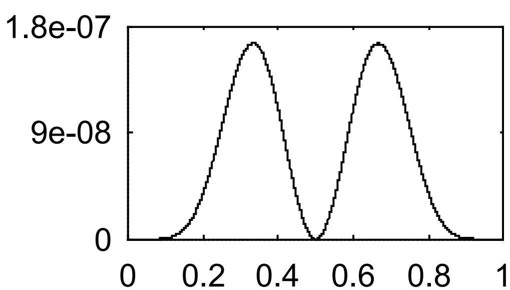




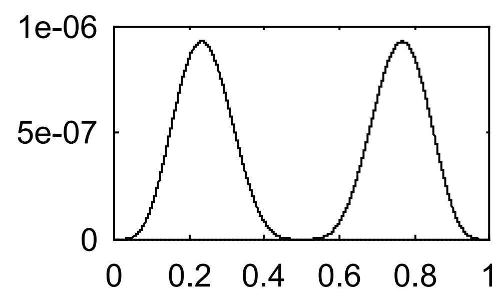
s=1 t=2



s=4 t=2



s=8 t=2



s=5 t=4

Further modifications of the representation function

- Discrete internal representation:

  (S allowed output values $r_1, \ldots, r_S$ for neurons from the last hidden layer)

$$F = \sum_p \sum_j \left(y_{j,p} - r_1\right)^{2t_1} \ldots \left(y_{j,p} - r_S\right)^{2t_S} = \sum_p \sum_j \prod_s \left(y_{j,p} - r_s\right)^{2t_s}$$

- Condensed internal representation for all hidden layers:

$$F = \sum_{l'} \sum_p \sum_{j_{l'}} y_{j_{l'},p}^{s} \left(1 - y_{j_{l'},p}\right)^{s} \left(y_{j_{l'},p} - 0.5\right)^{2}$$

Unambiguous internal representation

- Patterns with highly different outputs should form highly different internal representations
- Formulate the requirements as a modified objective function: $G = E + F + H$
- Ambiguity criterion for the internal representation:

$$H = -\frac{1}{2}\sum_{p}\sum_{q \neq p}\sum_{j}\sum_{o}\left(d_{o,p} - d_{o,q}\right)^2\left(y_{j,p} - y_{j,q}\right)^2$$

patterns

hidden neurons

output neurons

= const. for a given p

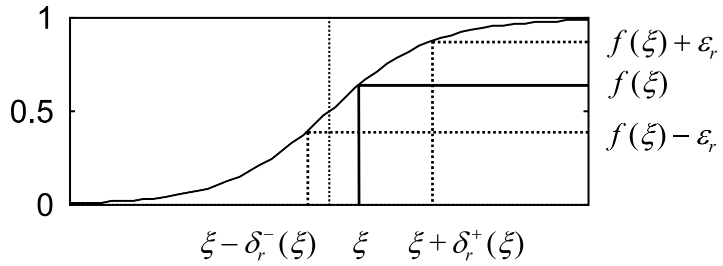
= const. for a given p

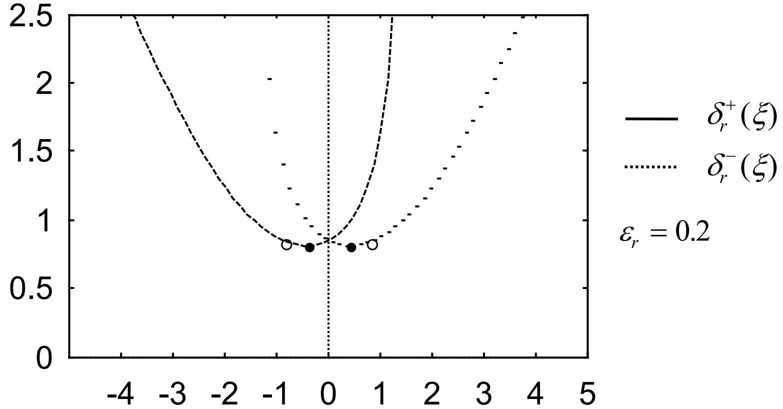
= const. for a given p

Modular structure of BP-networks

- Decompose the task into the particular subtasks
- Propose and form the modular architecture
  - strategy for extracting ε-equivalent BP-modules
    - elimination of superfluous hidden and/or input neurons
    - suitable for "already trained" networks
    - a compromise between the desired accuracy of the extracted module and its optimal architecture
- Communication between the particular modules
  - serial and parallel composition of BP-networks

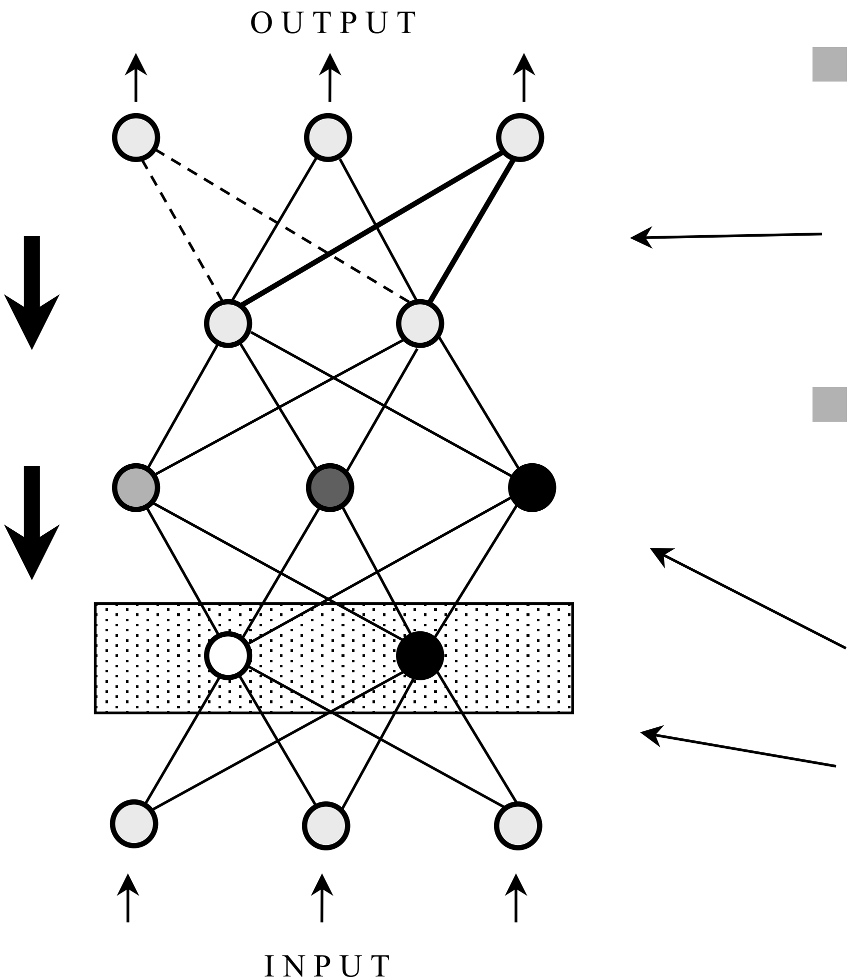
Extracting BP-modules
- allowed potential deviations





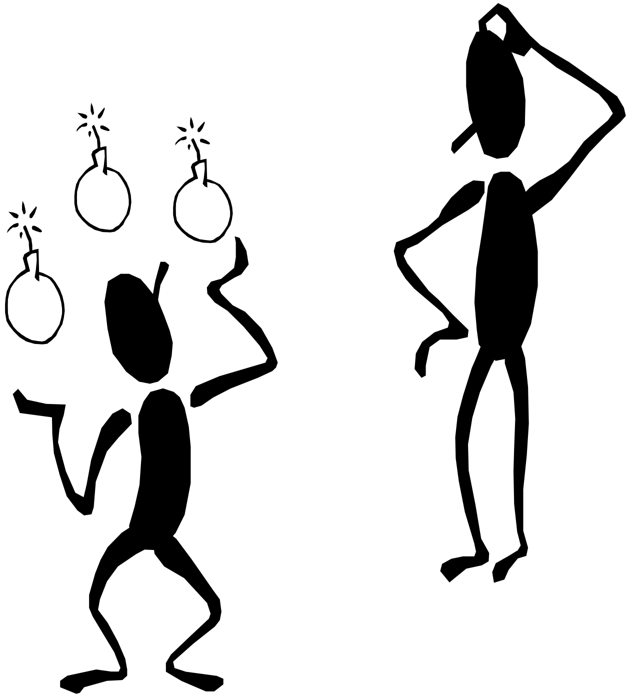
- The potential change $\delta_r^-(\xi)$ is in this case smaller than the potential change $\delta_r^+(\xi)$
- The potential should change "towards the separating hyper-plane"
- The changed potential should preserve the location of the input pattern in the same half-space
- The allowed potential changes should be independent of each particular input pattern (from S)

Notes on the construction of an ε-equivalent network



- possible improvements of network properties:
  - "egalitarian" versus "differentiated" approach
- the relationship of the construction to "more robust" networks
  - necessary knowledge of ε_r-boundary regions
  - preserve the created internal representation

Desired properties of “experts” for training (modular) BP-networks



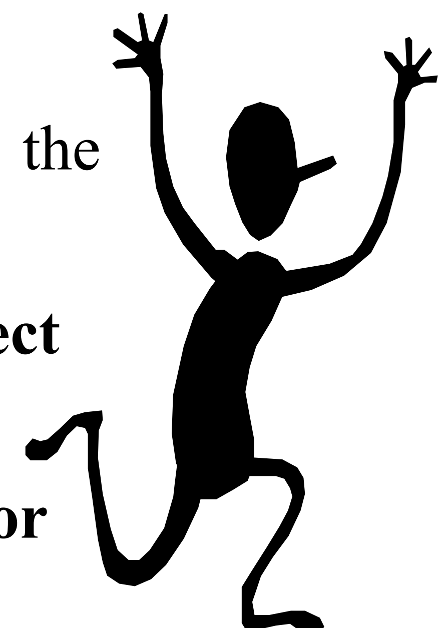
- **evaluate the error connected with the actual response of a BP-network**
- **“explain” the BP-network its error during training**
- not require the knowledge of the desired network output
- but should recognize a correct behavior
- “suggest” a “better” behavior

Desired properties of "experts" for training BP-networks

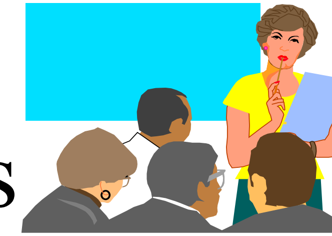
- evaluate the error connected with the actual response of a BP-network
- "explain" the BP-network its error during training
- **not require the knowledge of the desired network output**
- but should recognize a correct behavior
- "suggest" a "better" behavior

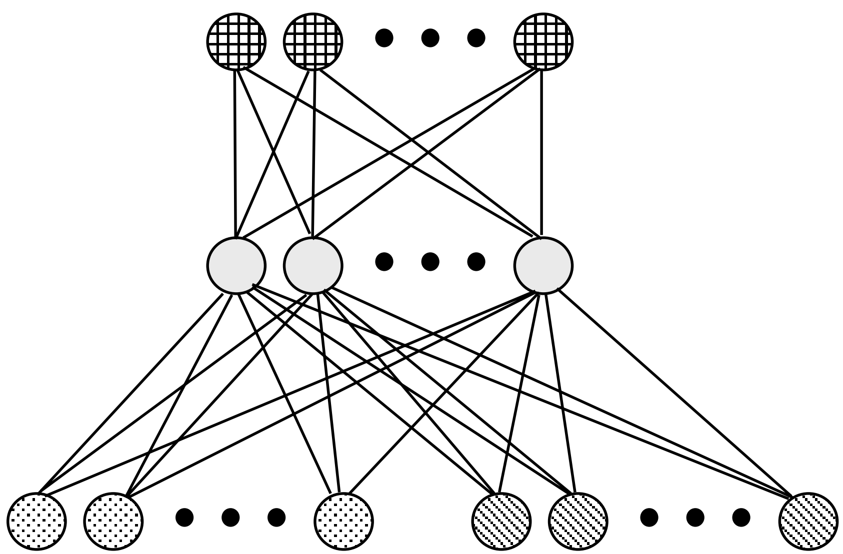
Desired properties of “experts” for training BP-networks

- evaluate the error connected with the actual response of a BP-network
- “explain” the BP-network its error during training
- not require the knowledge of the desired network output
- **but should recognize a correct behavior**
- **“suggest” a “better” behavior**



GREN-networks: Generalized relief error networks





- assign the error to the pairs *[input pattern, actual output]*
- trained e.g. by the standard BP-training algorithm
- should have good approximation and generalization abilities
- "approximates" the error function by:

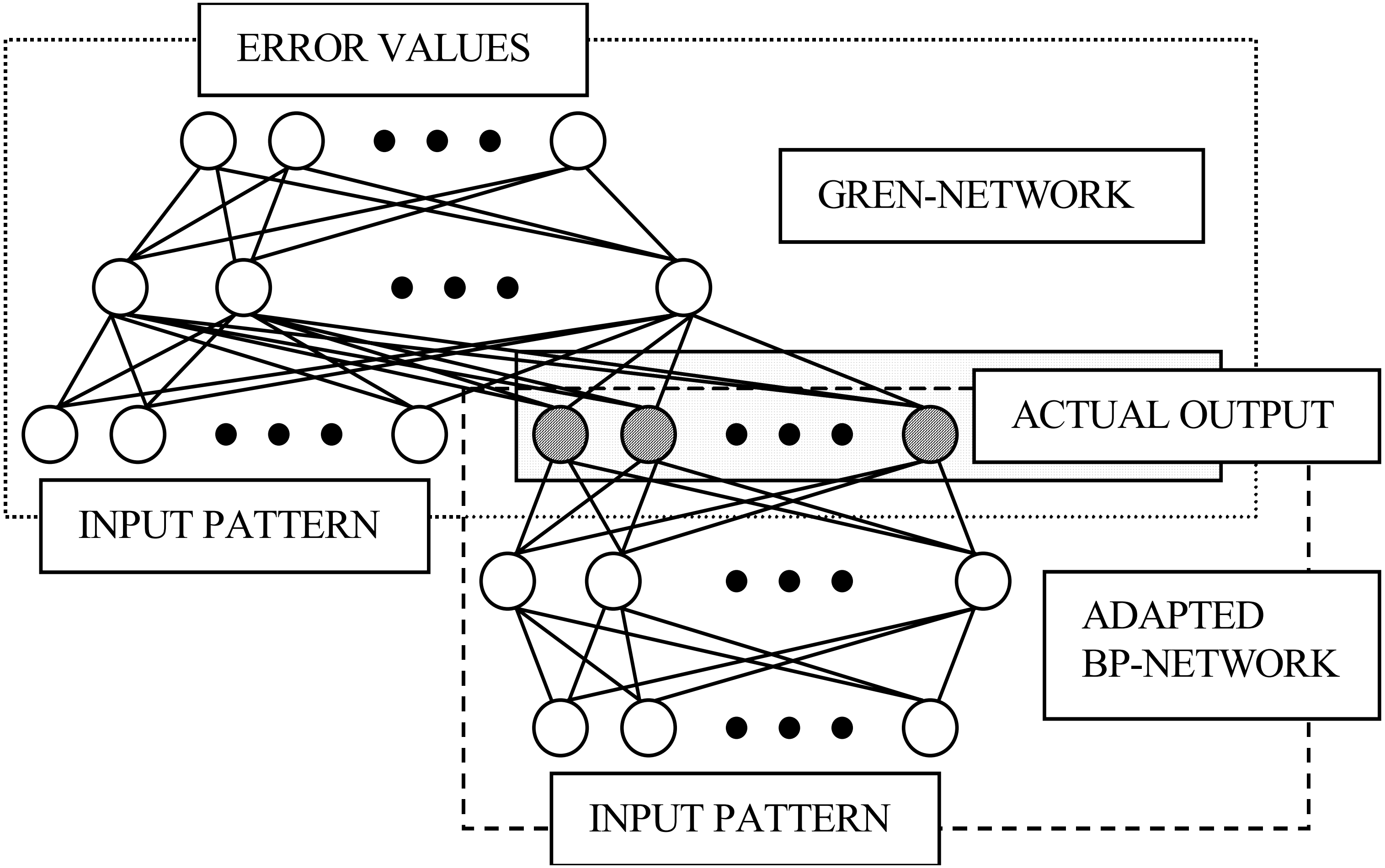
$$E = \sum_{p} \sum_{e} e_{e,p}^{GR_B}$$

patterns

output neurons of the GREN-network

output values of the GREN-network

A modular system for training BP-networks with GREN-networks



Training with a GREN-network

- Applied the basic idea of Back-Propagation
- **How to determine the error terms at the output of the trained BP-network?**

⇨ **Use error terms back-propagated from the GREN-network**

- Weight adjustment rules similar to the standard Back-Propagation

Training with a GREN-network

- Applied the basic idea of Back-Propagation

error computed by the GREN-network

$$\Delta_E w_{ij}^B = -\frac{\partial E}{\partial w_{ij}^B} = -\frac{\partial E}{\partial y_j^B}\frac{\partial y_j^B}{\partial \xi_j^B}\frac{\partial \xi_j^B}{\partial w_{ij}^B}$$

potential of neuron j

actual output

weight of a BP-network B

- **How to determine $\partial E / \partial y_j^B$ at the output layer of the BP-network *B*?**

Weight adjustment rules

- **Use error terms back-propagated from the GREN-network**
- Rules similar to the standard Back-Propagation

$$w_{ij}^{B}(new) = w_{ij}^{B}(old) + \alpha \delta_{j}^{B} y_{i}^{B}$$

weight

learning rate

error term

actual output of neuron *i*

- For output neurons, compute δ_j^B by means of $\delta_k^{GR_B}$ propagated from the GREN-network GR_B

$$\delta_{k}^{GR_B} = -\frac{\partial E}{\partial y_{k}^{GR_B}} \frac{\partial y_{k}^{GR_B}}{\partial \xi_{k}^{GR_B}}$$

error

actual output of neuron *k*

error term

potential of neuron *k*

Error terms for the trained BP-network

- The back-propagated error terms δ_j^B correspond for $E = \sum_e e_e$ to:

$$
\delta_j^B = \begin{cases}
-\left(\sum_e e_e^{GR_B} \left(1 - e_e^{GR_B}\right) w_{je}^{GR_B} \right) f'(\xi_j^B) & \text{for an output neuron of } B \text{ and } GR_B \text{ with no hidden layer} \\
-\left(\sum_k \delta_k^{GR_B} w_{jk}^{GR_B} \right) f'(\xi_j^B) & \text{for an output neuron of } B \text{ and } GR_B \text{ with hidden layers} \\
\left(\sum_k \delta_k^B w_{jk}^B \right) f'(\xi_j^B) & \text{for a hidden neuron of } B
\end{cases}
$$

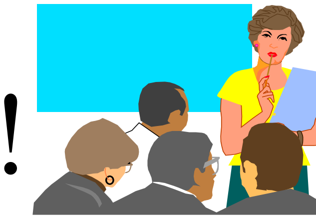
Is the GREN-network an "expert"?

- Has not to "know the right answer"
- But should "recognize the correct answer"

for an input pattern, yield the <u>minimum error</u> only <u>for one</u> actual <u>output</u> - the right one

- Simple test for "problematic" GREN-experts:
  - zero-weights from the actual output y^B
  - zero "y-terms" of potentials in the 1. hidden layer
  - "too many large negative weights" $\sum_i |w_i^-| >> \sum_i w_i^+$

Find “better” input patterns!



- input patterns of a GREN-network
- “similar” to those presented to and recalled by the BP-network
- with a smaller error

- **minimize** the **error** at the output of the GREN-network, e.g. **by back-propagation**
- **adjust input patterns** against the gradient of the GREN-network error function

Avoid "problematic" GREN-networks!

- Insensitive to the outputs of trained BP-networks
  - inadequately small error terms back-propagated by the GREN-network
- Incapable of training further BP-networks
  - small error terms even for large errors

Our goal:

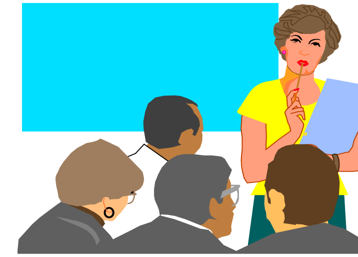
Increase the sensitivity of GREN-networks to their inputs!

How to handle the sensitivity of BP-networks?

Increase their robustness:

- over-fitting leads to functions with a lot of structure and a relatively high curvature
- favor “smoother” network functions
- alternative formulation of the objective function
  - penalizing large second-order derivatives of the network function
  - penalizing large second-order derivatives of the transfer function for hidden neurons
  - weight-decay regularizers

Controlled learning of GREN-networks



- Require GREN-networks sensitive to their inputs
  - non-zero error terms for incorrect BP-network outputs
- Favor larger values of the error terms

Minimize during training

output values

$$E^{REG} = \sum_q E_q^{REG} = -\sum_q \sum_s \sum_{r>n} \left(\frac{\partial y_{s,q}}{\partial y_{r,q}} \right)^2$$

patterns

output neurons

controlled input neurons

controlled input values

Weight adjustment rules

- Regularization by means of

output

$$\Delta_{E^{REG}} w_{ij} = -\frac{\partial E^{REG}}{\partial w_{ij}} = -\frac{\partial}{\partial w_{ij}}\left(-\sum_{s}\sum_{r>n}\left(\frac{\partial y_s}{\partial y_r}\right)^2\right)$$

weight

controlled inputs

- Rules similar to the standard Back-Propagation

controlled weight adjustment

$$w_{ij}^{GR_B}(T+1) = w_{ij}^{GR_B}(T) + \alpha\,\Delta_E w_{ij} + \alpha_c\,\Delta_{E^{REG}} w_{ij} + $$
$$+\alpha_m\left(w_{ij}^{GR_B}(T) - w_{ij}^{GR_B}(T-1)\right)$$

BP-weight adjustment

moment

Characteristics of the method

- Applicable to any BP-network and/or input neuron
- Quicker training of “actual” BP-networks
  - larger “sensitivity terms” $\partial y_s / \partial y_r$ transfer better the errors from the GREN-network
- Oscillations during training “actual” BP-networks
  - due to the “linear” nature of the GREN-specified error function

$$E = \sum_{p} \sum_{e} e_{e,p}^{GR_B}$$

patterns

output neurons of the GREN-network

output values of the GREN-network

Modification of the method

- Use "quadratic" GREN-specified error terms for training "actual" BP-networks

$$\hat{E} = \sum_{p} \sum_{e} \left(e_{e,p}^{GR_B} \right)^2$$

patterns

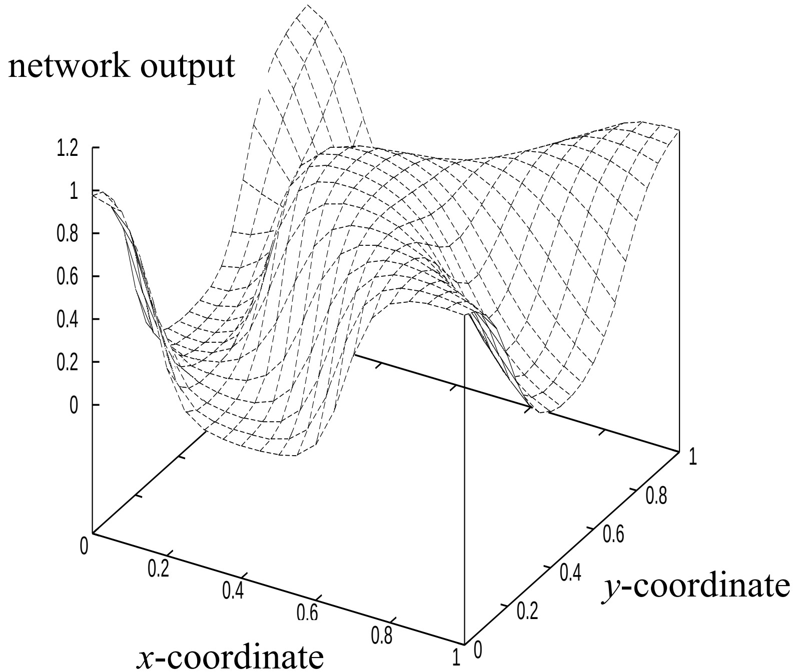
output neurons of the GREN-network

output values of the GREN-network

- Considers both the GREN-network outputs $e_{e,p}^{GR_B}$ and the "sensitivity" terms $\partial e_{e,p}^{GR_B} / \partial y_{j,p}^{B}$
- Crucial for low sensitivity to erroneous training patterns

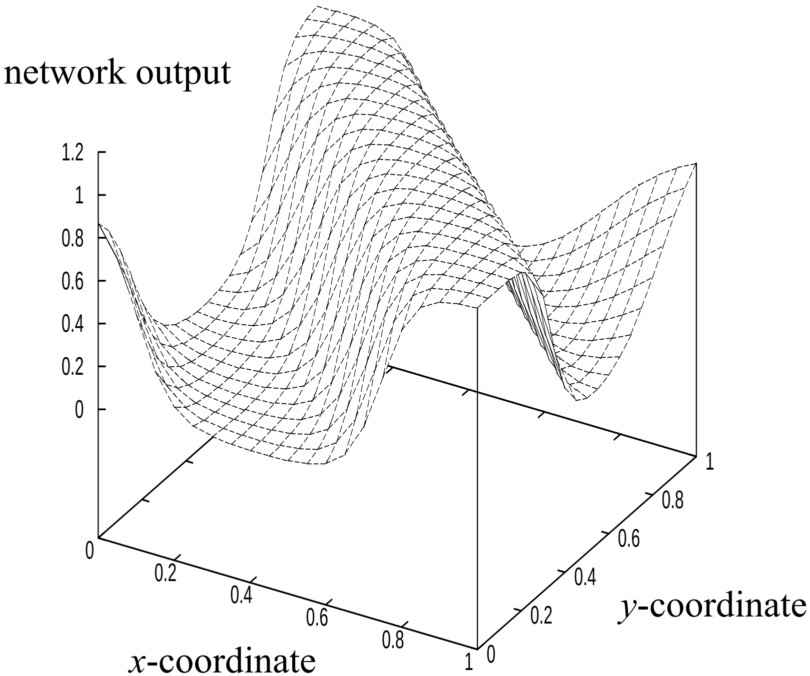
Supporting experiments

Output of the standard BP-network



3000 cycles, SSE = 0.89

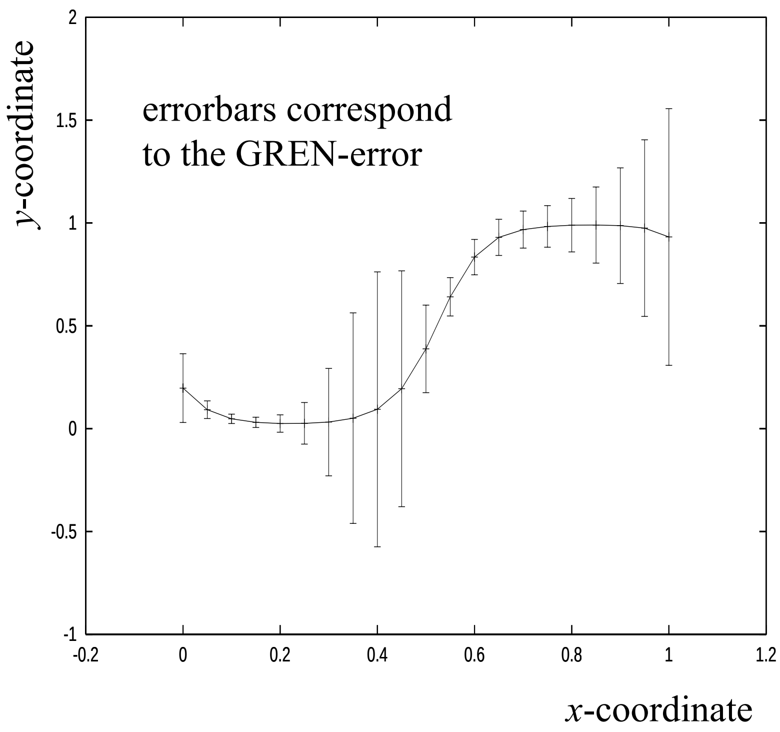
Output of the GREN-trained BP-network



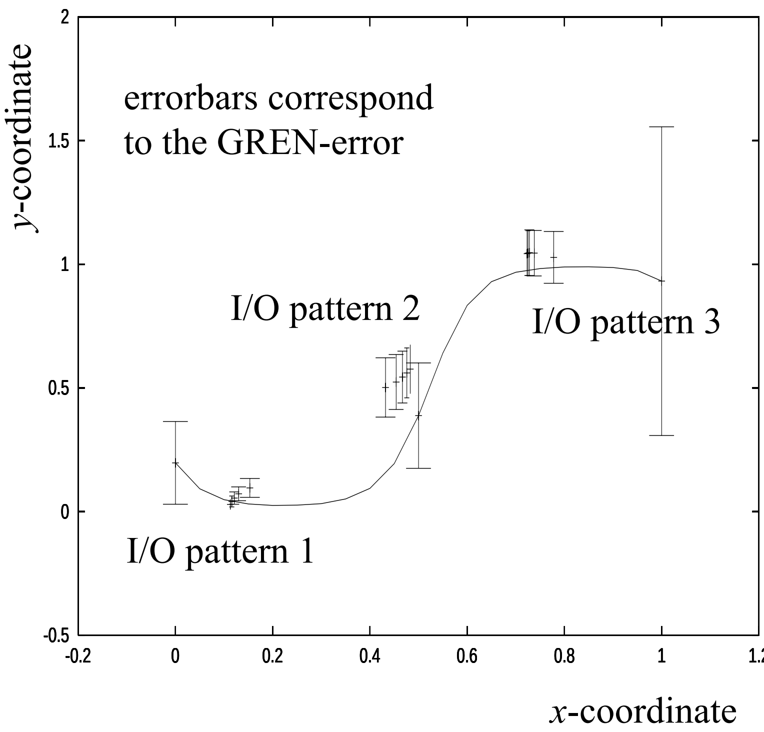
3000 cycles, SSE = 0.05

Supporting experiments

BP-network output
for a constant *y=0.25*

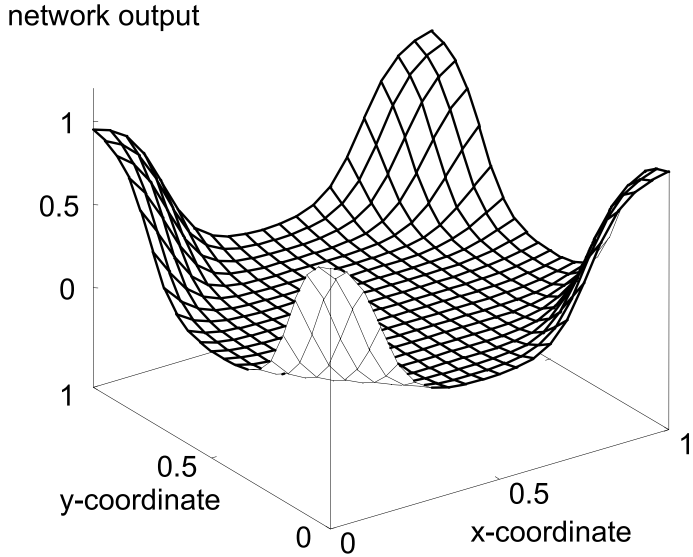


GREN-adjusted input/output patterns
for a constant *y=0.25*



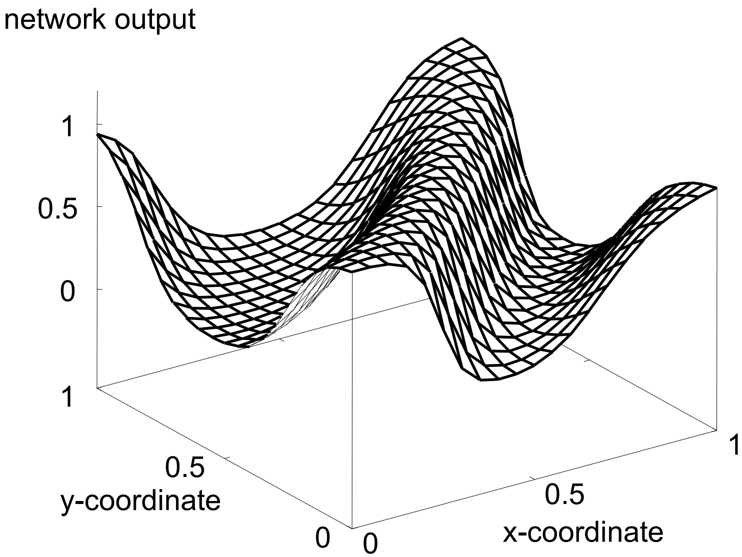
Supporting experiments

Output of the standard BP-network (with 8 hidden neurons)



1500 cycles, SSE = 0.51

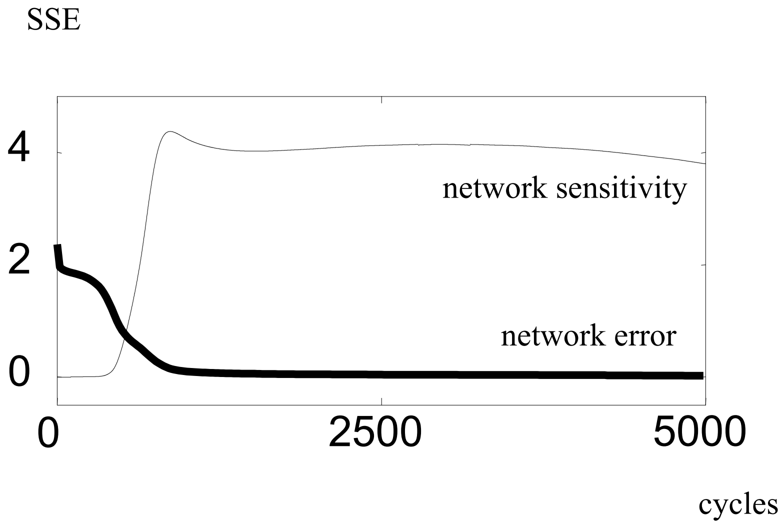
Output of the GREN-trained BP-network (with 8 hidden neurons)



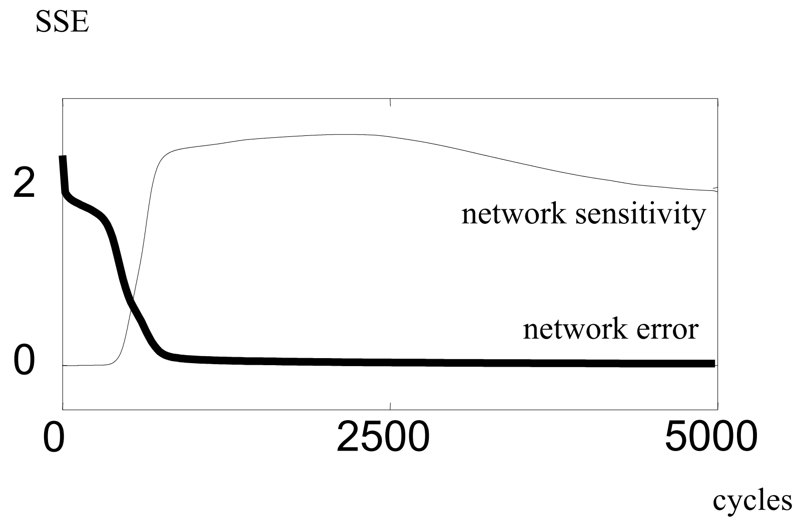
1500 cycles, SSE = 0.06, GREN-error = 1.2

Supporting experiments

Sensitivity and error for a standard BP-trained GREN-network

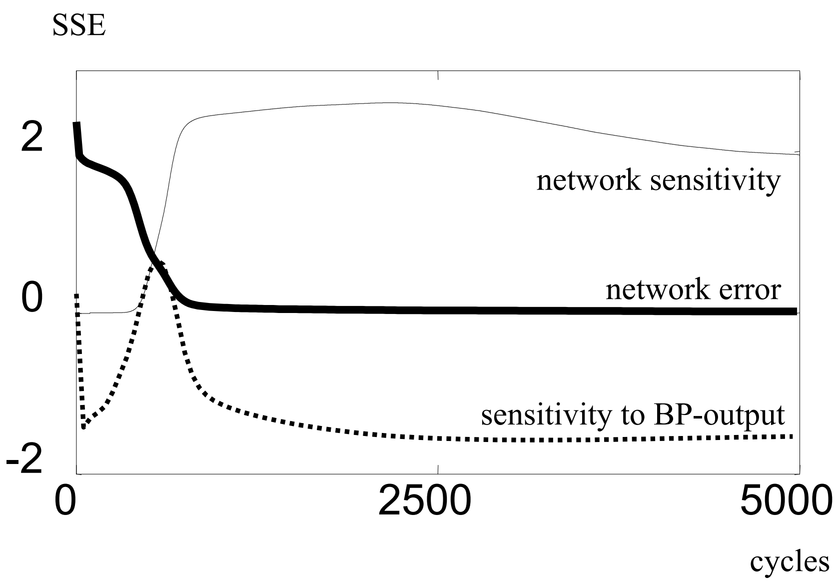


Sensitivity and error for a controlled-trained GREN-network (control rates = 0.2)

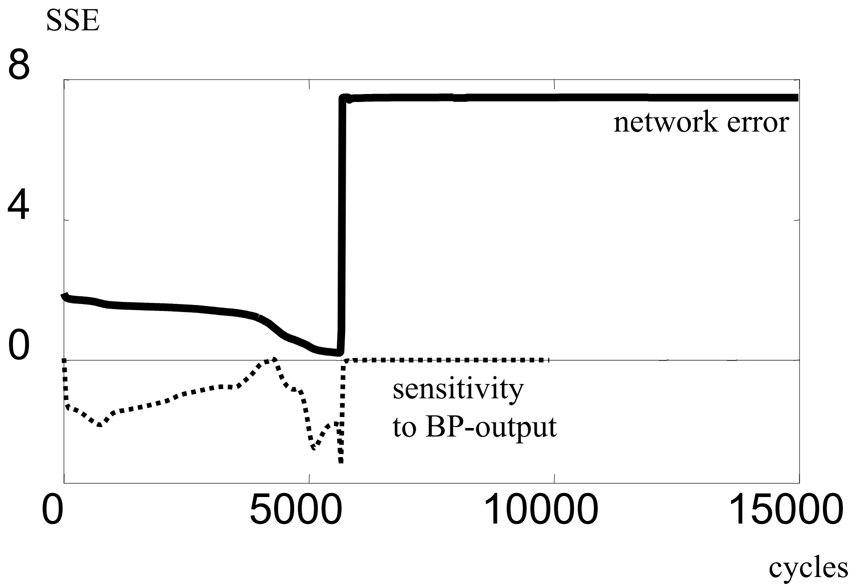


Supporting experiments

Sensitivities and error for a controlled-trained GREN-network (control rates = 0.2)



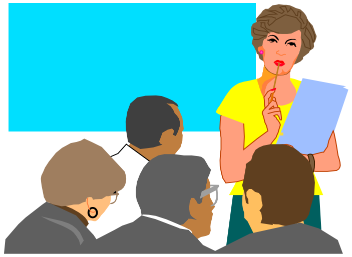
Sensitivities and error for an over-trained GREN-network (control rates = 0.2)



GREN-networks: conclusions

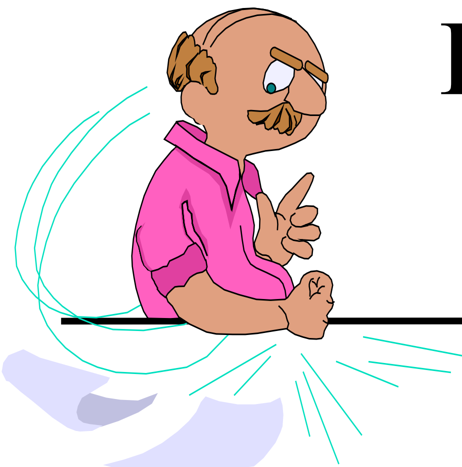
- GREN-networks can train BP-networks without the knowledge of their desired outputs
- A simple detection of “problematic” GREN-experts
- GREN-networks can find “similar” input patterns with a lower error

Conclusions: Sensitivity of GREN-networks



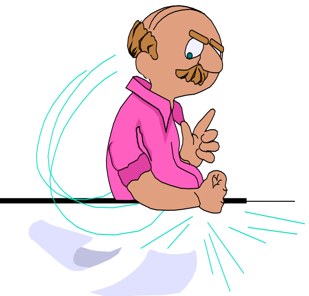
- Increase the sensitivity of trained GREN-networks to their inputs
- Detect “over-training” in GREN-networks
- Train BP-networks more efficiently by minimizing squared GREN-network outputs instead of the “linear” ones
- <u>Further research:</u> simplified sensitivity control

Acoustic Emission and Feature Selection Based on Sensitivity Analysis



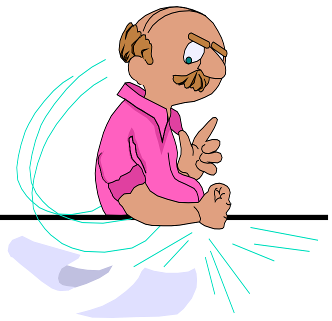
with M. Chlada and Z. Převorovský,
Institute of Thermomechanics, Academy of Science

Acoustic Emission and Feature Selection Based on Sensitivity Analysis

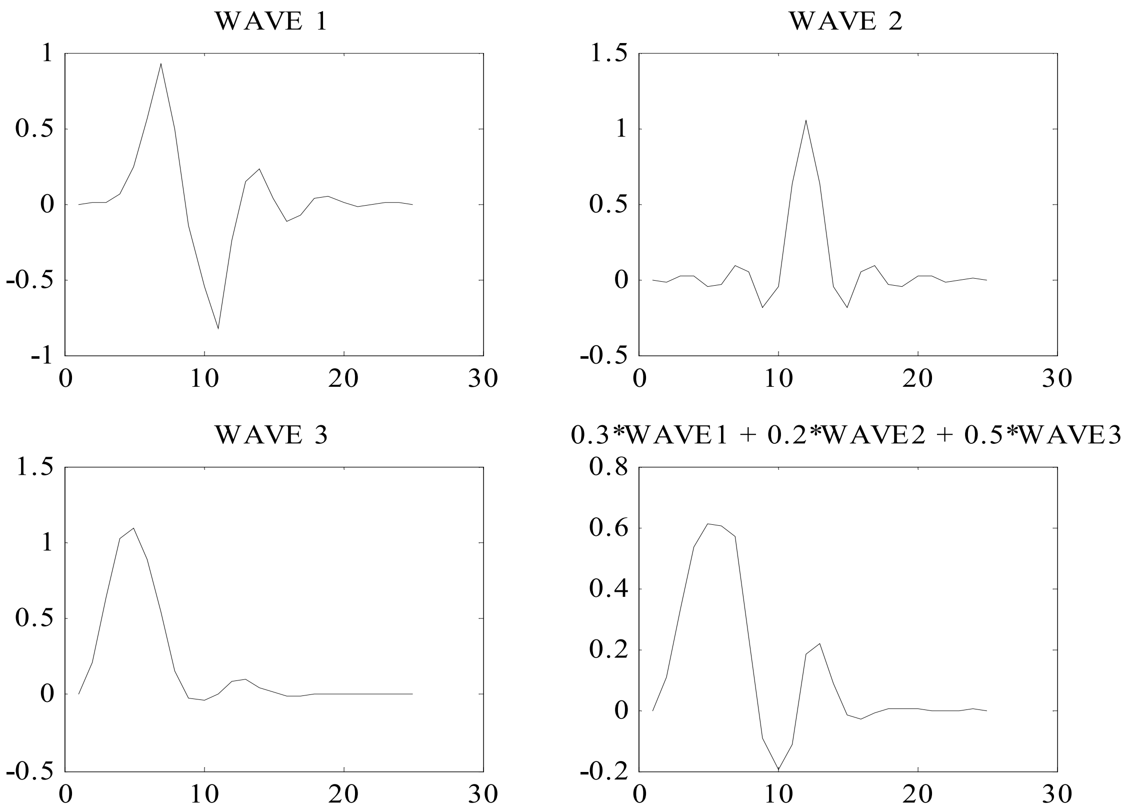


- BP-networks and sensitivity analysis
  - larger "sensitivity terms" $|\partial y_j / \partial x_i|$ indicate higher importance of the feature i
- numerical experiments
  - *acoustic emission:*
    - classification of simulated AE data
  - *feature selection:*
    - reduction of original input parameters (from 14 to 6)
    - model dependence between parameters

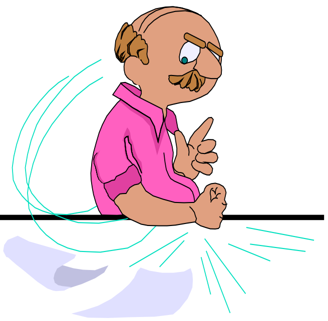
Simulation of AE-data



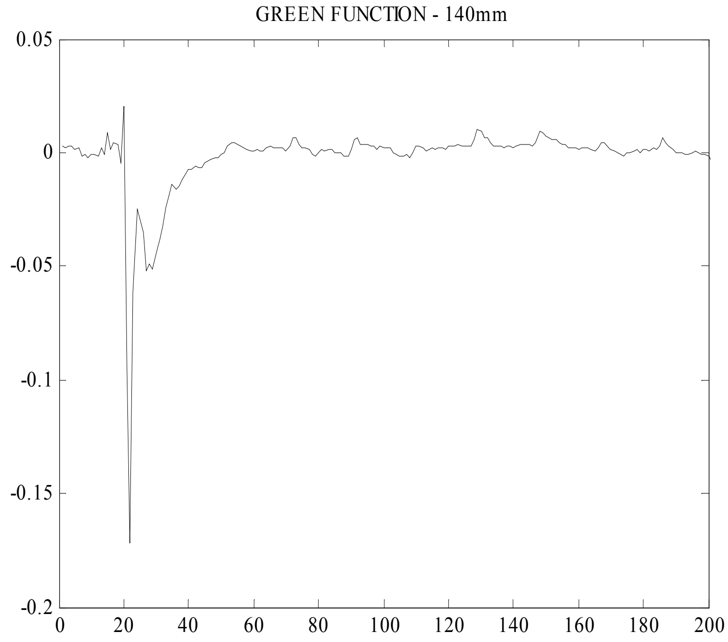
MODEL PULSES

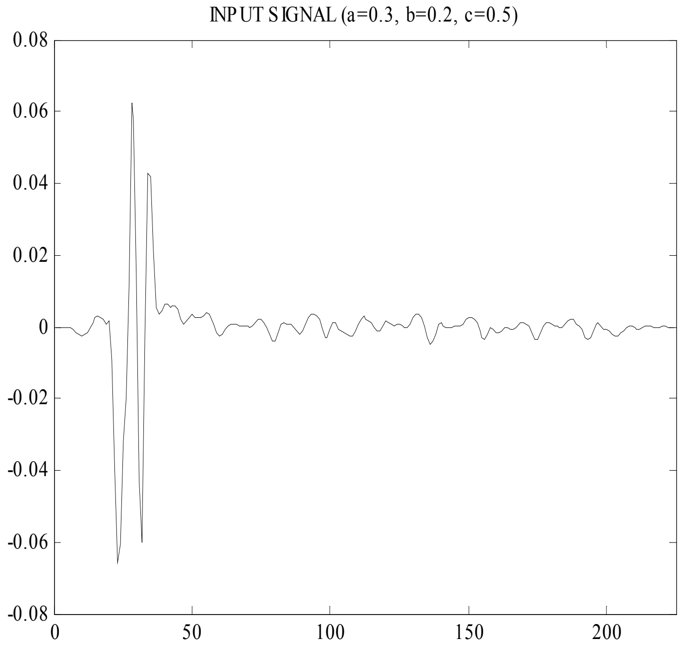


Simulation of AE-data

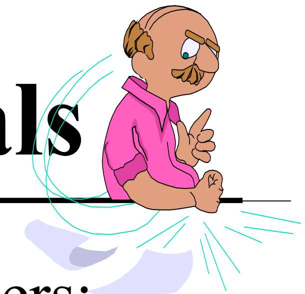


CONVOLUTION WITH THE GREEN FUNCTION





Original Features for AE-signals



- amplitude: $z_{\max} = \max_{t \in T}\{|z(t)|\}$
- rise time
- effective value (RMS)

$$RMS = \sqrt{\frac{1}{T}\int_T z^2(t)\,dt}$$

- energy moment: $T_E = \int_T t \cdot z^2(t)\,dt$
- mean value: $t_s = \left(\int_T t \cdot z(t)\,dt\right)/T$
- deviation: $\sigma^2 = \left(\int_T (t_s - z(t))^2\,dt\right)/T$
- asymmetry: $\eta^2 = \left(\int_T (t_s - z(t))^3\,dt\right)/\sigma^3$
- excess: $\xi^2 = \left(\int_T (t_s - z(t))^4\,dt\right)/\sigma^4$

- 6 spectral parameters:

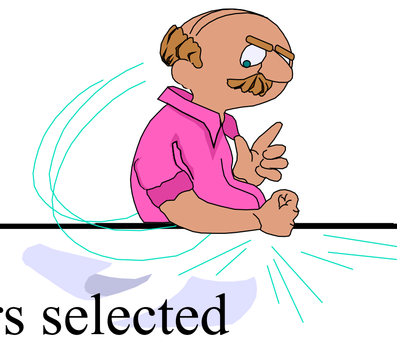
$$P_X = \frac{\int_X f(k)\,dk}{\int_G f(k)\,dk}; \quad X \in \{A, B, C, D, E, F\}$$

with
$$\begin{aligned} A &= ([0, 0.12] f_N/2) \\ B &= ([0.12, 0.24] f_N/2) \\ C &= ([0.24, 0.36] f_N/2) \\ D &= ([0.36, 0.48] f_N/2) \\ E &= ([0.48, 0.6] f_N/2) \\ F &= ([0.6, 1] f_N/2) \end{aligned}$$

and $G = ([0, 1/2] f_N)$

f_N is the Nyquist frequency

Factor analysis for input parameters

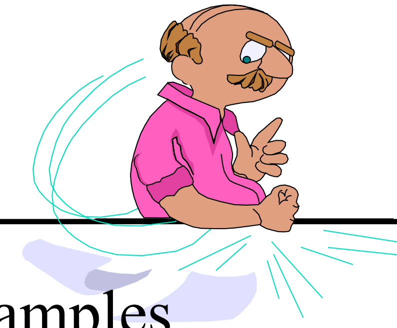


input parameters \ selected factors	1	2	3	4	5	6	7	8	9
1	0.04	0.91	0.16	0.10	0.01	0.07	0.06	0.01	0.23
2	0.09	0.02	0.01	0.19	0.03	0.03	0.95	0.03	0.16
3	0.10	0.96	0.15	0.00	0.03	0.00	0.02	0.05	0.07
4	0.13	0.91	0.02	0.03	0.05	0.06	0.04	0.06	0.20
5	0.30	0.05	0.04	0.41	0.06	0.49	0.17	0.07	0.66
6	0.26	0.00	0.04	0.03	0.08	0.93	0.02	0.10	0.20
7	0.29	0.06	0.03	0.27	0.03	0.17	0.16	0.04	0.86
8	0.12	0.06	0.01	0.88	0.02	0.02	0.24	0.03	0.36
9	0.90	0.15	0.09	0.08	0.15	0.17	0.06	0.21	0.18
10	0.93	0.14	0.06	0.08	0.10	0.17	0.06	0.09	0.19
11	0.25	0.10	0.12	0.03	0.25	0.11	0.03	0.90	0.05
12	0.20	0.07	0.14	0.02	0.93	0.09	0.03	0.23	0.04
13	0.04	0.09	0.97	0.00	0.05	0.00	0.00	0.06	0.02
14	0.08	0.18	0.95	0.02	0.09	0.04	0.01	0.06	0.02

selected factors

- 9 factors selected
- "explain" 98.4% of all variables, e.g.
  - higher energy of signals lead to higher amplitudes and RMS (parameters 1, 3, 4)
- allow to reduce linearly dependent input parameters
  - in our case to: 2, 3, 5, 6, 7, 8, 11, 12 and 2 new spectral parameters

Sensitivity analysis of trained BP-networks



SENSITIVITY COEFFICIENTS

INPUTS \ OUTPUTS	1	2	3
1	0.173	0.266	0.149
2	0.093	0.068	0.047
3	0.320	0.193	0.184
4	0.301	0.178	0.196
5	0.564	0.250	0.206
6	0.196	0.322	0.158
7	0.099	0.063	0.043
8	0.065	0.015	0.030
9	0.022	0.014	0.016
10	0.053	0.020	0.012
11	0.035	0.012	0.032
12	0.039	0.050	0.022
13	0.081	0.134	0.082
14	0.260	0.172	0.109

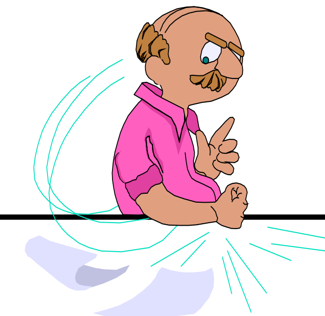
- 2000 samples
  - 500 training s.
- 14-27-19-3
- 180 iterations
- <u>selected inputs:</u>
  - sensitivity analysis

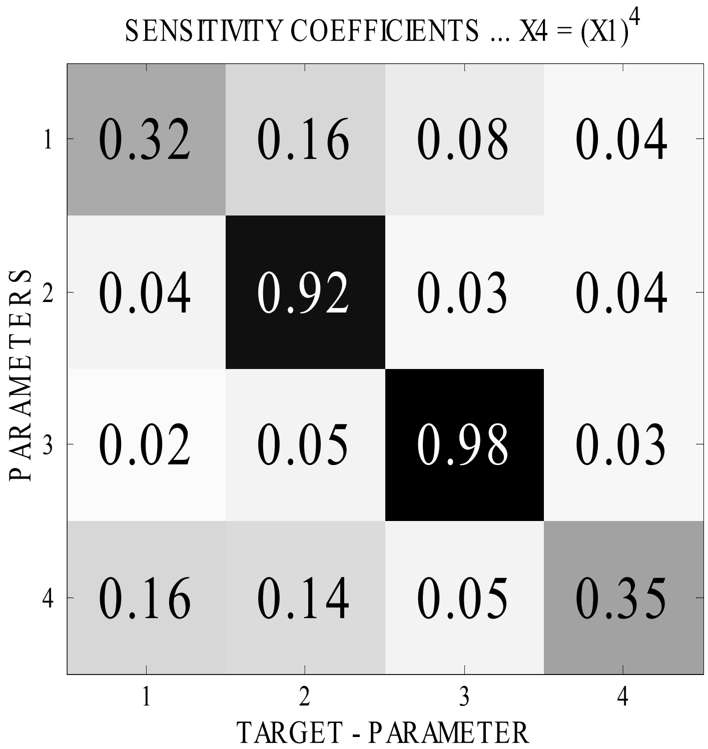
    1, 3, 4, 5, 6, 13, 14
  - + factor analysis

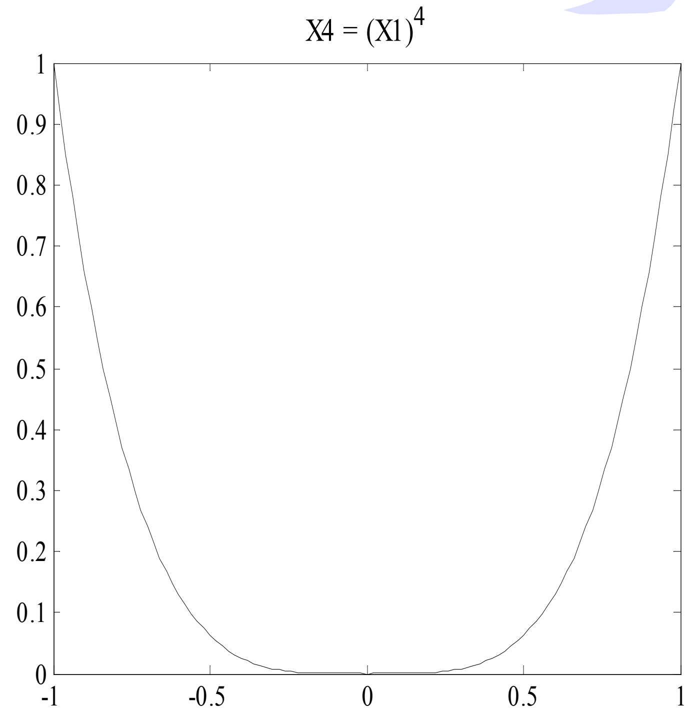
    1, 3, 5, 6, 13, 14
- new architecture:

  6-13-7-3 (even with slightly better MSE-results)

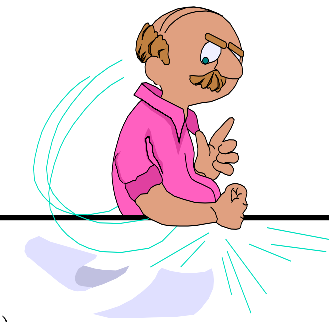
Model dependence

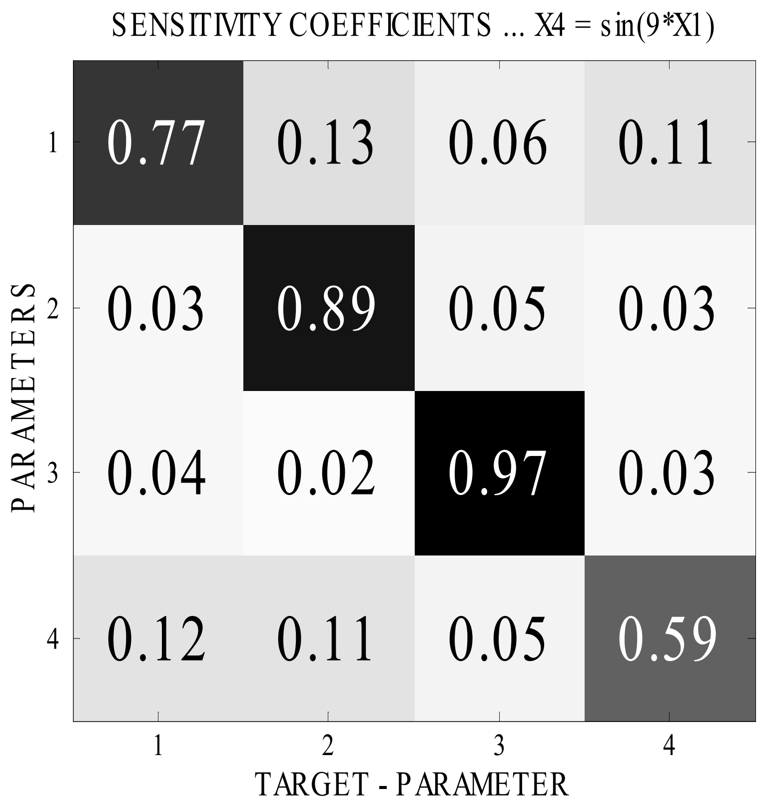


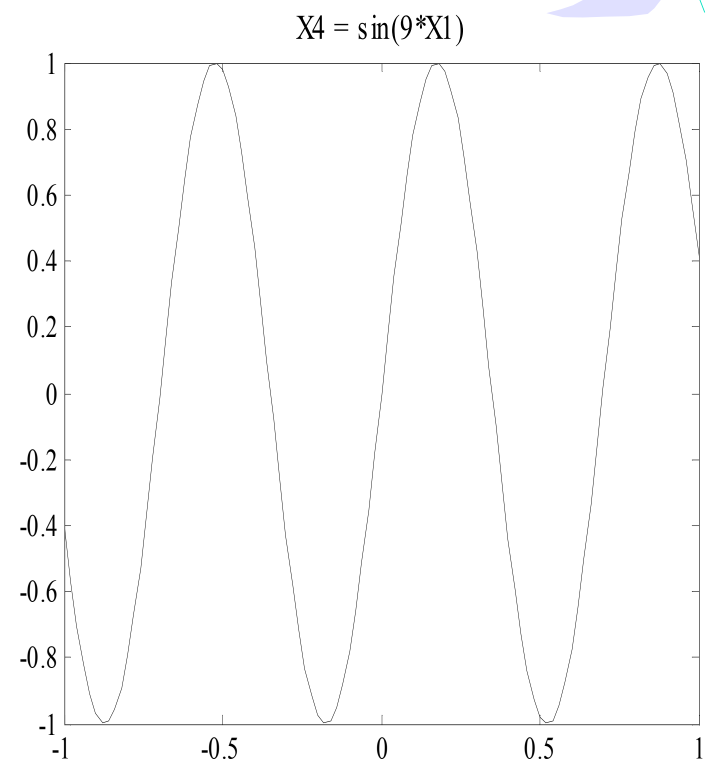




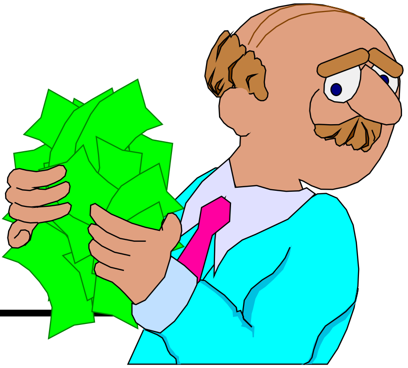
Model dependence







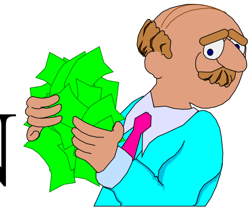
Knowledge extraction in neural networks (students' questionnaire)

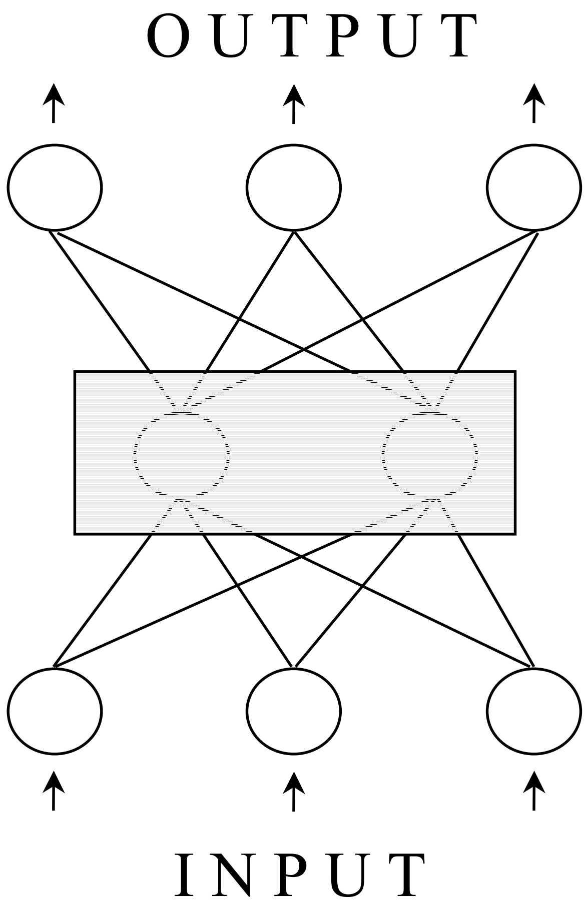


with Eva Poučková,

Department of Software Engineering, Charles University Prague

Knowledge representation in NN





Distributed!

- Which inputs are the most important ones?
- What and how does the network do?

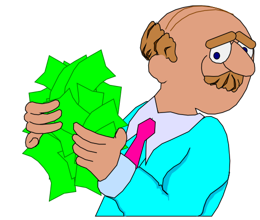
Knowledge extraction in NN

- Dimension reduction and sensitivity analysis for inputs
- Rule extraction from trained networks
  - Structural learning with forgetting
    - BP-networks
    - GREN-networks
  - Babsi-trees (B. Hammer et al.)
    - GRLVQ

Dimension reduction

- PCA: linear transformation of the input data
- Sensitivity analysis:
  Feature Subset Selection (FSS)
- Correlation-based Feature Selection (CFS):
  select a group of features with a high average correlation *input_feature - output* but with a low mutual correlation

Dimension reduction: results



PCA: method not suitable for further processing – knowledge and rule extraction

25 original features

FSS: 7 features	**CFS: 7** features

8 features selected as a union of the results for FSS and CFS

Features selected for the overall evaluation

Feature subset selection (FSS):

(1) understandable subject
(2) structured and prepared presentations
(3) interesting classes
(4) quality of education
(5) understandable classes
(6) start/end of class on time
(7) ***relationship to students***

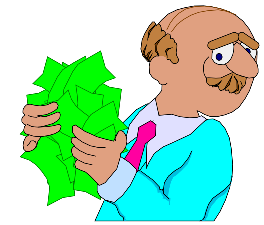
Correlation-based feature selection (CFS):

(1) understandable subject
(2) structured and prepared presentations
(3) interesting classes
(4) quality of education
(5) understandable classes
(6) start/end of class on time
(8) ***students prepare for classes***

Methods for knowledge extraction

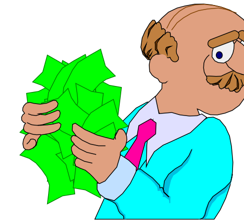
- **SLF – Structural learning with forgetting**
  - **Learning with forgetting**
  - **Learning with forcing internal representations on hidden neurons**
  - **Learning with selective forgetting**
- **Babsi-trees**
  - **Form a tree from a neural network trained by means of the GRLVQ-method**

Generalized relevance learning vector quantization (GRLVQ)



- a robust combination of GLVQ and RLVQ
- provides weighing factors (λ) for input features
  - **larger λ corresponds to a “more important” feature**
- applicable to pruning of input features
- **<u>GLVQ:</u>** considers class representatives
  - separating surfaces approach the optimum Bayessian ones
- **<u>RLVQ:</u>** input features can have different importance
  - relatively unstable, sensitive to noise

Generalized LVQ - GLVQ



- Select a fixed number of representatives $\mathbf{w}_1, \ldots, \mathbf{w}_L$ for all classes C_i, $i=1, \ldots, q$.
- Receptive field of the class representative $\mathbf{w}_i$

$$R_i = \{\mathbf{x} \in T \mid \forall k \neq i : \| \mathbf{x} - \mathbf{w}_i \| < \| \mathbf{x} - \mathbf{w}_k \| \}$$

- **Receptive fields of class representatives should be as small as possible!**
  - minimize $E = \sum_{k=1}^{p} \sigma\left(\eta\left(\mathbf{x}^{(t)}\right)\right)$; σ denotes the sigmoid
  - and $\eta\left(\mathbf{x}^{(t)}\right) = \dfrac{\| \mathbf{x}^{(t)} - \mathbf{w}^{+} \| - \| \mathbf{x}^{(t)} - \mathbf{w}^{-} \|}{\| \mathbf{x}^{(t)} - \mathbf{w}^{+} \| + \| \mathbf{x}^{(t)} - \mathbf{w}^{-} \|}$

correct classification — wrong classification

Generalized LVQ - GLVQ

- Weight adjustment:

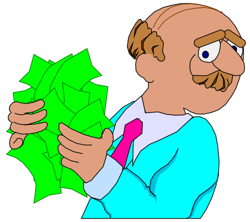
$$\Delta \mathbf{w}^{+} = \alpha \sigma' \frac{\| \mathbf{x}^{(t)} - \mathbf{w}^{-} \|}{\left(\| \mathbf{x}^{(t)} - \mathbf{w}^{+} \| + | \mathbf{x}^{(t)} - \mathbf{w}^{-} \| \right)^2} \left(\mathbf{x}^{(t)} - \mathbf{w}^{+} \right)$$

$$\Delta \mathbf{w}^{-} = -\alpha \sigma' \frac{\| \mathbf{x}^{(t)} - \mathbf{w}^{+} \|}{\left(\| \mathbf{x}^{(t)} - \mathbf{w}^{+} \| + | \mathbf{x}^{(t)} - \mathbf{w}^{-} \| \right)^2} \left(\mathbf{x}^{(t)} - \mathbf{w}^{-} \right)$$

with $\sigma' = \sigma(\eta(\mathbf{x}^{(t)}))' = \sigma(\eta(\mathbf{x}^{(t)}))(1 - \sigma(\eta(\mathbf{x}^{(t)})))$

and learning rates α

Relevance LVQ - RLVQ



- Input features can have different importance λ

$$dist(\mathbf{x},\mathbf{w})_\lambda = \sum_{i=1}^{n} \lambda_i (x_i - w_i)^2 \quad ; \quad \sum_{i=1}^{n} \lambda_i^2 = 1$$

- Receptive field of the class representative $\mathbf{w}_i$

$$R_{i\lambda} = \left\{ \mathbf{x} \in T \mid \forall j \neq i : \| \mathbf{x} - \mathbf{w}_i \|_\lambda < \| \mathbf{x} - \mathbf{w}_j \|_\lambda \right\}$$

- Weight adjustment according to GLVQ with **adaptive importance factors** λ_i for input features ($0 < \varepsilon < 1$):

$$\Delta \lambda_i^{(t)} = \begin{cases} \max\left(\lambda_i^{(t-1)} - \varepsilon\left(x_i^{(t)} - w_{ij}\right)^2, 0\right) & d^{(t)} = c_j \\ \lambda_i^{(t-1)} + \varepsilon\left(x_i^{(t)} - w_{ij}\right)^2 & \text{else} \end{cases}$$

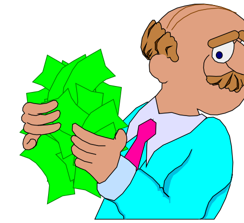
Generalized Relevance Learning Vector Quantization (GRLVQ)

- Weight adjustment according to GLVQ with **adaptive importance factors** λ_i for input features:

$$\Delta\lambda_i^{(t)} = -\varepsilon\sigma'\left(\frac{\|\mathbf{x}^{(t)} - \mathbf{w}^-\|}{\left(\|\mathbf{x}^{(t)} - \mathbf{w}^+\| + |\mathbf{x}^{(t)} - \mathbf{w}^-\|\right)^2}\left(x_i^{(t)} - w_i^+\right)^2 - \right.$$

$$\left. - \frac{\|\mathbf{x}^{(t)} - \mathbf{w}^+\|}{\left(\|\mathbf{x}^{(t)} - \mathbf{w}^+\| + |\mathbf{x}^{(t)} - \mathbf{w}^-\|\right)^2}\left(x_i^{(t)} - w_i^-\right)^2\right)$$

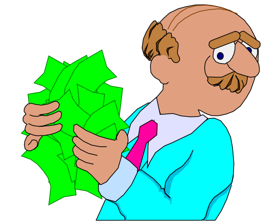
Babsi-trees



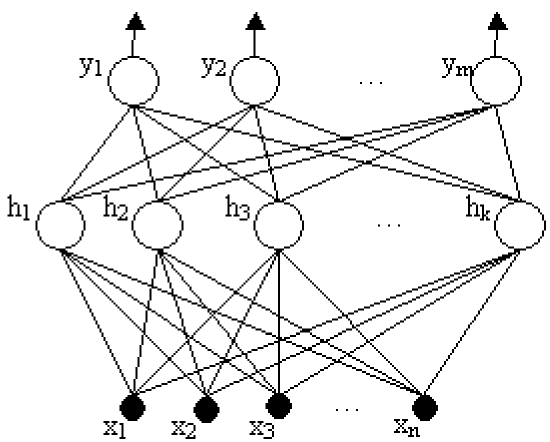
Root-trees $G=(V,E)$ satisfying the following conditions:

- all vertices $v_i \in V$ can have an arbitrary number n_i of sons
- all leaves v_J are labeled with the corresponding classification class C_J
- all vertices v_i which are not leafs are labeled with I^{v_i}
  - I^{v_i} stands for the currently processed input dimension i
  - dimensions are "ordered" according to their importance (λ)
- All edges going from a vertex v_i to its sons are labeled with intervals $\left\langle st_k^{v_i}, st_l^{v_i} \right)$
  - interval boundaries are placed in the middle between two neighboring cluster representatives

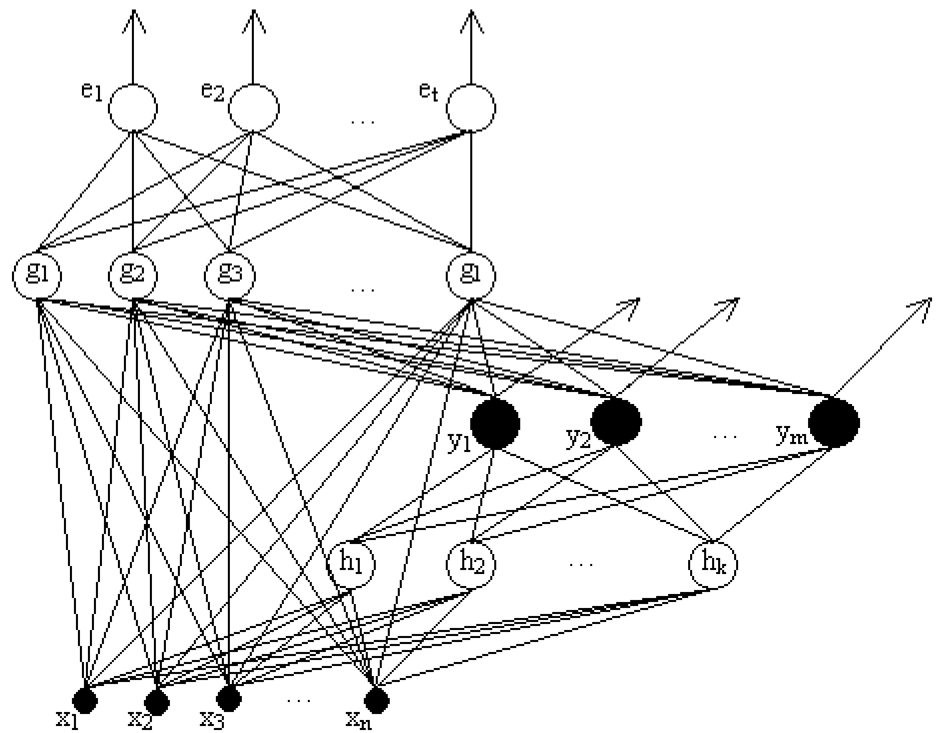
SLF for layered networks



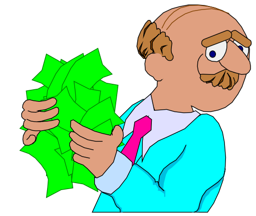
feed-forward
neural networks



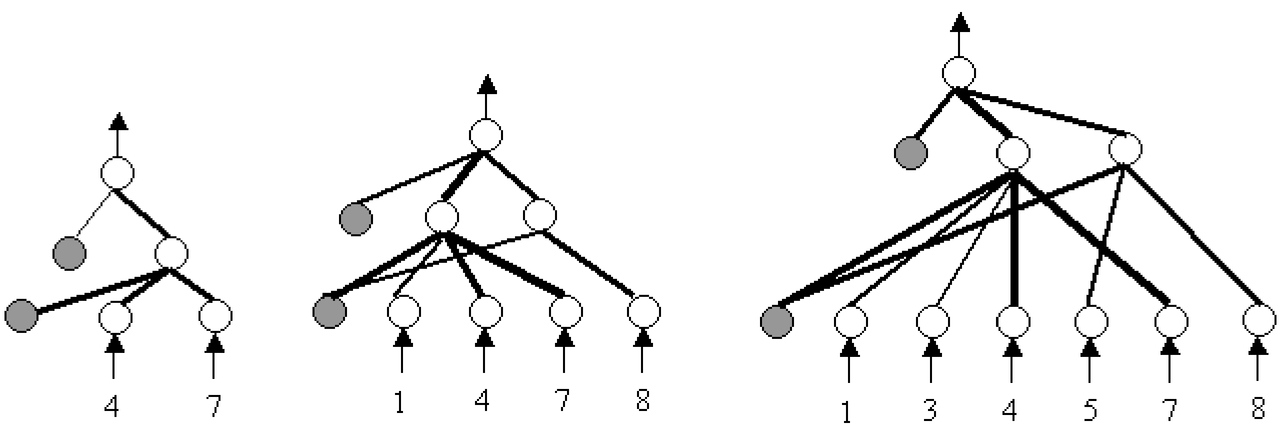
GREN-networks



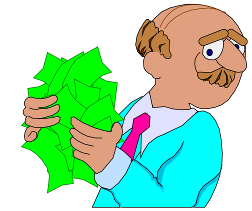
Results for the SLF-method



Both BP-networks and GREN-trained networks lead to similar sets of rules:



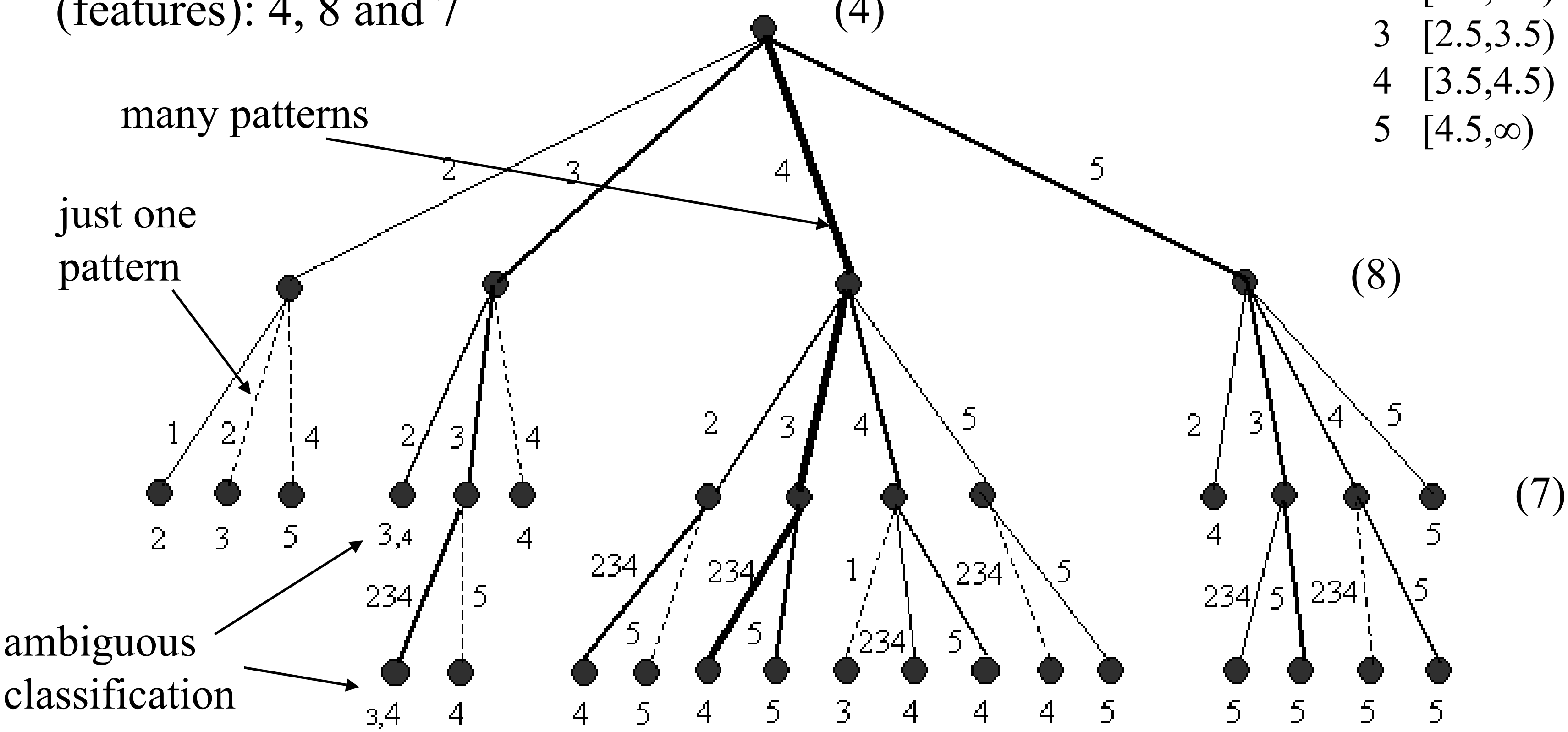
The resulting Babsi-tree



Relevant dimensions (features): 4, 8 and 7

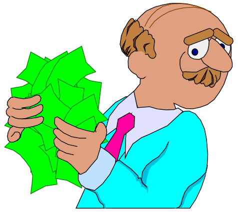
Values for intervals:

1	(-∞,1.5)
2	[1.5,2.5)
3	[2.5,3.5)
4	[3.5,4.5)
5	[4.5,∞)



Overall evaluation

Comparison of the results



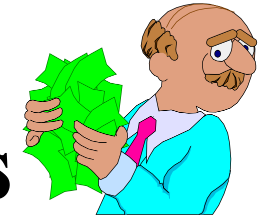
SLF

- Few simple hierarchically ordered rules
- Possibility to add rules after achieving the desired accuracy
- Rule correctly applicable - 71% and 73%, resp.

Babsi-trees

- Many simple rules
- Quick training of the network
- Few training parameters
- Rule correctly applicable - 67%

Knowledge extraction: Conclusions



- **Main results achieved:**
  - dimension reduction for the input space
  - analysis of various models for knowledge extraction
  - rule extraction from GREN-networks
  - comparison with other neural network models
- **Further research:**
  - adjusting rules extracted from a neural network trained with the GRLVQ-algorithm
  - (automatic) selection of training parameters for the SLF-algorithm

Genetic Algorithms (GA)

Apply genetics and natural selection to find optimal parameters of a predictive function!

- GA use “genetic” operators to evolve successive generations of solutions:
  - selection
  - crossover
  - mutation
- Best candidates “survive” to further generations until convergence is achieved
- Directed data mining

The basic Genetic Algorithm

- **Step 1:** Create an initial population of individuals
- **Step 2:** Evaluate the fitness of all individuals in the population
- **Step 3:** Select candidates for the next generation
- **Step 4:** Create new individuals (use genetic operators - crossover and mutation)
- **Step 5:** Form a new population by replacing (some) old individuals by new ones
- **GOTO Step 2**

ANTARES

STUDENT SOFTWARE PROJECT

supervised by I. Mrázová, F. Mráz

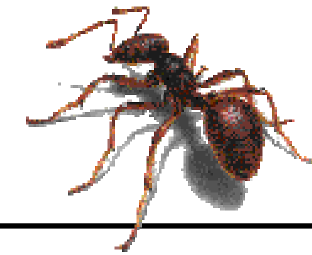
participating students:

D. Bělonožník, D. J. Květoň, M. Šubert,

J. Tomaštík, J. Tupý

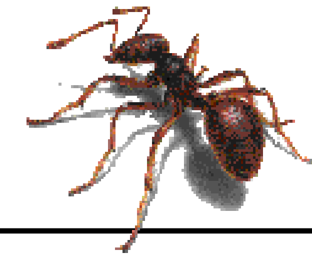
http://www.ms.mff.cuni.cz/~mraz/antares

Project ANTARES



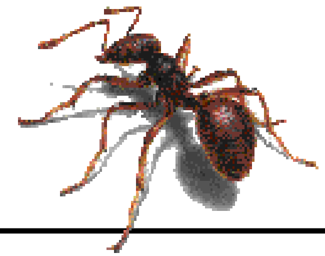
- **<u>Generate melodies</u>** with genetic algorithms
  - the fitness of candidate solutions is evaluated by the cooperating feed-forward neural network
- Parallel implementation of genetic algorithms
  - open system for the design and testing of genetic algorithms and neural networks
  - supports mutual cooperation between neural networks and genetic algorithms

Fitness evaluation with neural networks

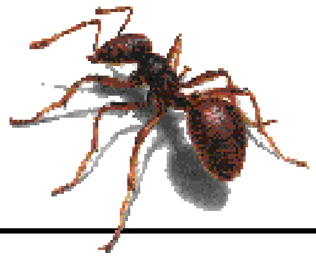


- For some problems, it might be difficult to define explicitly the fitness function
  - e.g. „evaluate“ the beauty of generated melodies
- Fitness of candidate melodies (generated by GA) is evaluated by the pre-trained NN:
  - provide a set of positive and negative examples (supervised learning)
  - train a feed-forward network to approximate the „unknown“ fitness function (on the training set)

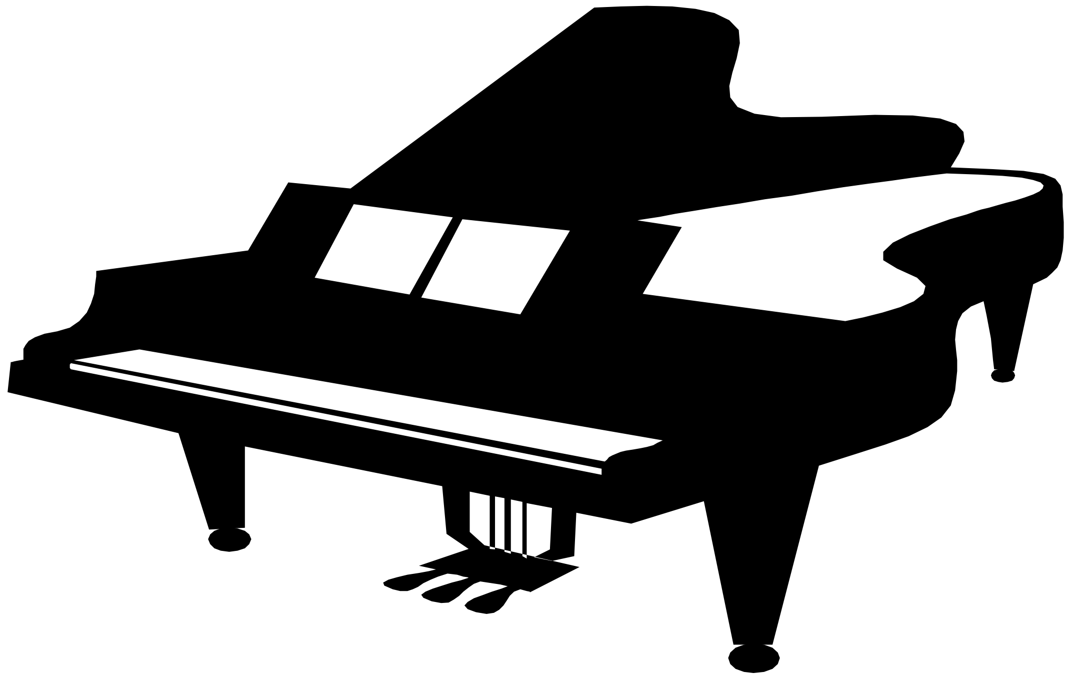
Generating melodies: positive training samples



Generating melodies



- **Positive training samples**
- **Negative training samples**
- **Test samples (with a high fitness value)**
- **Generated melodies**



Thank you for your attention!