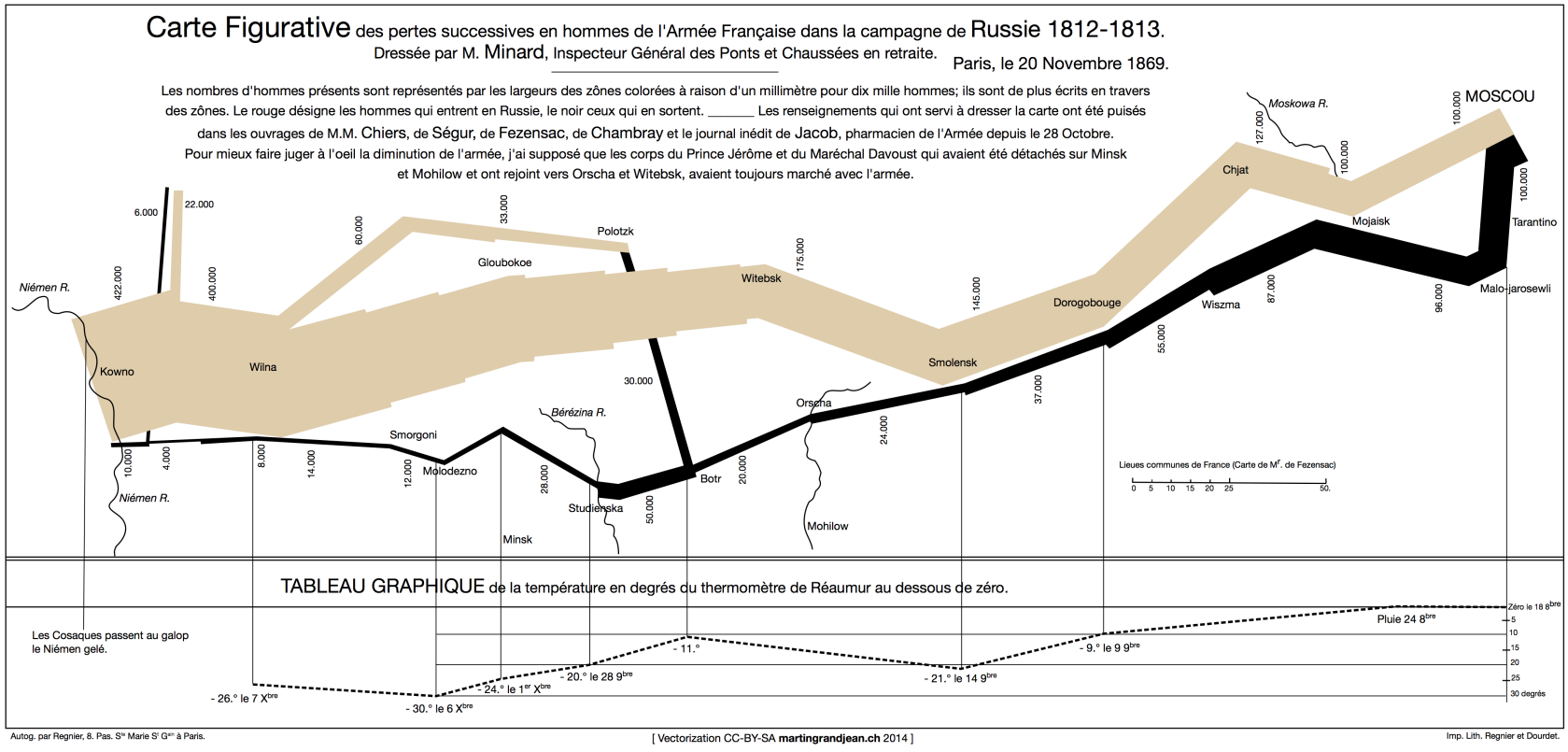
Multidimensional Data Visualization

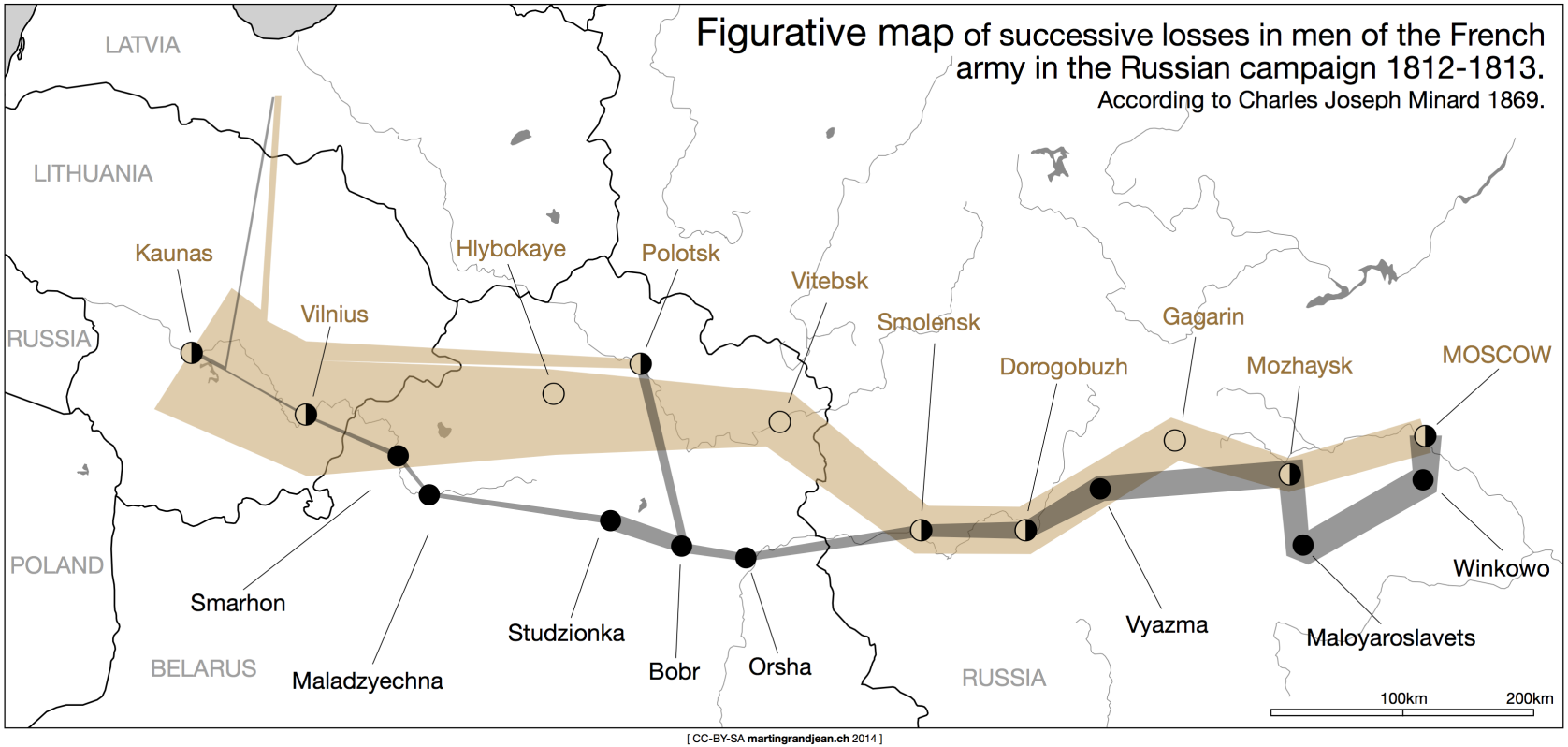
Introduction

A Good Sketch is Better Than a Long Speech



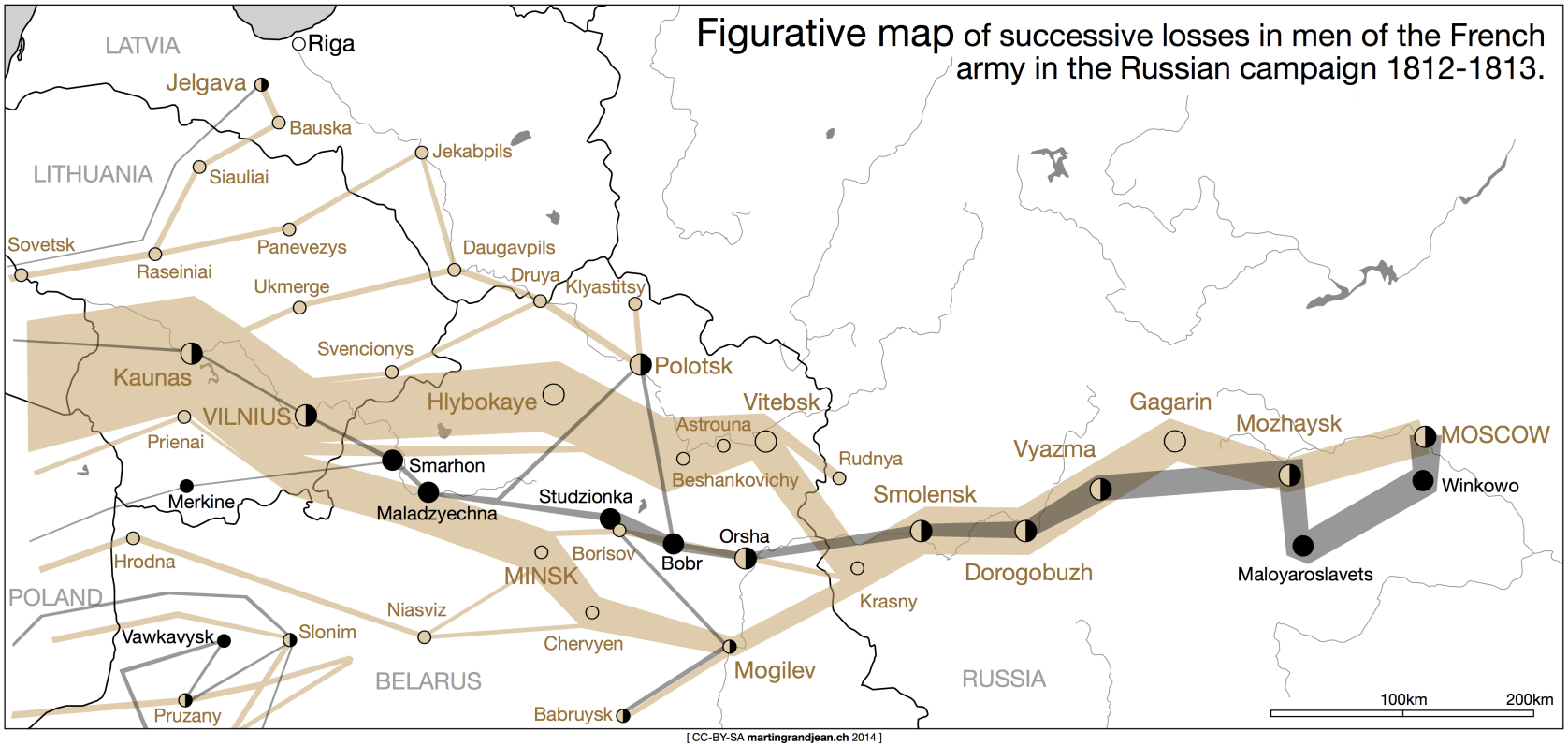
Charles Joseph Minard's 1869 diagram of Napoleon's March. The original map shows the road to Moscow in brown and the way back in black. The path is simplified into a single stream, as explained in the description of Minard, under the title. Only a few cities are displayed, the path is summarized in segments between these points. On the way back, a graph shows the temperatures at irregular intervals.

Minard's Map: Geographic Display



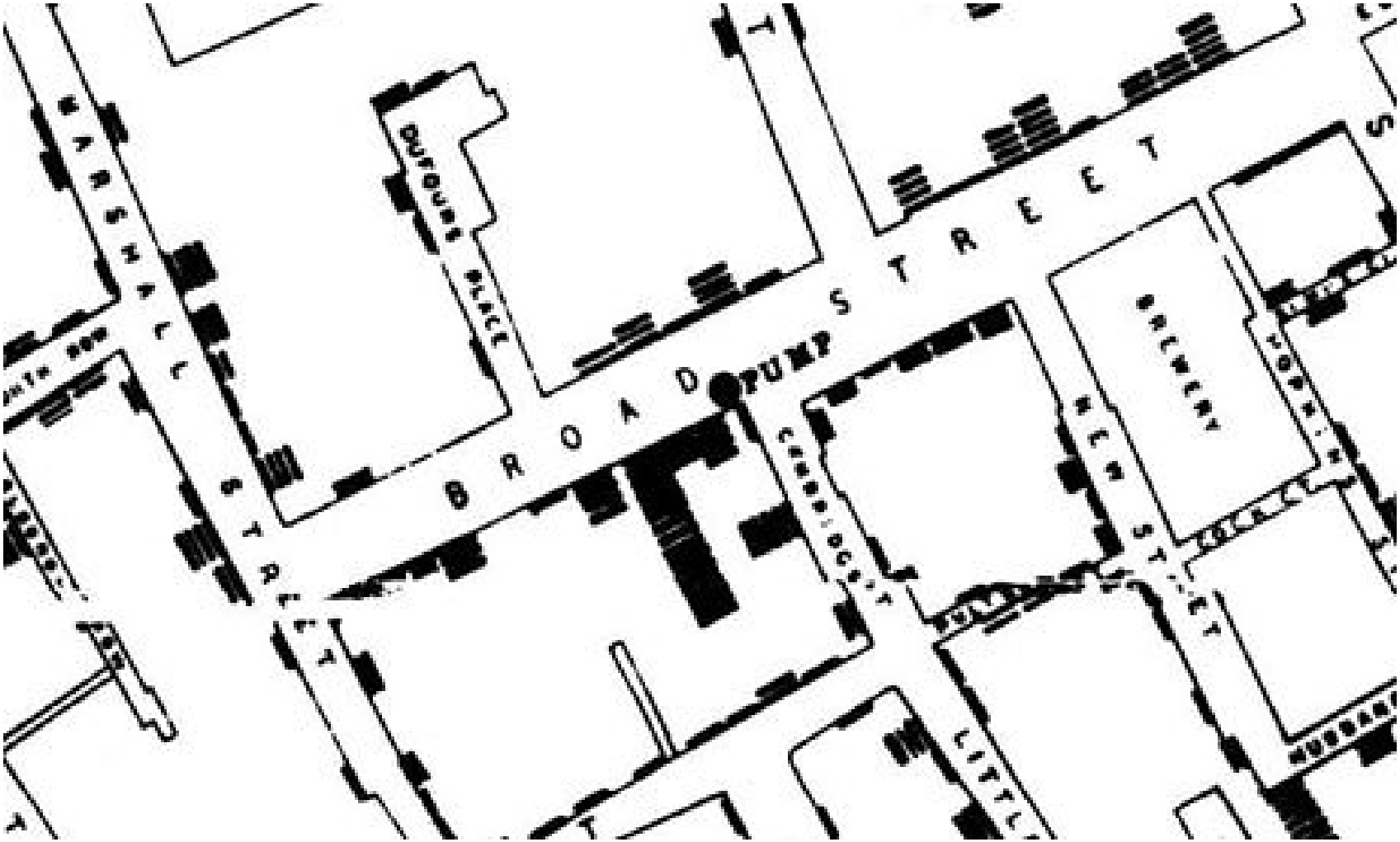
Using data from Minard, this map projects the path taken by Napoleon's troops in the geographical reality. To make this map understandable, places and borders reflect the current situation (2014). Brown/Black dots indicate the cities crossed twice.

Minard's Map: Historical Map



The reality is not as simple as the visualization of 1869 suggests: Napoleon's army was divided into several corps which followed different paths and fortunes. This third map combines Minard's codes and the most accurate informations we have about the actual route of the different corps of the "Great Army". The small dots indicate the places that Minard didn't mention.

Visualization Based Discovery



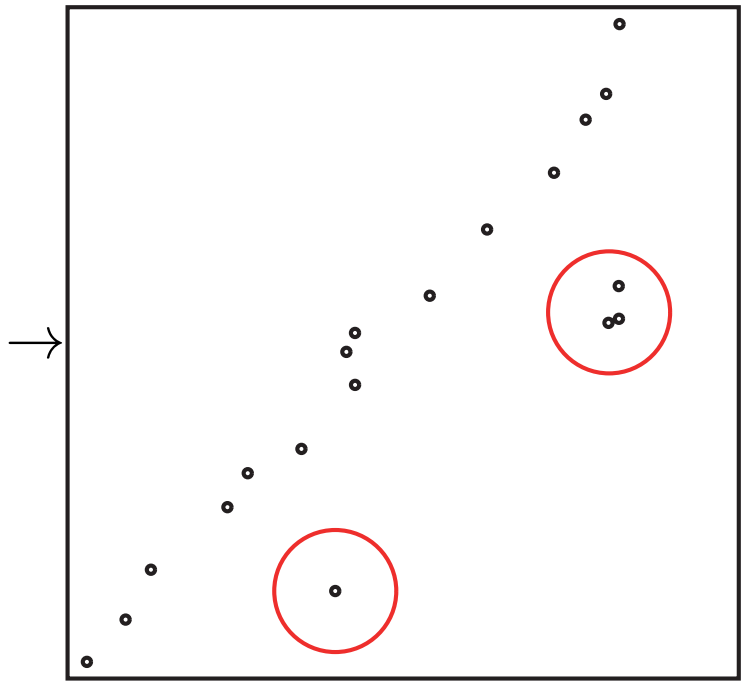
A map by John Snow showing the clusters of cholera cases in the London epidemic of 1854. Snow used a dot map to illustrate the cluster of cholera cases around the pump.

Multidimensional Data and Visualization

- It is often desirable to visualize a data set the items of which are described by more than three features. Therefore, we have multidimensional data and our goal is to make some visual insight into the data set analyzed.
- For human perception, the data must be represented in a low-dimensional space, usually of two or three dimensions. The goal of visualization methods is to represent the multidimensional data in a low-dimensional space so that certain properties (e.g. clusters, outliers) of the structure of the data set were preserved as faithfully as possible.
- Such a visualization of data is highly important in data mining because recent applications produce a large amount of data that require specific means for knowledge discovery.
- The dimensionality reduction or visualization methods are recent techniques to discover knowledge hidden in multidimensional data sets.

Example of Visualization

5.86	2.91	−4.19	−8.49	0.43	−1.13
0.31	−1.14	−2.25	−2.60	−1.58	2.17
11.58	2.97	−14.31	−14.18	3.46	−0.99
15.14	5.46	−20.15	−16.61	0.87	0.31
−1.25	0.39	0.40	2.50	0.16	−0.13
−14.42	−3.81	12.65	13.92	1.94	0.93
5.90	3.36	−10.09	−7.96	−0.85	0.89
−9.55	−0.93	9.71	11.53	2.54	−1.41
13.98	3.41	−20.10	−11.60	−0.59	−1.55
0.85	0.37	−2.40	−3.83	−1.15	0.86
−5.96	−2.06	7.90	9.44	1.06	−1.46
6.39	6.82	−12.52	−8.35	2.05	0.49
−3.92	−1.66	6.54	2.82	−1.70	0.65
3.99	−0.83	−3.87	−1.85	−1.05	1.08
−10.36	−2.47	12.88	10.64	0.76	−0.75
2.62	3.72	9.95	7.88	−0.91	−0.37
0.76	2.63	9.47	10.40	0.35	1.02
2.71	2.99	8.75	10.28	−0.59	2.34
13.84	7.71	−7.00	−6.33	−0.68	1.57



- The data visualization allows us to detect the presence of clusters, outliers, or regularities in the analyzed data.
- It is evident that some items of the data set form separate clusters – outliers and the remaining ones are scattered near to a line. These clusters – outliers and distribution around the line can be clearly observed visually on a plane and cannot be recognized directly from the table without a special analysis.

Multidimensional Data Visualization

- A natural idea arises to present multidimensional data, stored in such a table (matrix), in some visual form. It is a complicated problem followed by extensive researches, but its solution allows a human being to gain a deeper insight into the data, draw conclusions, and directly interact with the data.
- Such a possibility to present multidimensional data in a visual form is not one and only. A large number of methods have been developed for multidimensional data visualization.
- It is desirable to preserve certain properties of the structure of the data set as faithfully as possible when transferring from several dimensions to two.
- In this course we will review and discuss various methods for multidimensional data visualization.

Principal Notations

- At first, we determine the principal notions and terms used in this course.
- Here we confront with two principal terms: *object* and *feature*.
- The term *object* can cover various things: people, equipment, products of manufacturing, plants, natural phenomena, etc.
- An object is characterized by some *features*. For example, the patient is an object, he (she) can be described by a number of features, such as name, sex, age, and diagnostic test results like blood pressure, cholesterol level, etc.
- If the data set consists of a lot of objects, then the data set is called a large data set. If the number of features is large, then the data set is called a high-dimensional data set.

Multidimensional Data

- Objects are also called *items*, *instances*, *samples*, *observations*.
- Features are called *attributes*, *parameters*, *properties*, *variables dimensions*.
- Objects described by the same features $x_1, x_2, \ldots, x_n$ form a *data set*. Assume that any feature may take some numerical values.
- A combination of values of all features characterizes a particular object

$$X_i = (x_{i1}, x_{i2}, \ldots, x_{in}), \ i \in \{1, \ldots, m\},$$

  where n is the number of features, m is the number of objects, i is the order number of the object.
- If the objects $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$ are described by more than one feature, the data characterizing the objects are called *multidimensional data*.
- If the number of features is n, then $X_1, X_2, \ldots, X_m$ are the n-dimensional data items.

Multidimensional Points

- Often $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ are interpreted as points in the multidimensional space $\mathbb{R}^n$, where n defines the dimensionality of the space.
- The coordinate values of point X_i are values of the features $x_{i1}, x_{i2}, \ldots, x_{in}$. In such a case, we have a matrix (table) X of numerical data:

$$X = \{X_1, X_2, \ldots, X_m\} = \left\{x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\right\}, \tag{1}$$

  and the ith row of this matrix is a point $X_i \in \mathbb{R}^n$, where $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$ and x_{ij} is the jth coordinate of the ith point, m is the number of points in the data set. The data point X_i contains feature values of corresponding object.
- Sometimes it does not suffice to refer to X_i as a point, so the notion of a *vector* can be useful to enlarge the properties of points. The points $X_1, X_2, \ldots, X_m$ can be conceived as vectors bound to the origin $(0, 0, \ldots, 0)$.

Proximity of Data

- Note that there are cases where we do not have and cannot get a set of numerical values of the features characterizing a particular object. However, we can estimate proximities between two objects.
- Let us determine the notion of *proximity* between two objects X_i and X_j.
- A (dis)similarity is a proximity that indicates how two objects X_i and X_j are (dis)similar. The (dis)similarity is denoted by δ_{ij}.
- If δ_{ij} is a similarity, a high δ_{ij} value indicates that the objects X_i and X_j are very similar.
- For dissimilarities, a small δ_{ij} value indicates that the objects are very similar.
- When the proximities are known, the visualization of objects $X_1, X_2, \ldots, X_m$ may be carried out using the matrix of their proximities $\Delta = \{\delta_{ij}, i, j = 1, \ldots, m\}$. The advantage is that the dimensionality n can be unknown. This often happens, for example, in psychological tests.

Proximity Measures

- A proximity matrix can be obtained from matrix X applying some proximity measure, too.
- Often the proximity is measured using the Euclidean distance, which belongs to the group of Minkowski distances. The Minkowski distance between two objects $X_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ and $X_l = (x_{l1}, x_{l2}, \ldots, x_{ln})$ is defined by the formula:

$$d_q(X_k, X_l) = \left\{ \sum_{j=1}^{n} |x_{kj} - x_{lj}|^q \right\}^{\frac{1}{q}}.$$

- Some other proximity measures are also possible: Canberra distance, Bray-Curtis dissimilarity, correlation, etc.

Minkowski Distances

- The following Minkowski distances may be derived for different q:
- City-block or Manhattan distance, $q = 1$:

$$d_1(X_k, X_l) = \sum_{j=1}^{n} |x_{kj} - x_{lj}|.$$

- Euclidean distance, $q = 2$:

$$d_2(X_k, X_l) = \sqrt{\sum_{j=1}^{n} |x_{kj} - x_{lj}|^2}.$$

- Chebyshev distance, $q = \infty$:

$$d_\infty(X_k, X_l) = \max_j \left| x_{kl} - x_{lj} \right|.$$

Conditions of Distances

- Here the distances between two objects X_k and X_l satisfy the following conditions:
- $d(X_k, X_l)$ is a nonnegative real number;
- $d(X_k, X_k) = 0$;
- $d(X_k, X_l) = d(X_l, X_k)$, i.e. the distance from object X_k to object X_l is equal to the distance from object X_l to object X_k;
- $d(X_k, X_l) \leq d(X_k, X_j) + d(X_j, X_l)$, i.e. the distance between any two objects X_k and X_l cannot be larger than a sum of distances between objects X_k, X_j and X_l, X_j (triangle inequality).

Data Normalization

- Let the analyzed data set $X = \{X_1, X_2, \ldots, X_m\}$ contains n-dimensional vectors $X_i = (x_{i1}, x_{i2}, \ldots, x_{in}),\ i = 1, \ldots, m$, i.e. i-th row of the matrix X is a vector X_i.
- Often the values of features $x_1, x_2, \ldots, x_n$ of a multidimensional data set $X = \{X_1, X_2, \ldots, X_m\}$ are of different scales or measured in different units (e.g. kilograms, meters, degrees).
- Therefore the scales must be unified/normalized before analysis of the data set. There are several ways of normalization.

Data Normalization: Normalizing Residuals

1. Values of parameters are changed so that the mean was 0 and standard deviation was 1. For each feature x_j the average

$$\overline{x_j} = \frac{1}{m} \sum_{i=1}^{m} x_{ij}$$

and standard deviation

$$\sigma_j^2 = \frac{1}{m-1} \sum_{i=1}^{m} (x_{ij} - \overline{x_j})^2$$

is estimated. Every value of feature x_{ij} is transformed using

$$x_{ij} = \frac{x_{ij} - \overline{x_j}}{\sigma_j}.$$

Data Normalization: Feature Scaling

2. Feature scaling is used to bring all values into the range $[0, 1]$. It is also called unity-based normalization. For each feature x_j minimal $x_{j\,\mathrm{min}}$ and maximal $x_{j\,\mathrm{max}}$ values are estimated and every value of feature x_{ij} is transformed using

$$x_{ij} = \frac{x_{ij} - x_{j\,\mathrm{min}}}{x_{j\,\mathrm{max}} - x_{j\,\mathrm{min}}}.$$

Methods of Multidimensional Data Visualization

- The methods of multidimensional data visualization can be divided into two groups.
- direct visualization methods, where each feature, characterizing a multidimensional object, is represented in a visual form;
- projection, so-called dimensionality reduction, methods, allowing us to represent the multidimensional data on a low-dimensional space.
- Our aim is not to enumerate all methods and describe them in detail, but to present the most typical approaches and representatives of each group.

Direct Visualization Methods

- There is no formal mathematical criterion to estimate the visualization quality in direct visualization methods.
- All the features that characterize multidimensional data are represented in a visual form acceptable to a human.
- These methods may be classified into geometric, iconographic, and hierarchical visualization techniques.

Direct Visualization Methods

1. Geometric methods:
   a) scatter plots,
   b) matrix of scatter plots,
   c) multiline graphs,
   d) Andrews curves,
   e) parallel coordinates,
   f) radial visualization (RadViz) and its modifications GridViz and PolyViz.
2. Iconographic displays:
   a) Chernoff faces,
   b) star glyphs.
3. Hierarchical displays:
   a) dimensional stacking,
   b) trellis display,
   c) hierarchical parallel coordinates.

Projection (Dimensionality Reduction) Methods

- Methods that allow us to represent multidimensional data from $\mathbb{R}^n$ in a low-dimensional space $\mathbb{R}^d$, $d < n$, are called projection (dimensionality reduction) methods.
- If the dimensionality of the *projection space* is small enough ($d = 2$ or $d = 3$), these methods may be used to visualize the multidimensional data.
- In such a case, the projection space can be called a *display*, *embedding* or *image space*.
- These methods usually invoke formal mathematical criteria by which the projection distortion is minimized.

1. Linear projection methods:
   a) principal component analysis,
   b) linear discriminant analysis,
   c) projection pursuit.
2. Nonlinear projection methods:
   a) multidimensional scaling,
   b) locally linear embedding,
   c) isometric feature mapping,
   d) principal curves.

Artificial Neural Networks

- Artificial neural networks may also be used for visualizing multidimensional data. They realize various nonlinear projections.

1. Self-organizing map.
2. Neural gas.
3. Curvilinear component analysis.
4. Multidimensional scaling using artificial neural networks:
   a) supervised learning strategy,
   b) unsupervised learning strategy,
   c) combinations of self-organizing map and neural gas with multidimensional scaling.
5. Auto-associative neural network.
6. NeuroScales.

Multidimensional Data Sets

- We have presented one of the possible classifications of methods for multidimensional data visualization.
- The best way to investigate the visualization methods is to use the test data sets with the known structure. The performance of the methods, discussed in this course, is illustrated on the real-life and artificial data sets.

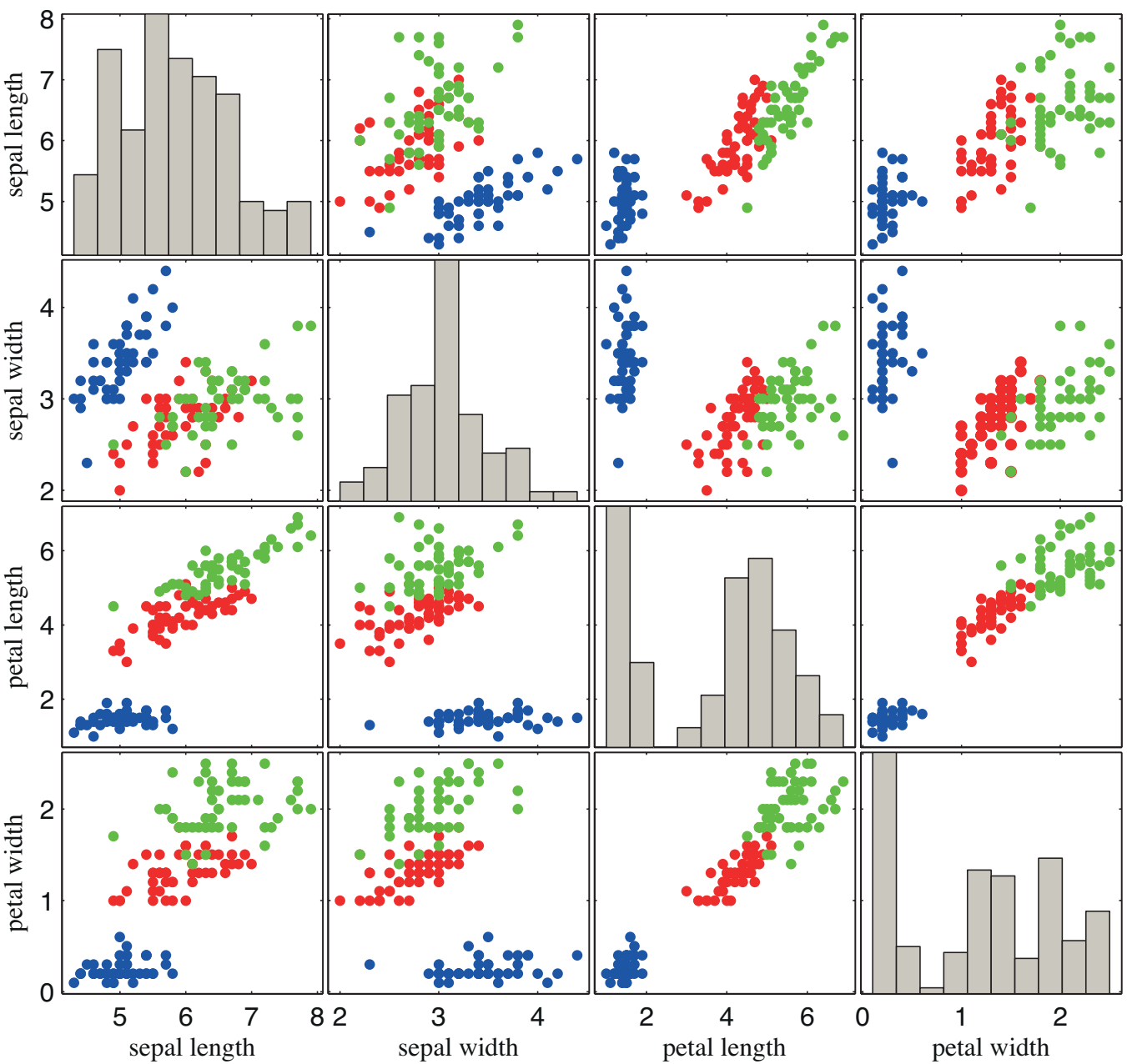
Examples of Multidimensional Data

- Some data sets are used to illustrate the methods for visualizing multidimensional data and experimental investigations. The data sets are described by n-dimensional points $X_1, X_2, \ldots, X_m$, where $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, or the dissimilarity matrix Δ of size $m \times m$. The coordinates of points are defined by the values of features $x_1, x_2, \ldots, x_n$ of corresponding objects.
- The elements δ_{ij} of dissimilarity matrix describe the proximity of the ith and jth objects.
- Some multidimensional data examples from the database „UCI Repository of Machine Learning Databases“ (http://archive.ics.uci.edu/ml/).
  - Fisher Iris data set.
  - Auto MPG data set.
  - Breast Cancer data set.

Fisher Iris Data Set

- The *Iris* data set consists of 150 flowers of three species: *Setosa*, *Virginica* and *Versicolor*. Each species is represented by 50 flowers ($m = 150$, $n = 4$).
- Four features of each flower were measured:
  - sepal length (x_1),
  - sepal width (x_2),
  - petal length (x_3),
  - petal width (x_4).

Scatter Plot Matrix of Iris Data



Setosa

Versicolor

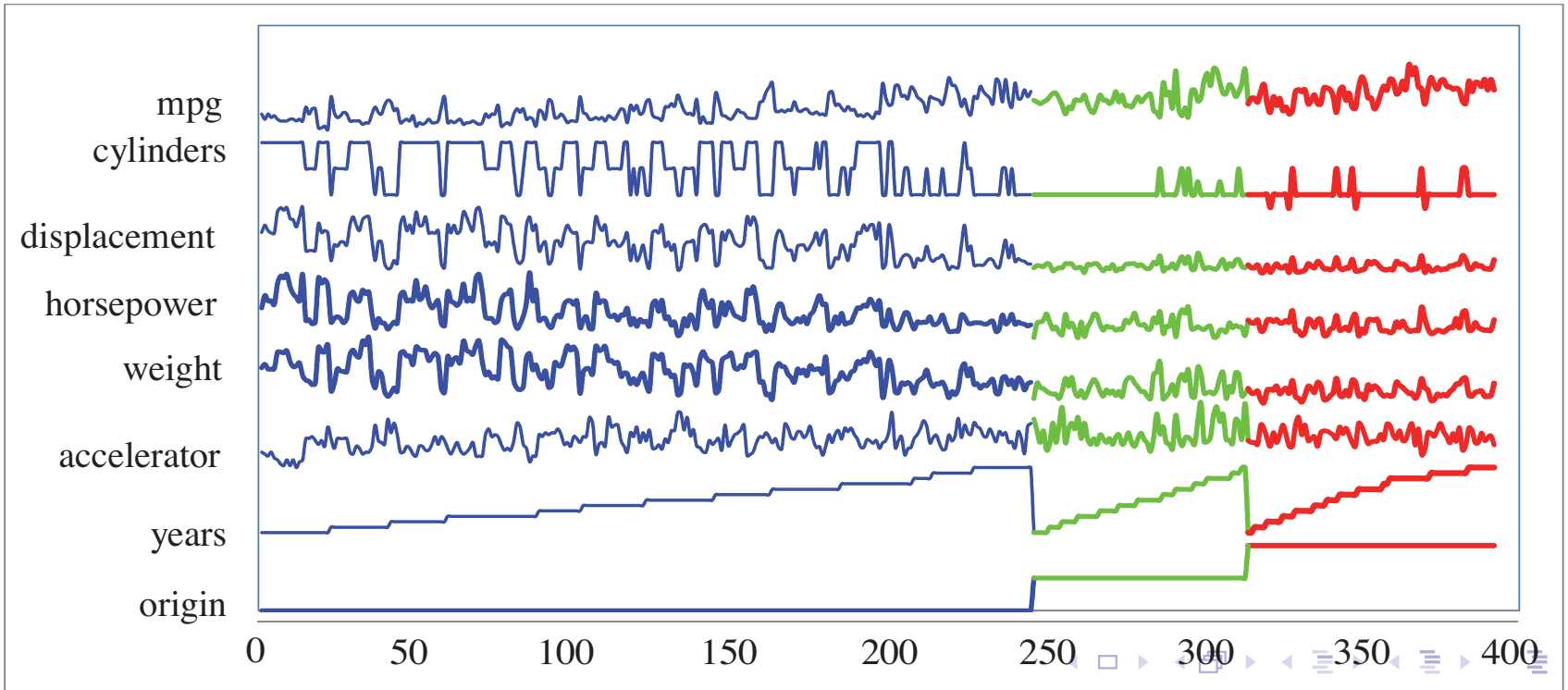
Virginica

Auto MPG Data Set

- The *Auto MPG* data set is the data on the car produced in the USA, Europe and Japan in 1970-1982 (398 cars). The cars are described by nine features:
  - MPG (miles per gallon) (x_1),
  - the number of cylinders (x_2),
  - displacement (x_3),
  - horsepower (x_4),
  - weight (x_5),
  - acceleration (x_6),
  - model year (x_7),
  - the origin (x_8),
  - the car name (x_9).
- The last two features are not numerical, therefore, they are not used in the visualization process. Therefore, the seven-dimensional ($n = 7$) points are used for visualization.

Multiline Graphs of Auto MPG data

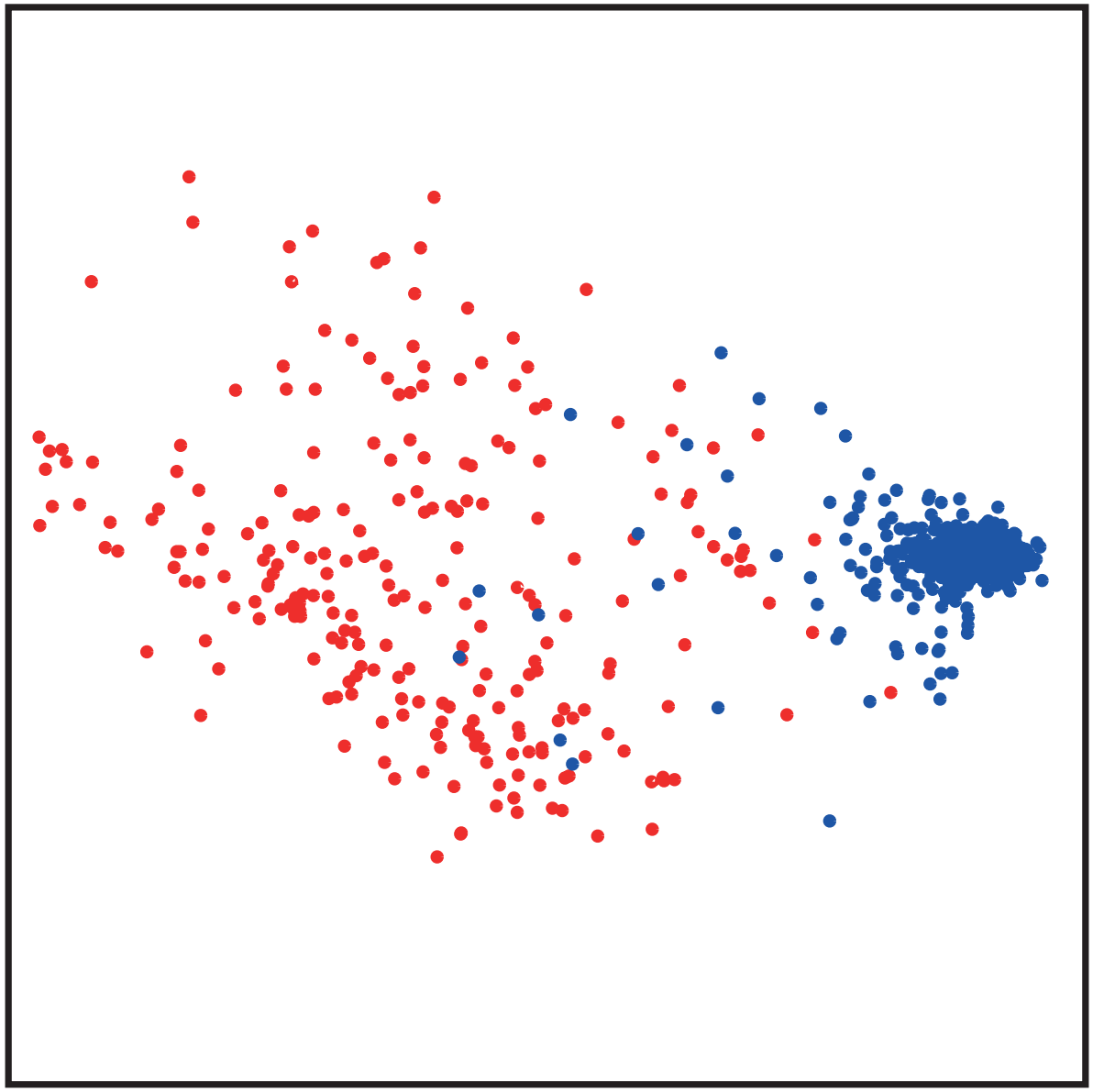
- MPG (miles per gallon) (x_1),
- the number of cylinders (x_2),
- displacement (x_3),
- horsepower (x_4),
- weight (x_5),
- acceleration (x_6),
- model year (x_7),
- the origin (x_8).



Breast Cancer Data Set

- The *Breast Cancer* data set was obtained from the University of Wisconsin Hospitals, USA. 699 observations of the breast cancer are collected. Each instance has one of the two possible classes: benign or malignant. There are nine features:
  - clump thickness (x_1),
  - uniformity of cell size (x_2),
  - uniformity of cell shape (x_3),
  - marginal adhesion (x_4),
  - single epithelial cell size (x_5),
  - bare nuclei (x_6),
  - bland chromatin (x_7),
  - normal nucleoli (x_8),
  - mitoses (x_9).
- There are some missing values of features, so the objects with missing values are eliminated from the data set for visualization. The visualized data set consists of 683 points ($m = 683$, $n = 9$).

Breast Cancer Data Visualized Using Principal Components

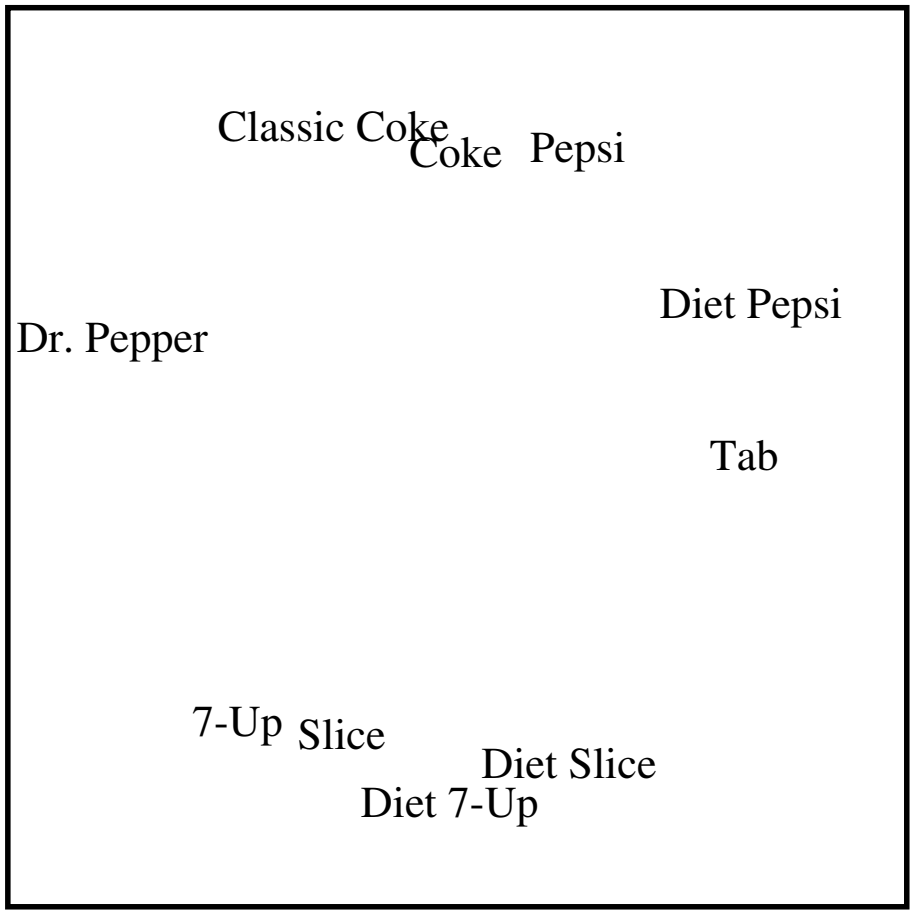


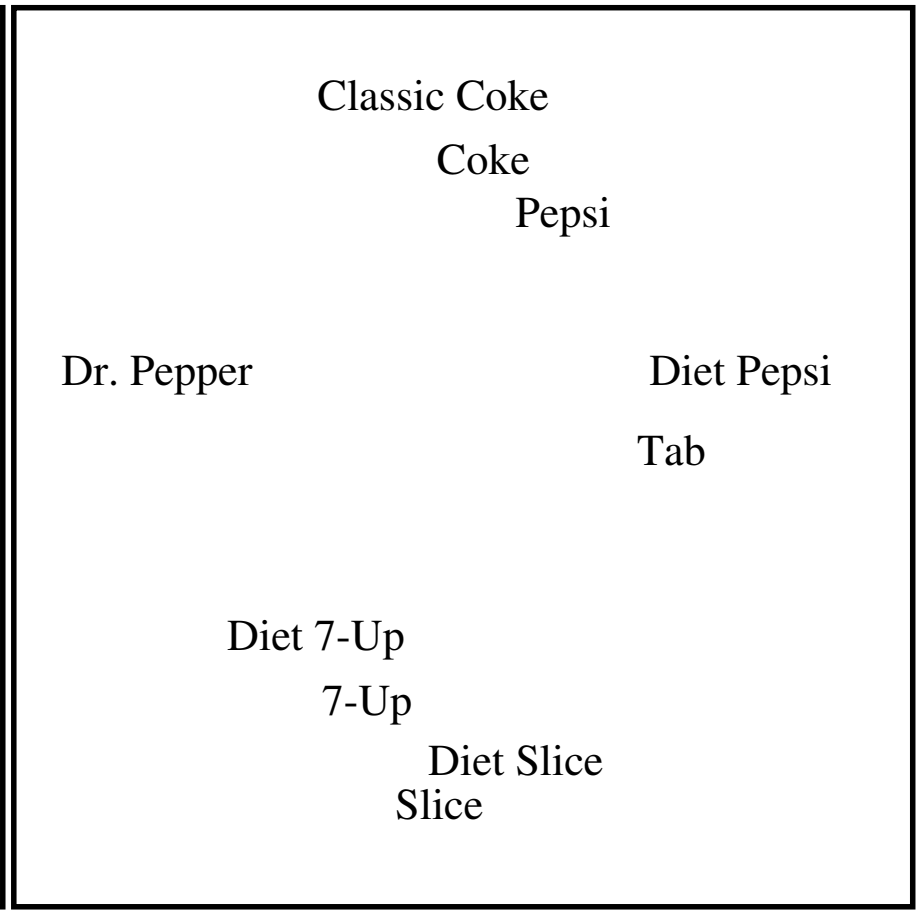
Soft Drinks Data Set

- The *Cola* data set is based on experimental testing of several soft drinks: Pepsi, Coke, Classic Coke, Diet Pepsi, Diet Slice, Diet 7-Up, Dr Pepper, Slice, 7-Up, Tab.
- 38 students have tested ten ($m = 10$) different brands of soft drinks.
- Each pair was judged on its dissimilarity in a nine-point scale (1 – very similar, 9 – completely different).
- The matrix of accumulated dissimilarities Δ_{cola}.

	1	2	3	4	5	6	7	8	9	10
1. Pepsi	0	127	169	204	309	320	286	317	321	238
2. Coke	127	0	143	235	318	322	256	318	318	231
3. Classic Coke	169	143	0	243	326	327	258	318	318	242
4. Diet Pepsi	204	235	243	0	285	288	259	312	317	194
5. Diet Slice	309	318	326	285	0	155	312	131	170	285
6. Diet 7-Up	320	322	327	288	155	0	306	164	136	281
7. Dr Pepper	286	256	258	259	312	306	0	300	295	256
8. Slice	317	318	318	312	131	164	300	0	132	291
9. 7-Up	321	318	318	317	170	136	295	132	0	297
10. Tab	238	231	242	194	285	281	256	291	297	0

Soft Drinks Data Visualized Using Multidimensional Scaling



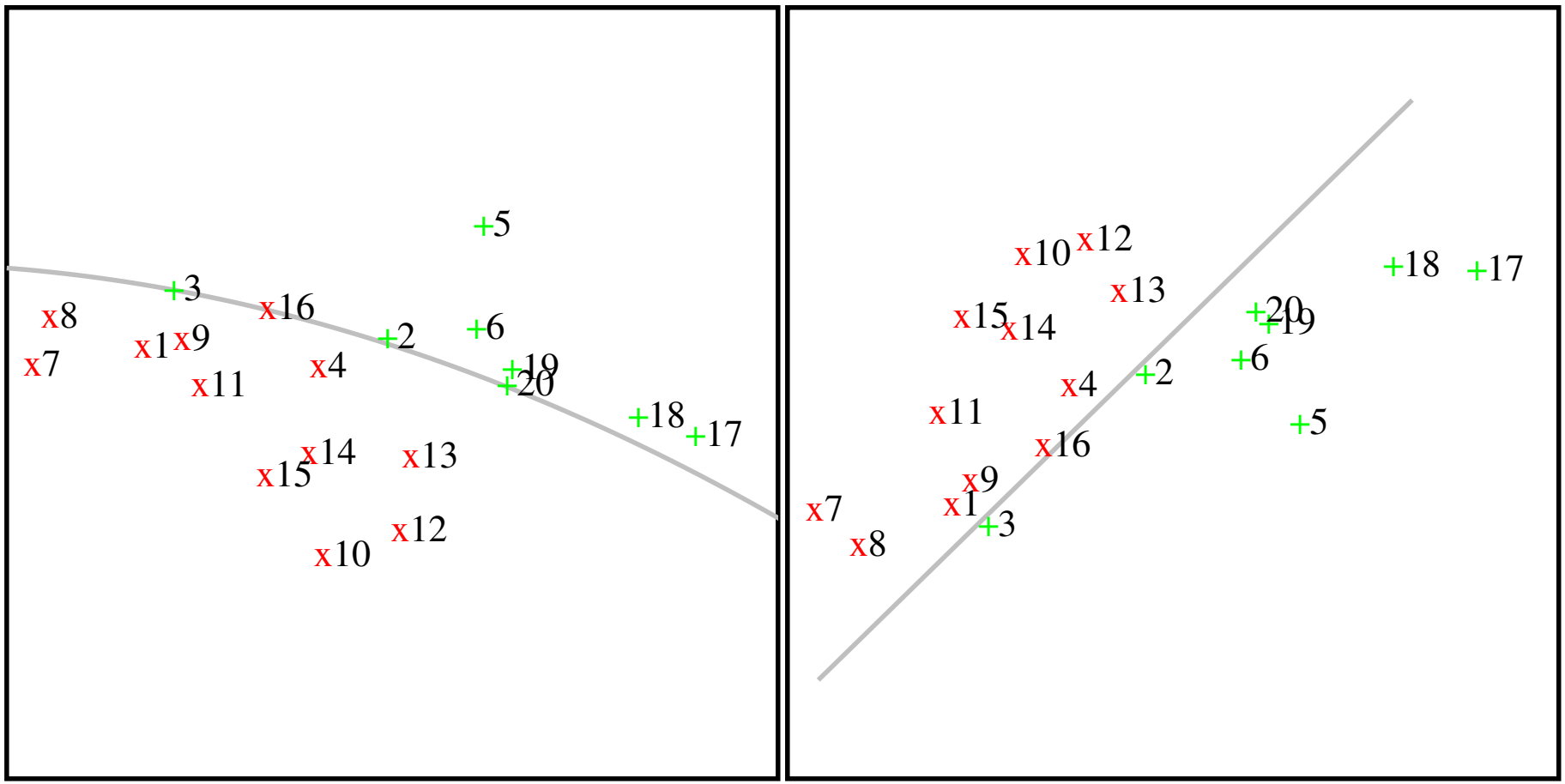


Pharmacological Data

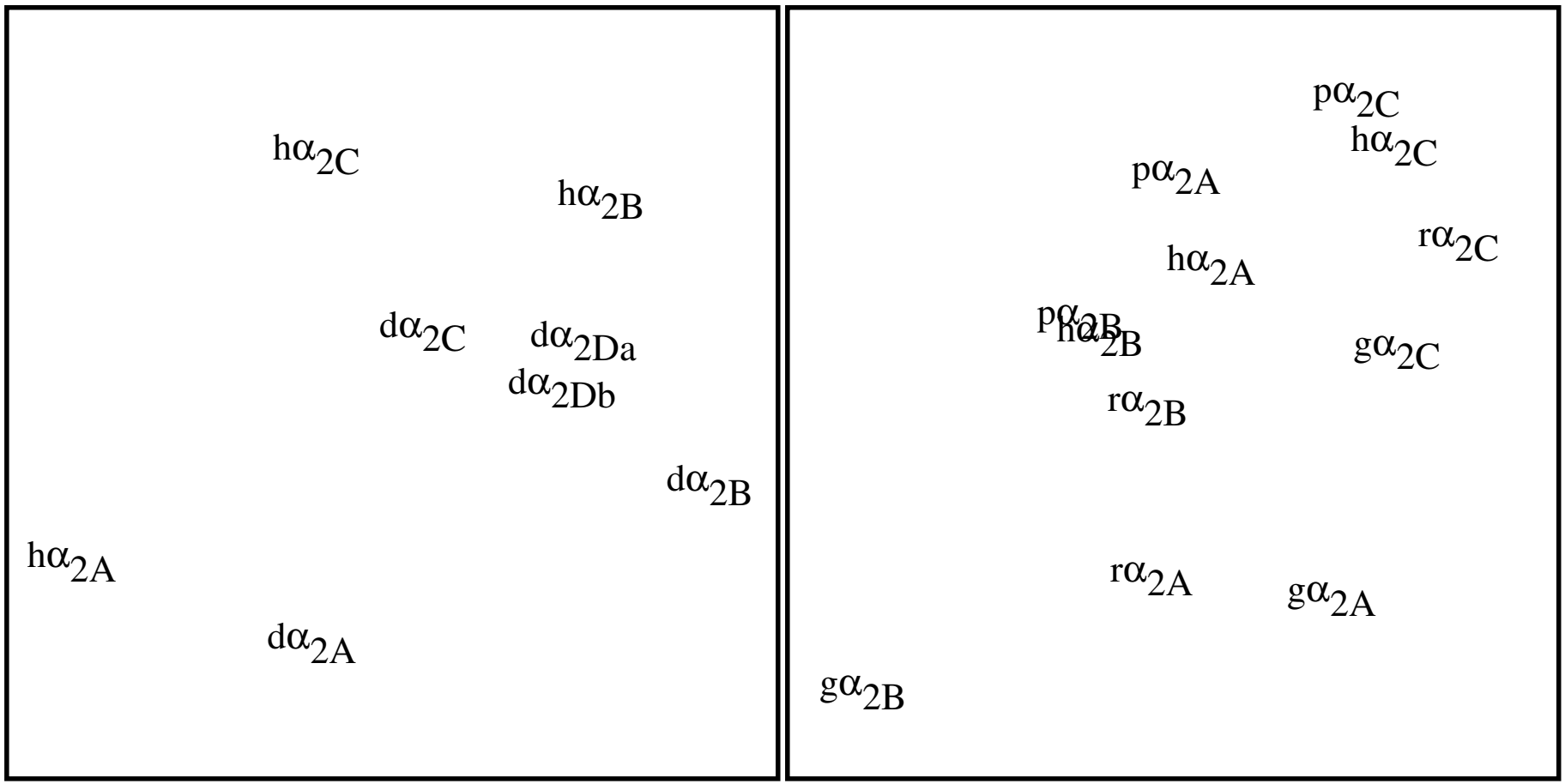
- Pharmacological data, e.g. the values of inhibition, are constants which under the given experimental conditions link the affinity of a given ligand to a given receptor protein.

	Proteins							
Ligands	hα_{2A}	zα_{2A}	hα_{2B}	zα_{2B}	hα_{2C}	zα_{2C}	zα_{2Da}	zα_{2Db}
1. Atipamezole	1.6	13	1.5	5.0	4.3	2.1	5.1	6.9
2. Clonidine	10	89	44	250	110	55	120	150
3. Dexmedetomidine	1.3	2.2	4.7	7.6	6.5	12	4.1	3.7
4. Idazoxan	17	85	24	40	17	17	52	94
5. Oxymetazoline	2.1	5.1	1100	1200	130	1300	1100	440
6. UK14,304	32	40	320	1200	190	700	260	280
7. L657.743	0.8	6.9	0.7	1.2	0.09	1.0	1.6	1.3
8. Rauwolscine	1.9	1.0	1.1	1.4	0.2	0.5	2.3	2.3
9. Yohimbine	5.9	5.2	7.5	9.3	4.6	3.4	6.4	4.0
10. Chlorpromazine	990	110	43	1.1	330	83	18	19
11. Clozapine	32	3.3	12	9.3	2.1	3.2	12	24
12. ARC239	2100	1800	9.6	36	66	280	55	44
13. Prazosin	1030	330	66	300	31	100	68	64
14. Spiperone	540	45	12	51	11	63	15	18
15. Spiroxatrine	320	150	2.4	93	3.1	35	11	11
16. WB-4101	5.4	11	60	51	1.9	19	31	16
17. 2-Amino-1-phenylethanol	1300	5400	4200	9400	8100	5100	3700	4000
18. Dopamine	2000	790	6300	4400	1200	3900	1300	1700
19. (–)-Adrenaline	150	140	710	910	130	1080	500	470
20. (–)-Noradrenaline	110	260	680	647	250	580	380	510

Ligands Visualized Using Multidimensional Scaling



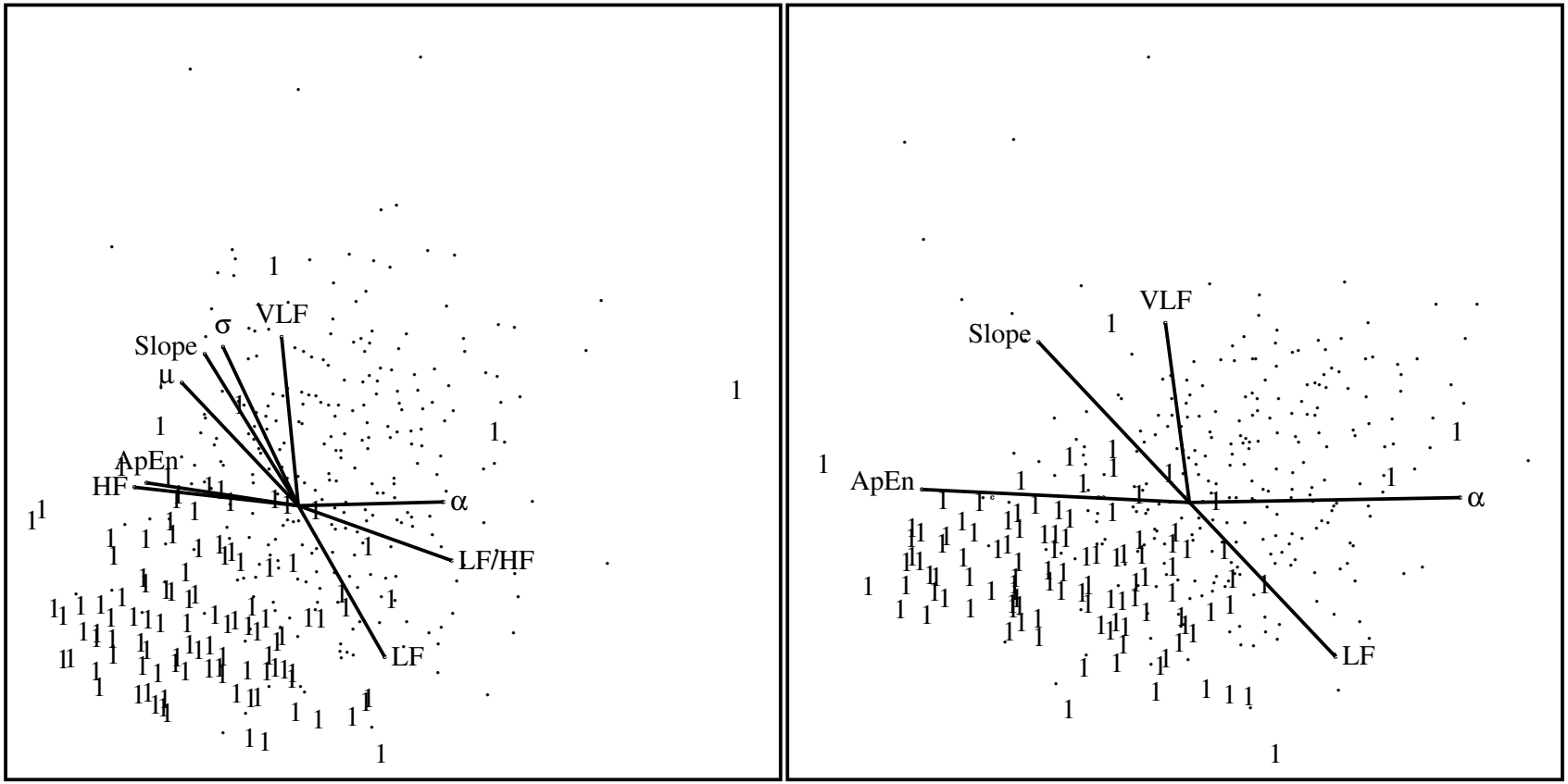
Proteins Visualized Using Multidimensional Scaling



Analysis of Heart Rate Oscillations with Respect to Characterization of Sleep Stages

- To diagnose sleep-related disorders and diseases, it is important to determine the sleep structure of a patient.
- The results of recent investigations show the dependence among characteristics of heart rate and sleep stages.
  - mean (x_1),
  - standard deviation (x_2),
  - very low frequency band 0.01–0.05 Hz (x_3),
  - low frequency band 0.05–0.15 Hz (x_4),
  - high frequency band 0.15–0.5 Hz (x_5),
  - ratio of normalized power in low and high frequency bands ($x_6 = x_4/x_5$),
  - approximate entropy that defines the complexity of behaviour of the time series (x_7),
  - fractal scaling exponent of a detrended fluctuation analysis (DFA) (x_8),
  - slope of a curve of a progressive detrended fluctuation analysis (x_9).

Heart Rate Data Visualized Using Multidimensional Scaling



Multidimensional Data Visualization

Direct Visualization Methods

Strategies for Multidimensional Data Visualization

- Direct visualization methods, where each feature, characterizing a multidimensional object, is represented in a visual form;
- Projection, so-called dimensionality reduction, methods, allowing us to represent the multidimensional data on a low-dimensional space. Artificial neural networks may also be used for visualizing multidimensional data. They realize various nonlinear projections.
- Our target is not to enumerate all methods and describe them in detail, but to present the most typical approaches and representatives of each group.

Direct Visualization Methods

- There is no formal mathematical criterion to estimate the visualization quality in direct visualization methods.
- All the features that characterize multidimensional data are represented in a visual form acceptable to a human.
- These methods may be classified into geometric, iconographic, and hierarchical visualization techniques.

Direct Visualization Methods

1. Geometric methods:
   a) scatter plots,
   b) matrix of scatter plots,
   c) multiline graphs,
   d) permutation matrix,
   e) survey plots,
   f) Andrews curves,
   g) parallel coordinates,
   h) radial visualization (RadViz) and its modifications GridViz and PolyViz.
2. Iconographic displays:
   a) Chernoff faces,
   b) star glyphs,
   c) stick figure,
   d) color icon.
3. Hierarchical displays:
   a) dimensional stacking,
   b) trellis display,
   c) hierarchical parallel coordinates.

Projection Methods

- Methods that allow us to represent multidimensional data from $\mathbb{R}^n$ in a low-dimensional space $\mathbb{R}^d$, $d < n$, are called projection (dimensionality reduction) methods.
- If the dimensionality of the projection space is small enough ($d = 2$ or $d = 3$), these methods may be used to visualize the multidimensional data. In such a case, the projection space can be called a *display*, *embedding* or *image space*.
- The projection methods usually invoke formal mathematical criteria by which the projection distortion is minimized.

Projection Methods

1. Linear projection methods:
   a) principal component analysis,
   b) linear discriminant analysis,
   c) projection pursuit.
2. Nonlinear projection methods:
   a) multidimensional scaling,
   b) locally linear embedding,
   c) isometric feature mapping,
   d) principal curves.

Direct Visualization

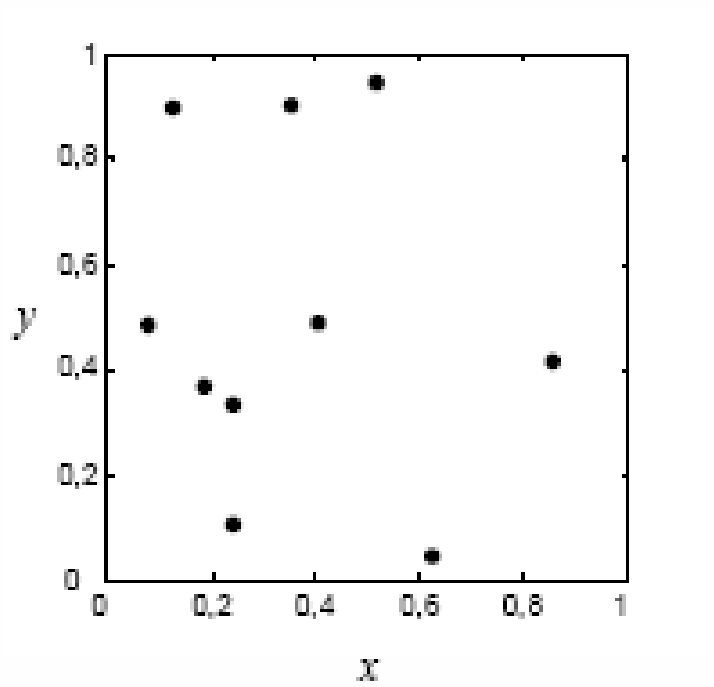
- The direct data visualization is a graphical presentation of the data set that provides a qualitative understanding of the information contents in a natural and direct way.
- The commonly used methods are scatter plot matrices, parallel coordinates, Andrews curves, Chernoff faces, stars, dimensional stacking, etc.
- The direct visualization methods do not have any defined formal mathematical criterion for estimating the visualization quality.
- Each of the features $x_1, x_2, \ldots, x_n$ characterizing the object $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, is represented in a visual form acceptable for a human being.

Geometric Methods

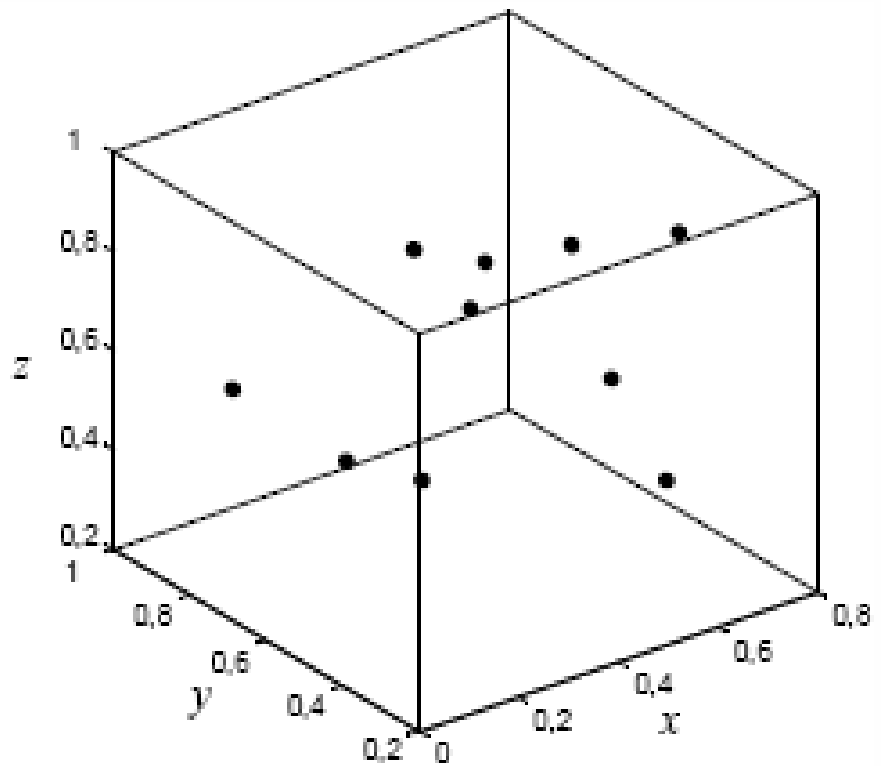
- Geometric visualization methods are the methods where multidimensional points are displayed using the axes of the selected geometric shape.

Scatter Plots

- *Scatter plots* are one of the most commonly used techniques for data representation on a plane $\mathbb{R}^2$ or space $\mathbb{R}^3$. Points are displayed in the classic (x, y) or (x, y, z) format.
- Usually, the two-dimensional ($n = 2$) or three-dimensional ($n = 3$) points are represented by this technique.



a)

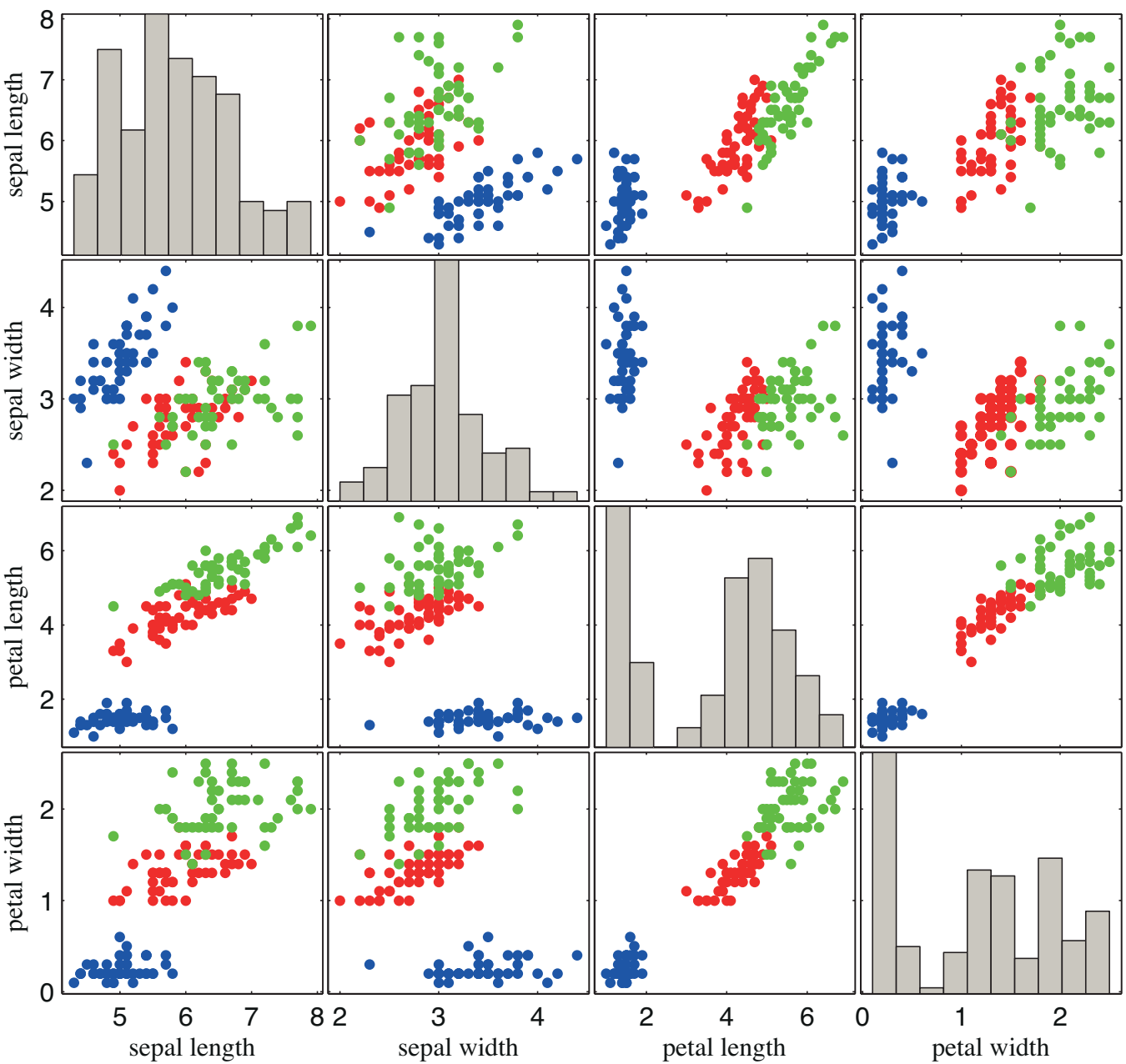


b)

Matrix of Scatter Plots

- Using a *matrix of scatter plots*, the scatter plots can be applied to visualize more higher dimensionality data.
- The matrix of scatter plots is an array of scatter plots displaying all possible pairwise combinations of features.
- If n-dimensional data are analyzed, the number of scatter plots is equal to $\frac{n(n-1)}{2}$.
- In the diagonal of the matrix of scatter plots, a graphical statistical characteristic of each feature can be presented, for example, a histogram of values.
- The matrix of scatter plots is useful for observing all possible pairwise interactions between features.
- The scatter plots can also be positioned in a non-array format (circular, hexagonal, etc.).
- The matrix of scatter plots of the Iris data is presented. We can see that Setosa flowers (blue) are significantly different from Versicolor (red) and Virginica (green).

Scatter Plot Matrix of the Iris Data



Setosa

Versicolor

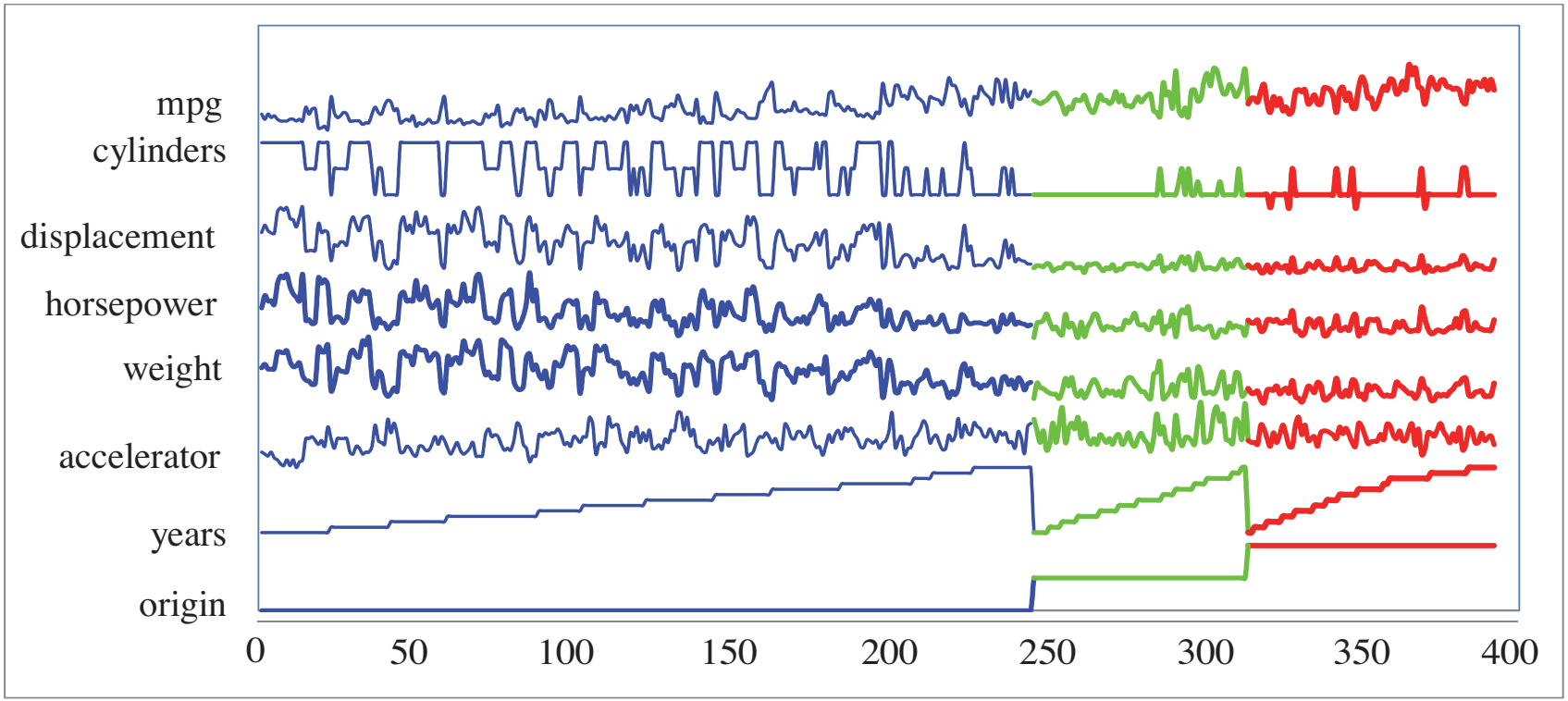
Virginica

Multiline Graphs

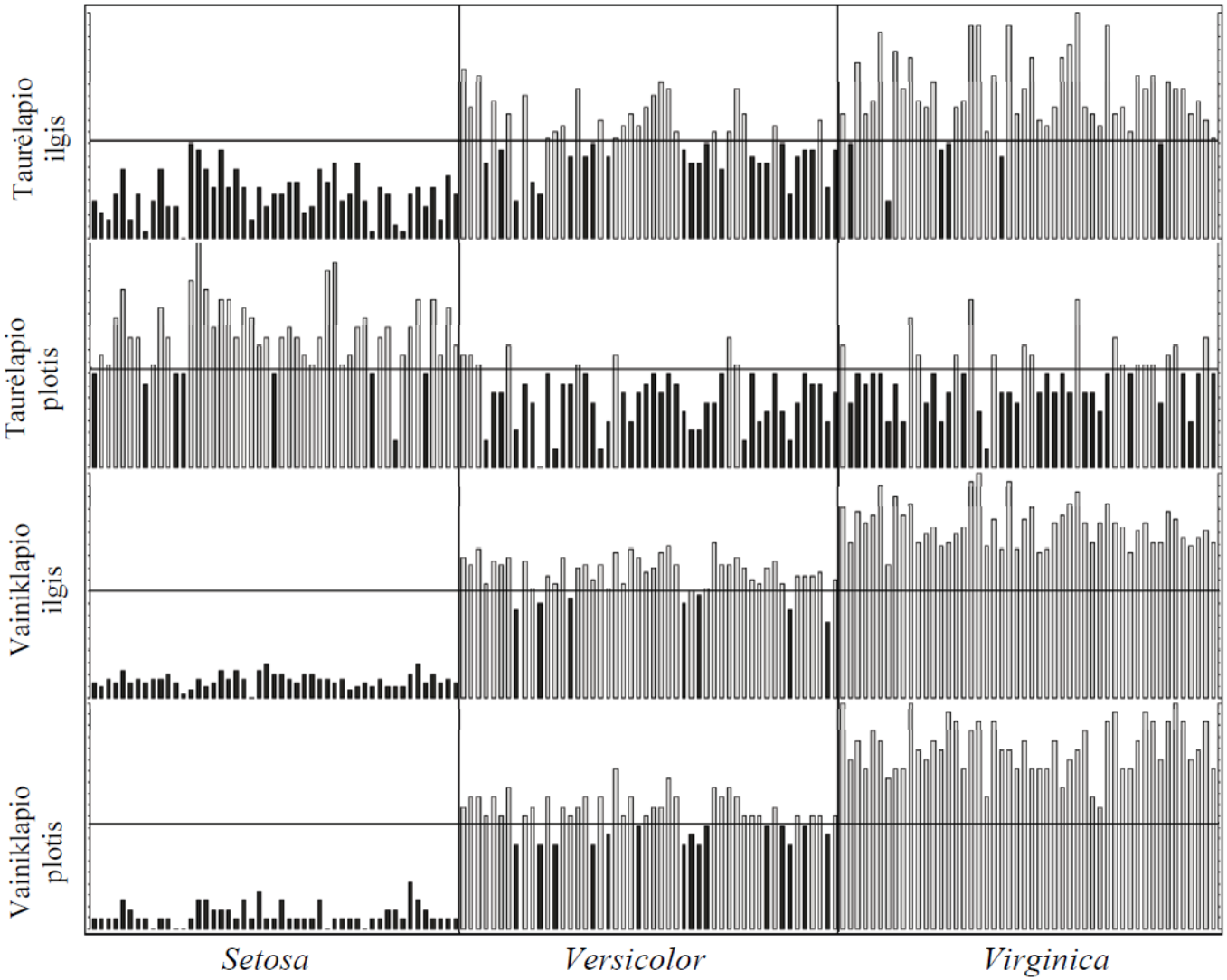
- In *Multiline graphs*, we draw n curves (line graphs) that represent the features depending on the order number of objects.
- An example for Auto MPG data is presented in the figure. The data set is the data on the car produced in the USA, Europe and Japan in 1970-1982 (398 cars). The cars are described by nine features:
  - MPG (miles per gallon) (x_1),
  - the number of cylinders (x_2),
  - displacement (x_3),
  - horsepower (x_4),
  - weight (x_5),
  - acceleration (x_6),
  - model year (x_7),
  - the origin (x_8),
  - the car name (x_9).

Multiline Graphs of the Auto MPG Data

- The data are aligned according to origin (USA, Japan, Europe).

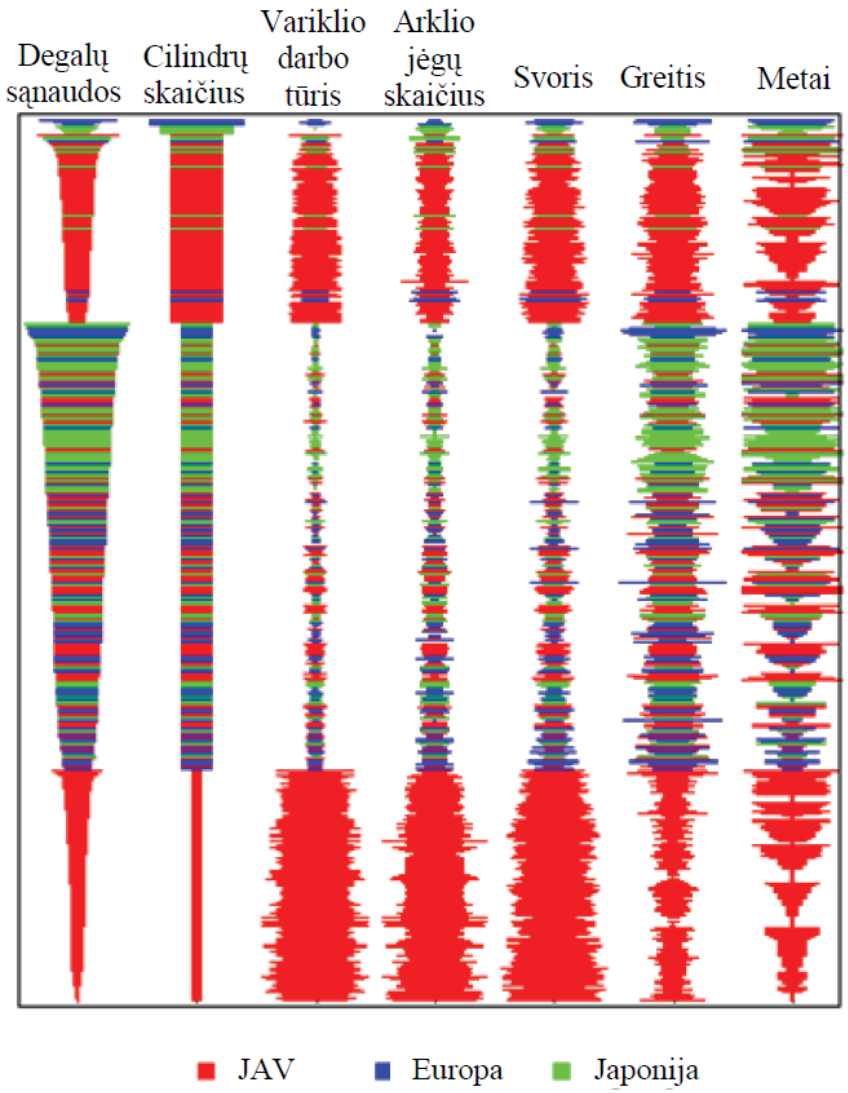


Permutation Matrix



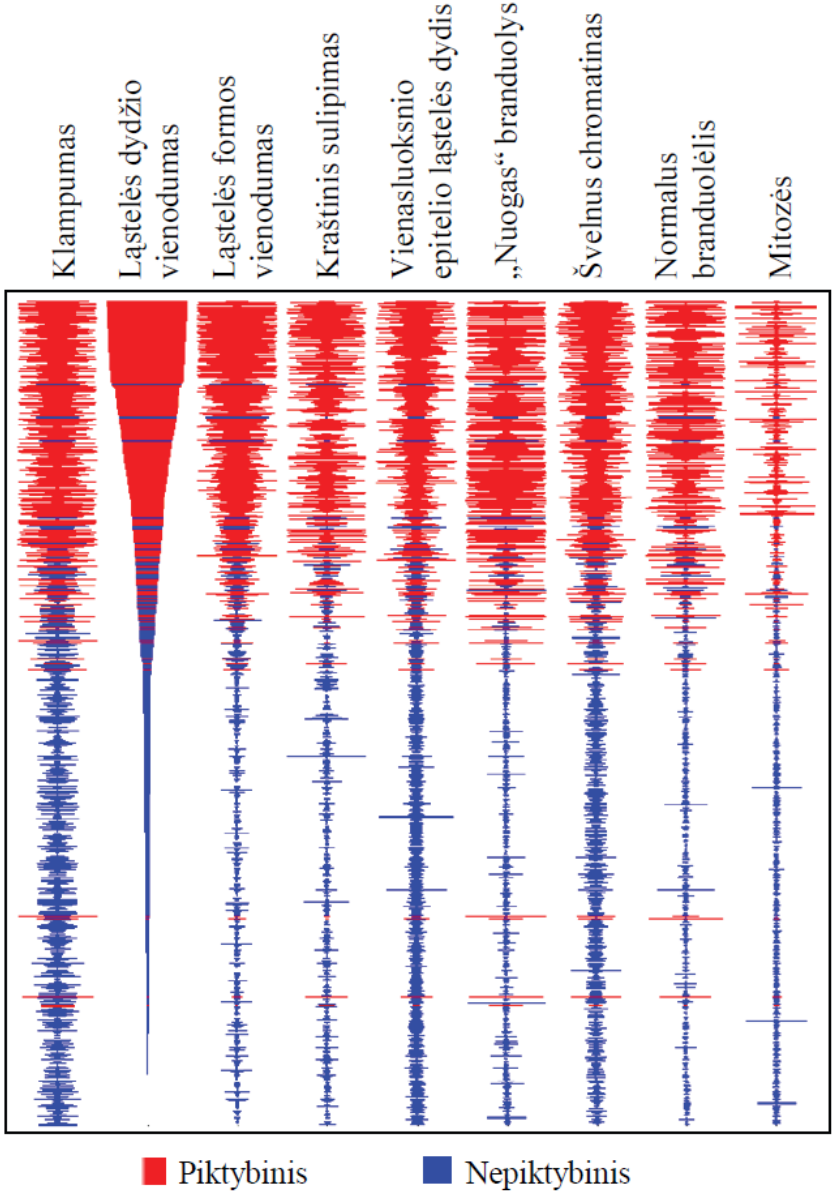
Survey Plot of Auto MPG Data

- Sorted according to the number of cylinders and MPG.



Survey Plot of Breast Cancer Data

- Sorted according to uniformity of cell size.



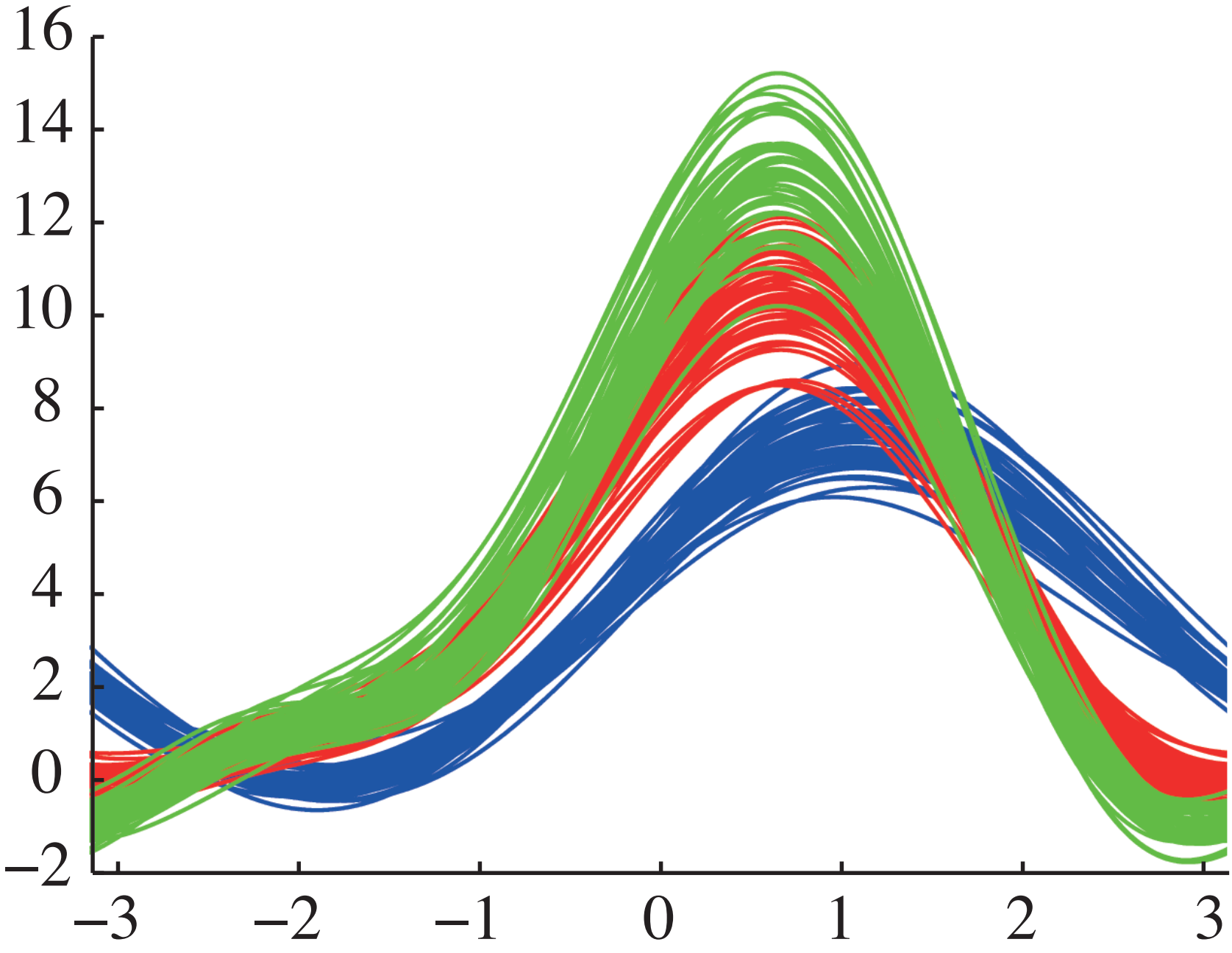
Andrews Curves

- *Andrews curves* plot each n-dimensional point X_i, $i \in \{1, \ldots, m\}$ as a curve (sum of sinusoids), using the function:

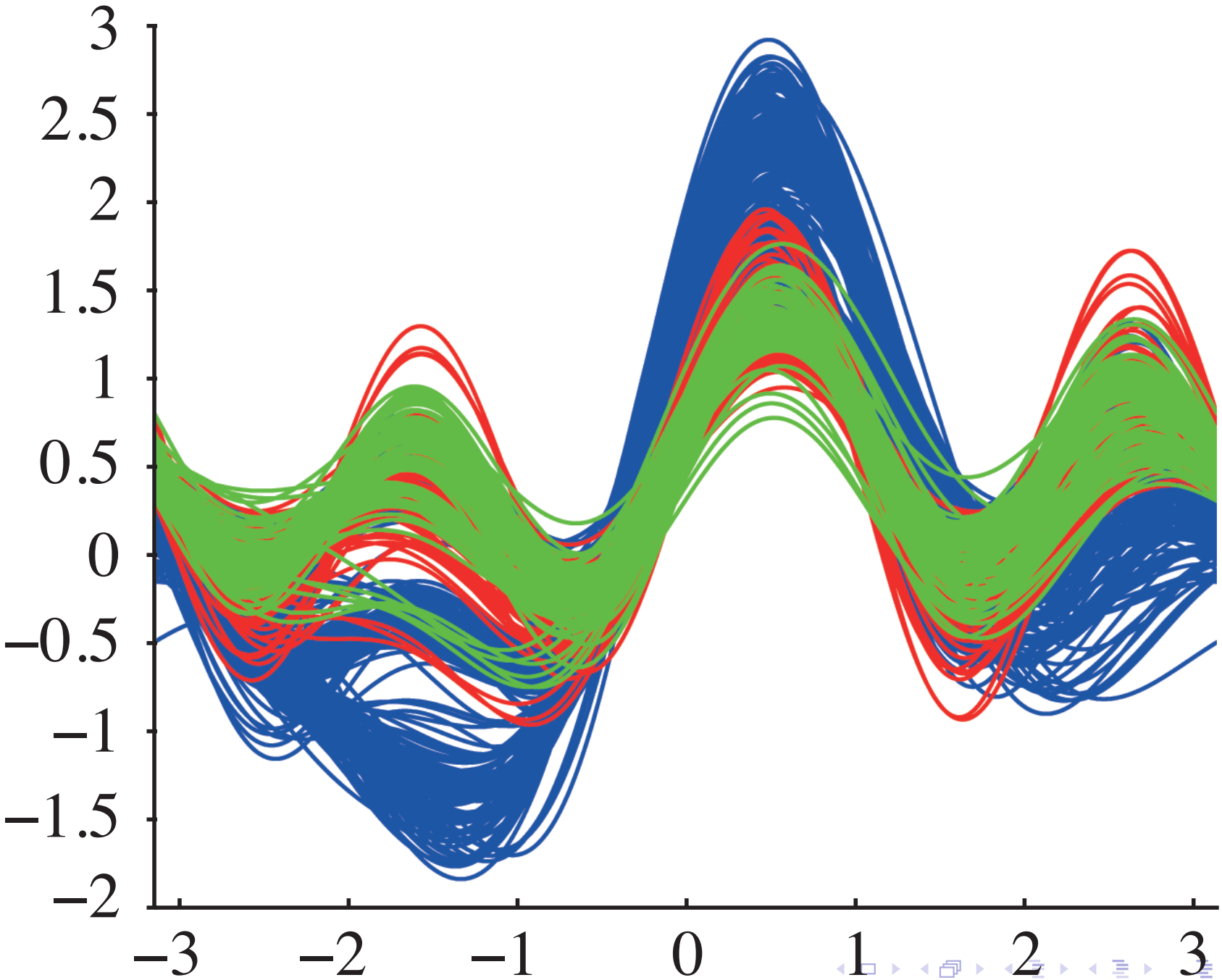
$$f_i(t) = \frac{x_{i1}}{\sqrt{2}} + x_{i2}\sin(t) + x_{i3}\cos(t) + x_{i4}\sin(2t) + x_{i5}\cos(2t) + \cdots, \quad -\pi <$$

  where $x_{i1}, x_{i2}, \ldots, x_{in}$ are the values of coordinates of the point X_i.
- Andrews curves of the Iris and Auto MPG data sets are presented. The curves are obtained by the *Matlab* system (`http://www.mathworks.com`), where different species of irises and classes of auto by the origin are painted in different colors.
- An advantage of this method is that it can be used for the analysis of data of a high dimensionality n. A shortcoming is that when visualizing a large data set, i.e. with large enough m, it is difficult to comprehend and interpret the results.

Andrews Curves of Iris Data

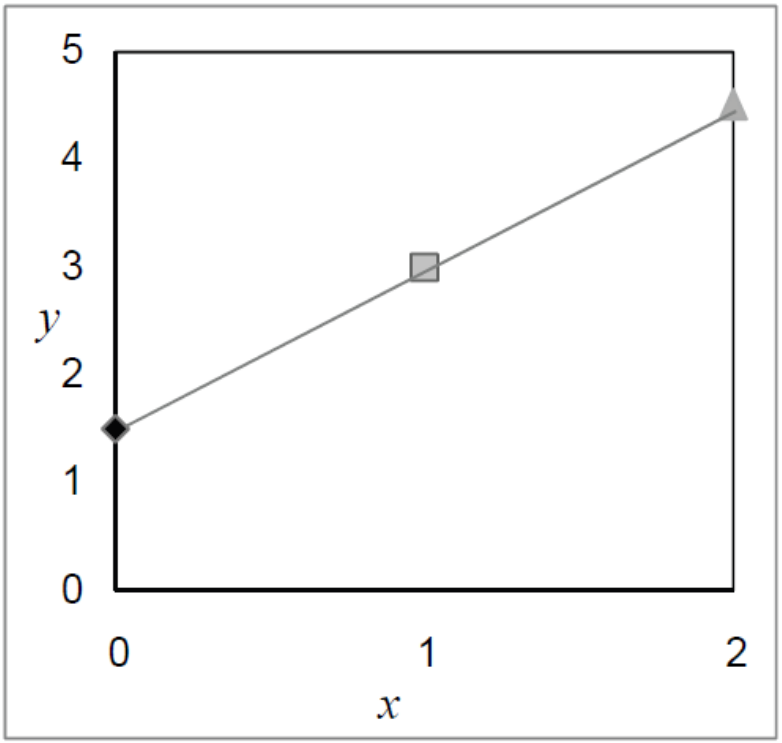


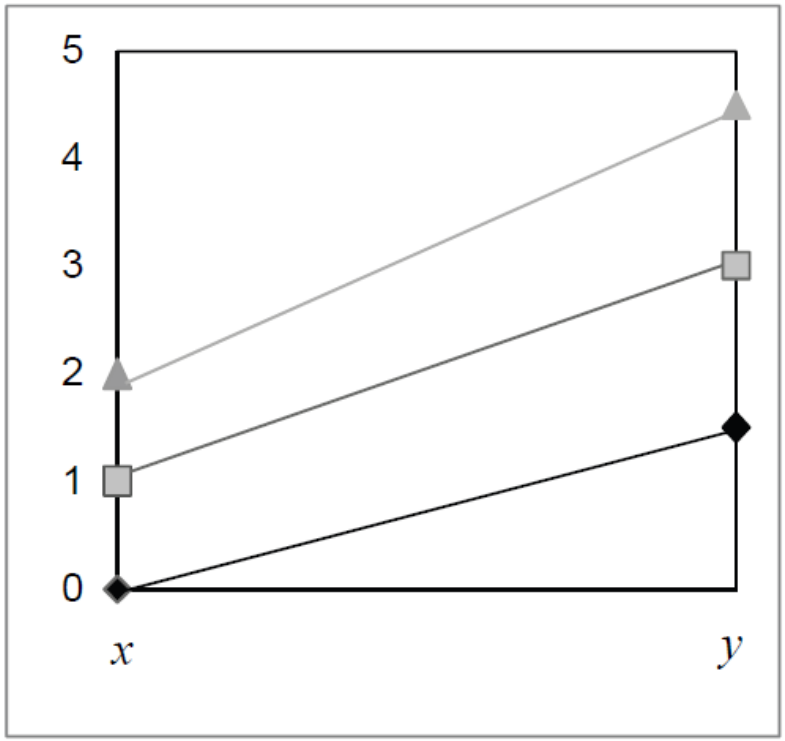
Andrews Curves of Auto MPG Data



Parallel Coordinates

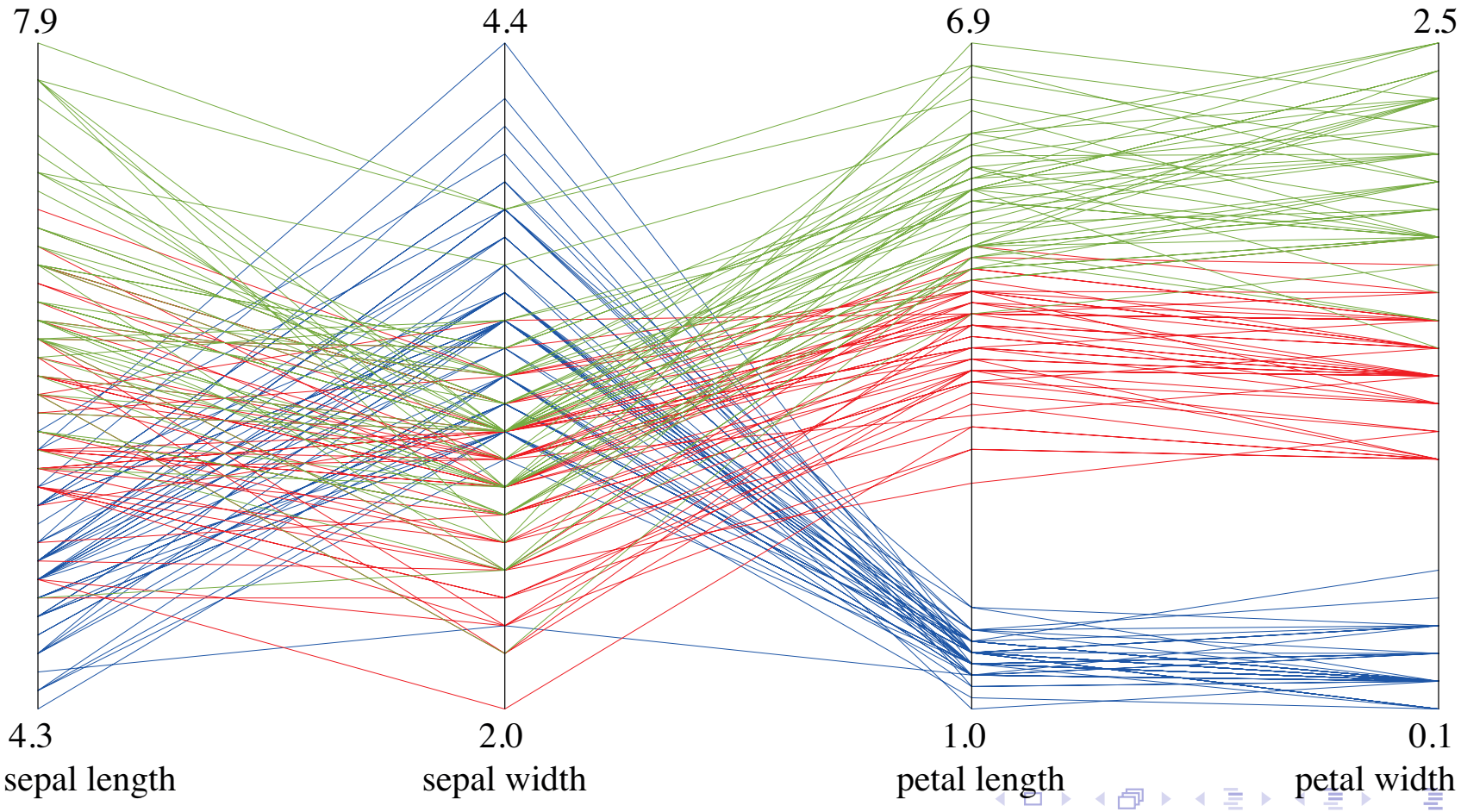
- *Parallel coordinates* as a way of visualizing multidimensional data are proposed by Inselberg in 1981.
- In this method, coordinate axes are shown as parallel lines that represent features.
- An n-dimensional point is represented as $n - 1$ line segments, connected to each of the parallel lines at the appropriate feature value.





Iris Data Represented on the Parallel Coordinates

- The image is obtained using the system *Orange*.
- Different colors correspond to the different species.
- We see that the species are distinguished best by the petal length and width. It is difficult to separate the species by the sepal length and width.



Parallel Coordinates

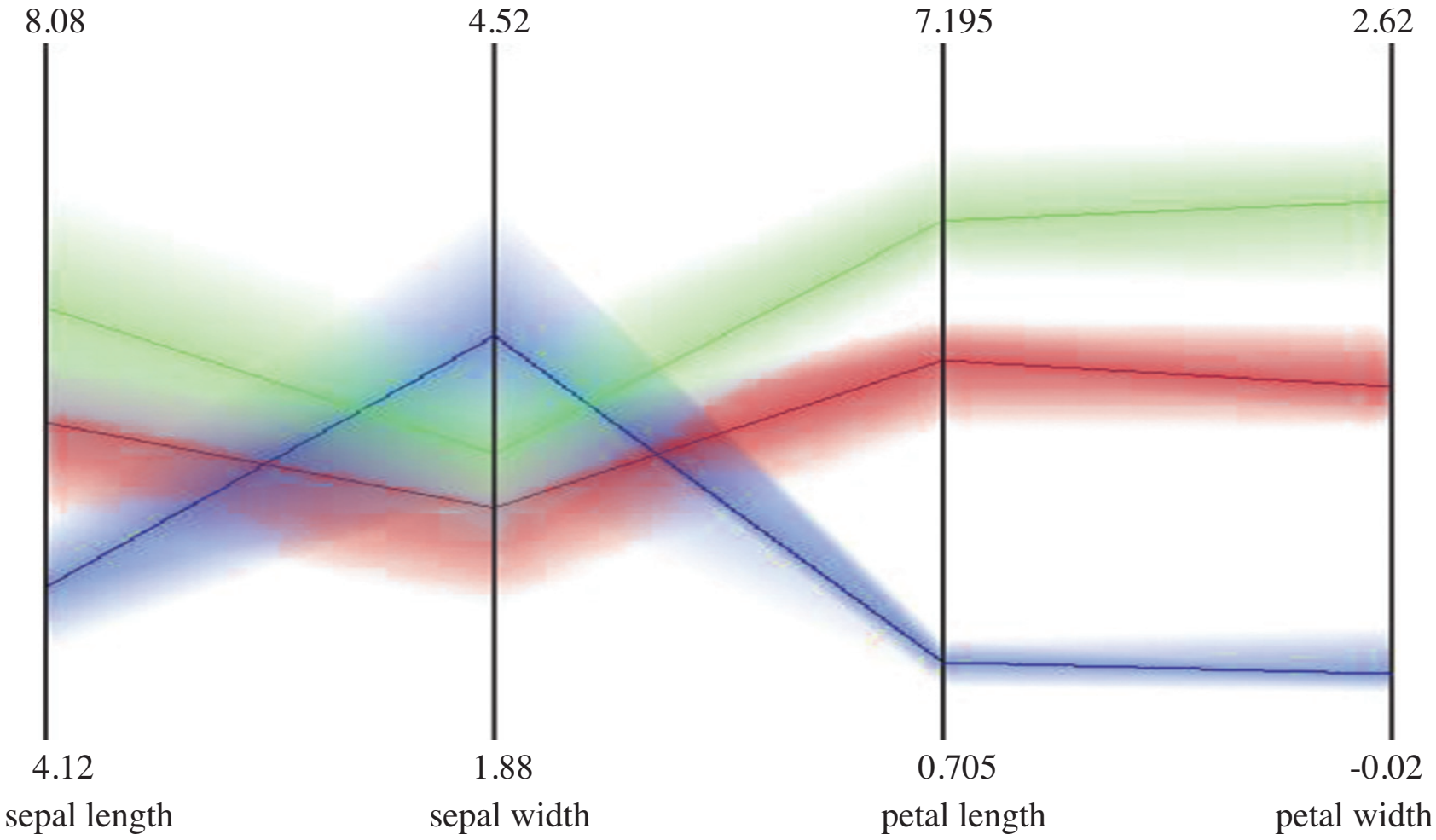
- The parallel coordinate method can be used for visualizing data of high dimensionality.
- However, then the coordinates must be spaced much nearer one to the other. When the coordinates are dense, it is difficult to perceive the data structure.
- When displaying a large data set, i.e. when the number m of objects is large, the interpretation of the results is very complicated, often it is almost impossible.

Hierarchical Parallel Coordinates

- *Hierarchical parallel coordinates* are one of variations of the parallel coordinates.
- When visualizing a large data set by the hierarchical parallel coordinates, the number of overlapping lines, obtained by the parallel coordinates, decreases.
- The data are represented on the hierarchical parallel coordinates as follows:
  - First, the data are grouped into some clusters by one of clustering methods;
  - Afterwards, the data are represented on the parallel coordinates, the centers of clusters are highlighted; the color intensity of the members of clusters depends on how far they are from the cluster center; different clusters are displayed by different colors.
- Hierarchical parallel coordinates allow a visual presentation of clustered data.

Iris Data Represented on the Hierarchical Parallel Coordinates

- The image is obtained using the system *Xmdv*.

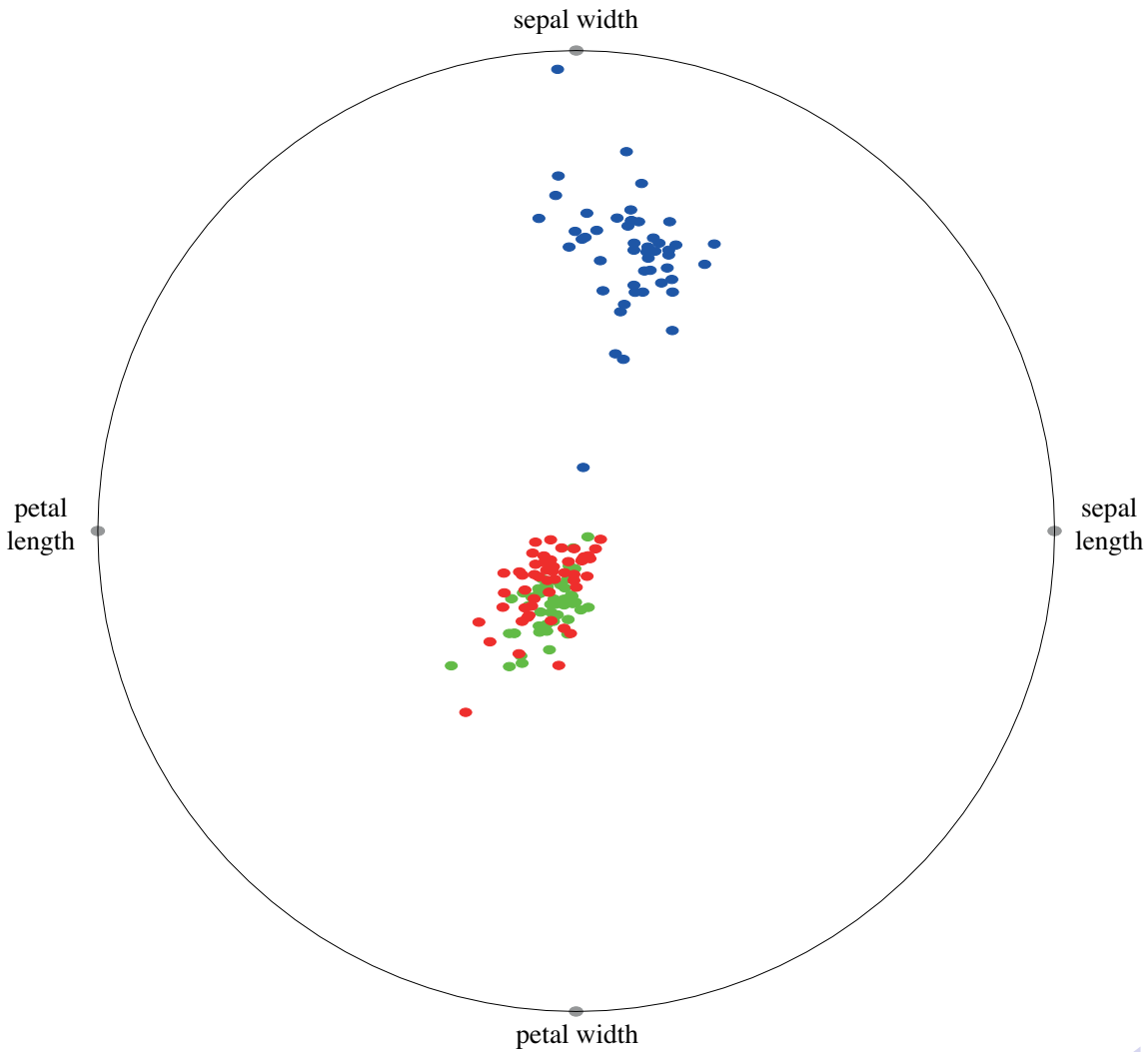


Radial Visualization – RadViz

- In *Radial visualization* (*RadViz*) method a circle is drawn and n so-called dimensional anchors representing features are fixed on this circle uniformly.
- The spring paradigm is applied to display multidimensional data. n springs are allocated to each n-dimensional object. One end of all the n springs is connected among them, other ends of the springs are connected to different dimensional anchors. The position of connection of n springs represents one object.
- The spring constants have the values of features of multidimensional objects. The values of features should be normalized in the range [0, 1] so that the minimal value of each feature were equal to 0, and the maximal one were equal to 1. Each object is displayed as a point in the position that produces a sum of spring forces equal to 0.

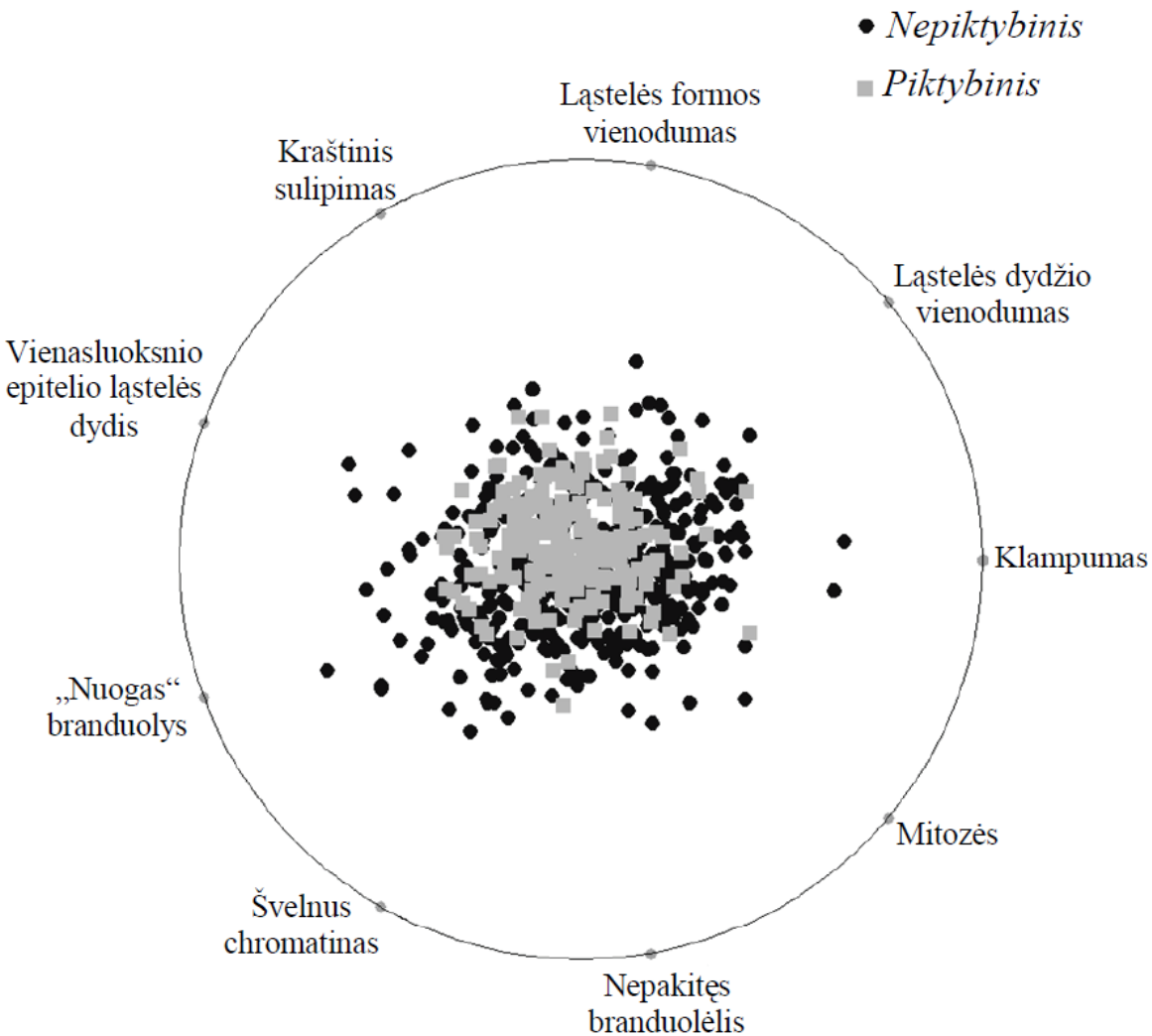
Iris data visualized by the RadViz method

- The petal length, petal width, sepal length, and sepal width are dimensional anchors. The image is obtained using the system *Orange*.



Breast Cancer Data Visualized by the RadViz Method

- Most of the malignant cases concentrate in the center, however, it is almost impossible to separate them from the benign cases.



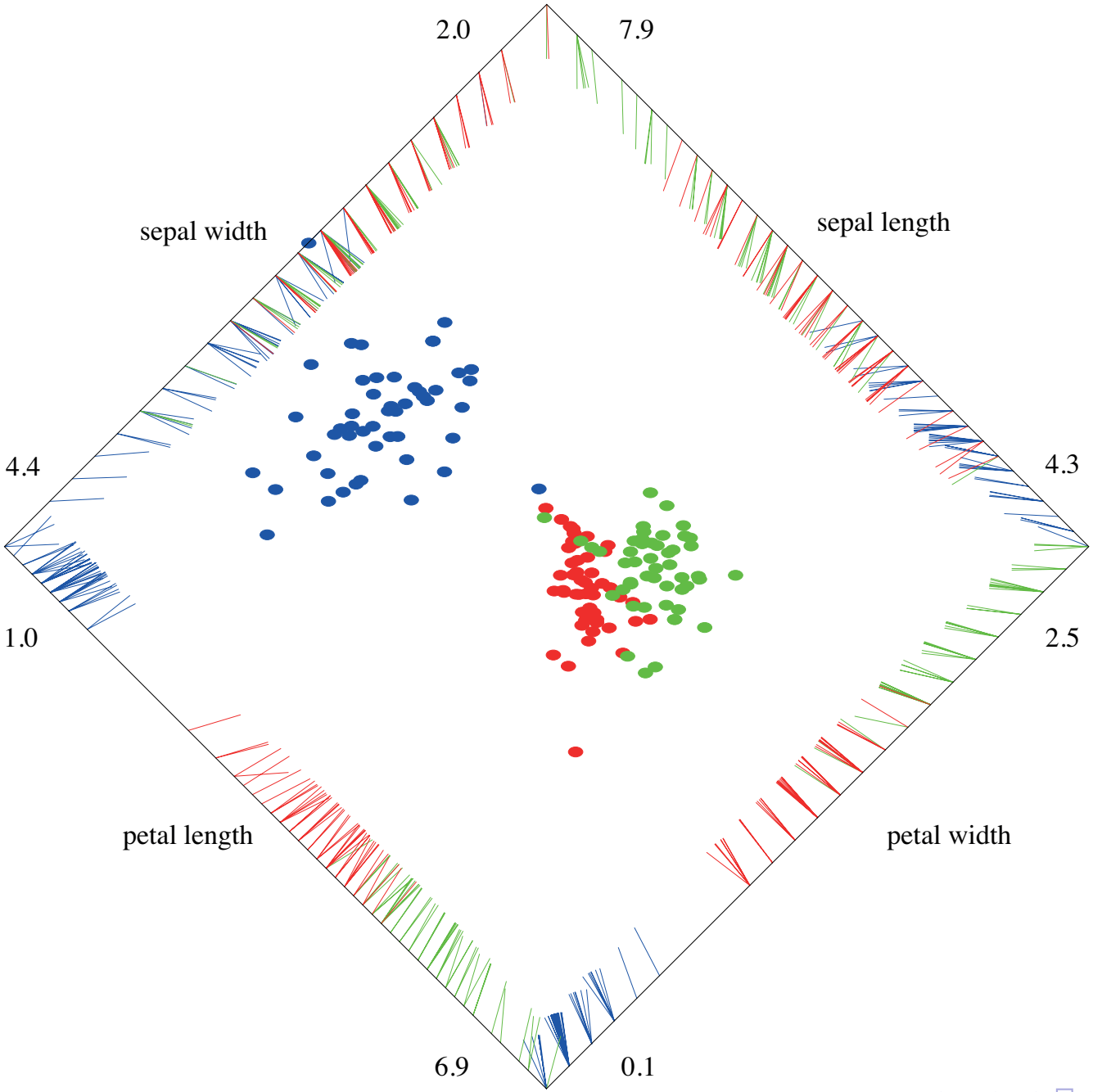
RadViz Modifications: GridViz

- Some modifications of the *RadViz* method are developed: *grid visualization* (*GridViz*) and *polygon visualization* (*PolyViz*).
- *GridViz* places the dimensional anchors (fixed spring end) on a rectangular grid but not on a circle. The spring paradigm is the same as in *RadViz*: the points are plotted where the sum of the spring forces is zero.
- In the *GridViz* case, feature labeling is difficult, but the displayed data dimensionality can be much higher.

RadViz modifications: PolyViz

- A shortcoming of *RadViz* is that *n*-dimensional objects with quite different values of features can appear at the same point.
- If the dimensional anchors are segments of lines, the overlapping of points is reduced. The segments are called anchor segments.
- The *Polygon visualization* (*PolyViz*) method was developed for this purpose.
- In the figure, springs start from the points, corresponding to the values of a particular feature on the anchor segment.

Iris Data Visualized by the PolyViz Method



Iconographic Displays

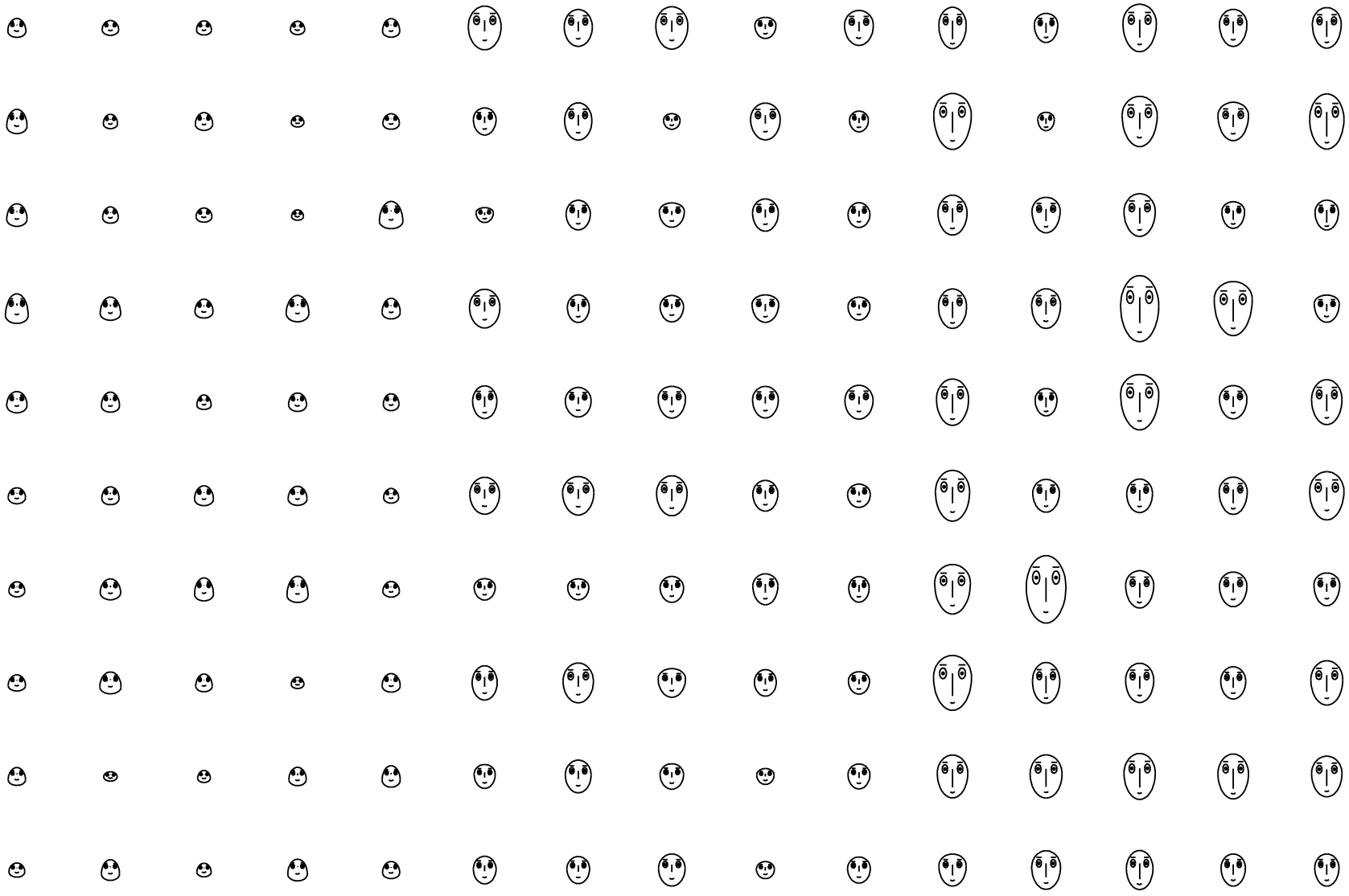
- The aim of visualization of multidimensional data is not only to map the data onto a two- or three-dimensional space, but also to help perceiving them.
- The second aim may be achieved visualizing multidimensional data by *iconographic display* methods. They are also called *glyph* methods.
- Each object that is defined by the n features is displayed by a glyph. Color, shape, and location of the glyph depend on the values of features.
- The most famous methods are *Chernoff faces* and the *star* method, however some methods of more complicated other glyphs may be used as well.

Chernoff Faces

- *Chernoff faces* are designed by Chernoff for visualization of multidimensional data.
- In Chernoff faces, data features are mapped to facial features, such as the angle of eyes, the width of a nose, etc.

Iris Data Visualized by Chernoff Faces

- Sepal length corresponds to the size of face, sepal width – shape of forehead, petal length – shape of jaw, and petal width – length of nose. *Matlab* is used to obtain the image.

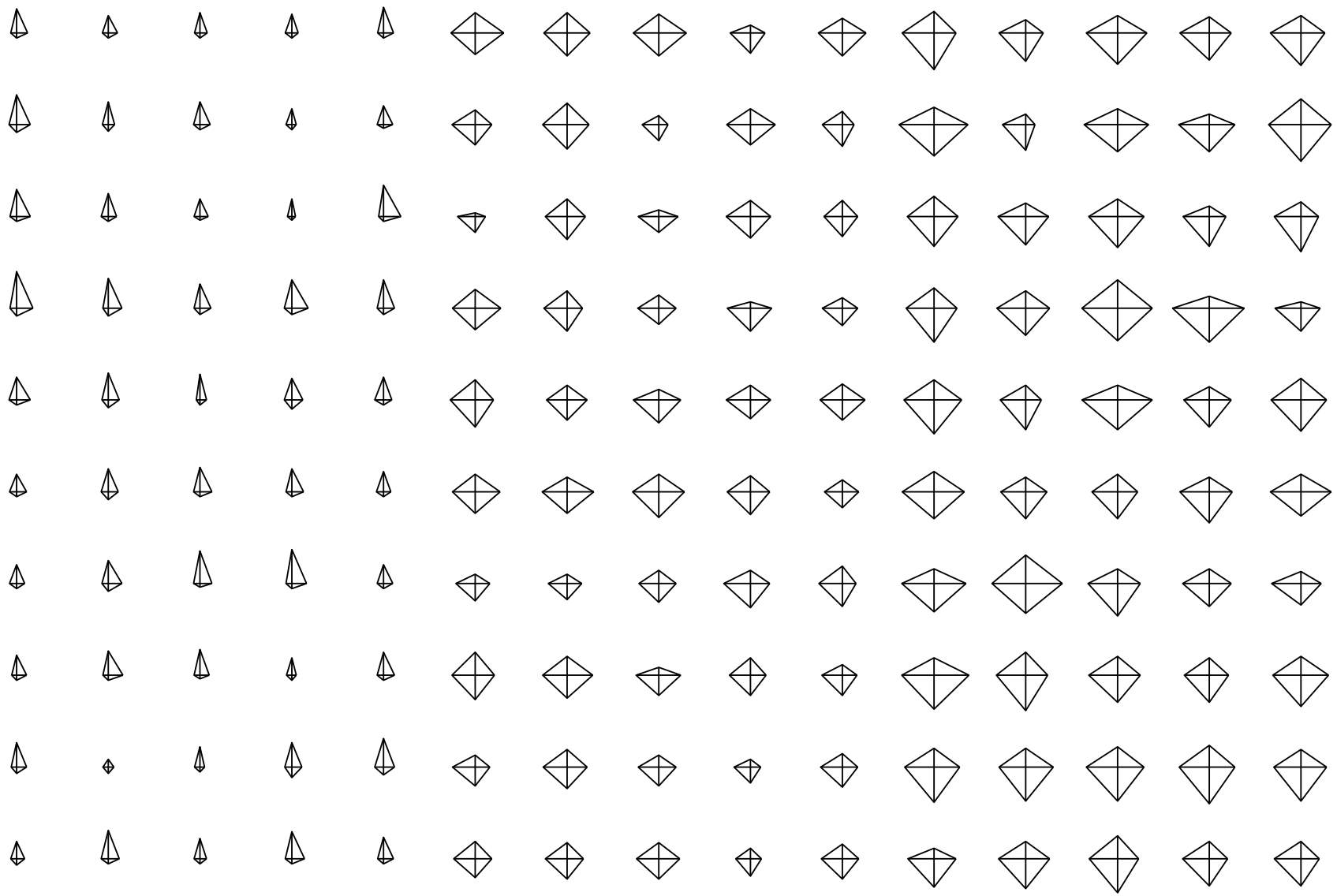


Star Glyphs

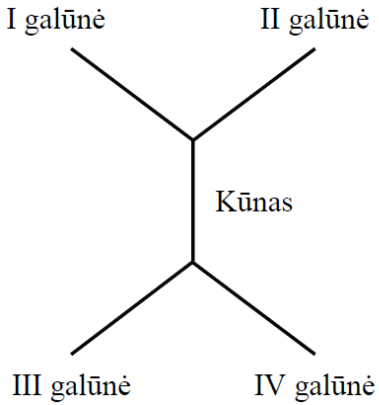
- Other glyphs commonly used for data visualization are *stars*.
- Each object is displayed by a stylized star.
- In the star plot, the features are represented as spokes of a wheel circle, but their lengths correspond to the values of features.
- The angles between the neighboring spokes are equal.
- The outer ends of the neighboring spokes are connected by line segments.

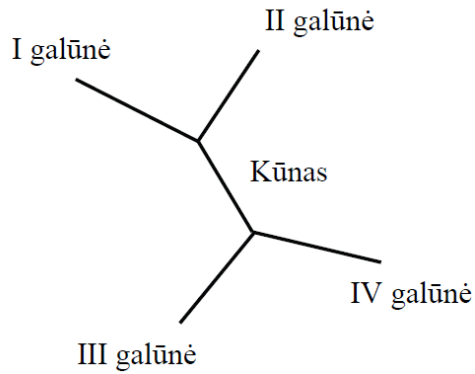
Iris Data Set Visualized by Star Glyphs

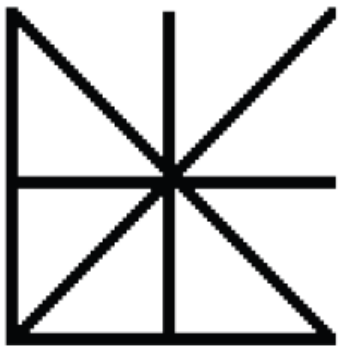
- The stars, corresponding to Setosa irises, are smaller than the other. The larger stars correspond to Virginica irises.

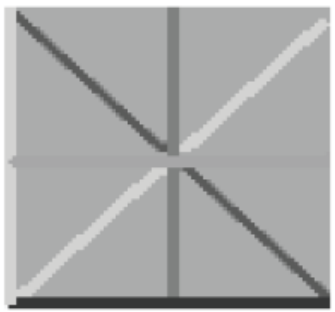


Stick Figure and Color Icon









Hierarchical Displays

- *Hierarchical displays* create a structure of an image such that some features are embedded in displays of other features.
- Visualization of some features is displayed in the structure depending on the values of other features.
- Here we introduce two such techniques: *dimensional stacking* and *trellis display*.

Dimensional Stacking

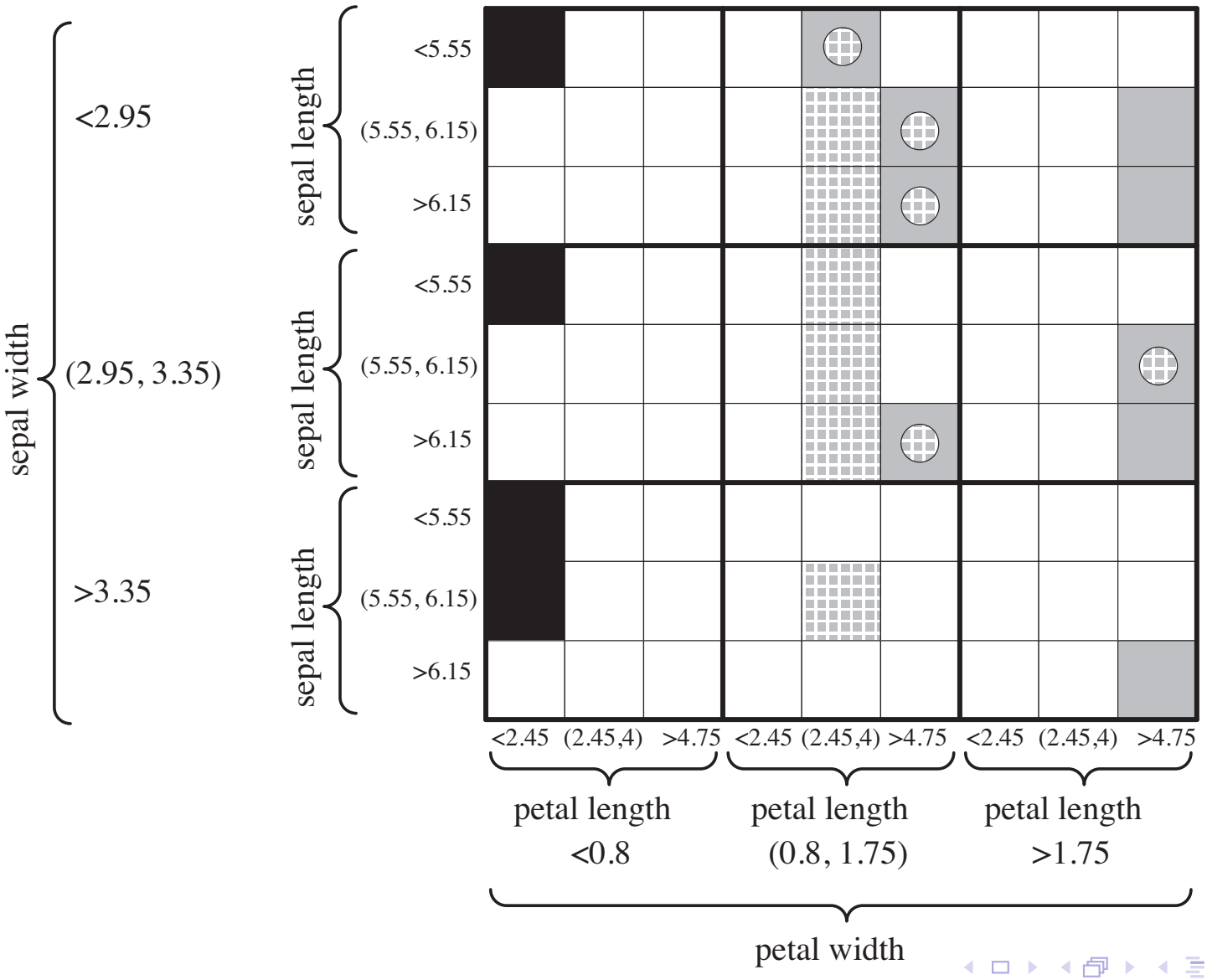
- A predecessor of *dimensional stacking* was the general logic diagrams. Only the Boolean data values 0 and 1 are displayed. M. Ward extends this method later on.
- The dimensional stacking method can be used for exploring clusters and outliers. However, when the dimensionality of data exceeds eight, the display of data and comprehension of the results are difficult.
- The dimensional stacking technique is implemented in the package *Xmdv*.

Scheme of Dimensional Stacking

- The ranges of values of a feature, characterizing the objects, are divided into subranges; a recommendation is that the number of such subranges is not more than five;
- two selected features, called the outer features, are represented by a grid, the numbers of rows and columns are equal to the numbers of subranges;
- when displaying other two features, called the inner features, a new grid is created at each cell of the outer grid; the grids, displaying the inner features, are embedded into all the cells of the outer grid; the recursive embedding continues until all features are displayed;
- the cell of the last embedding is colored, if there are objects the feature values of which are in subranges corresponding to this cell;
- if the classes of objects are known, the color of the cell is selected according to the class of the objects; moreover, the classes are overlapping, colors of the cell may overlap, too.

Iris data visualized by dimensional stacking

- Setosa irises (black cells) are displayed separately from the other two species. The other two species overlap.

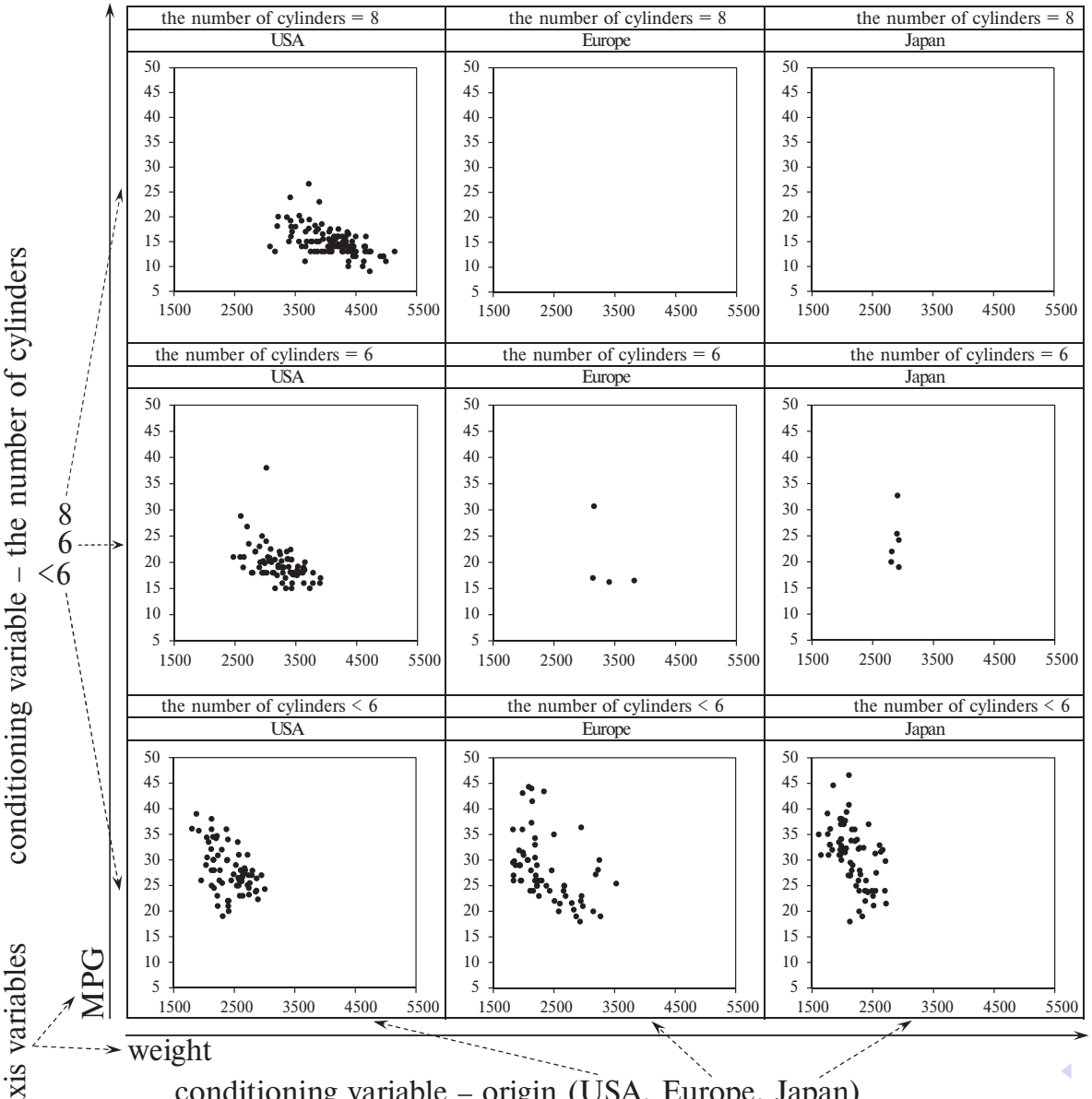


Trellis Display

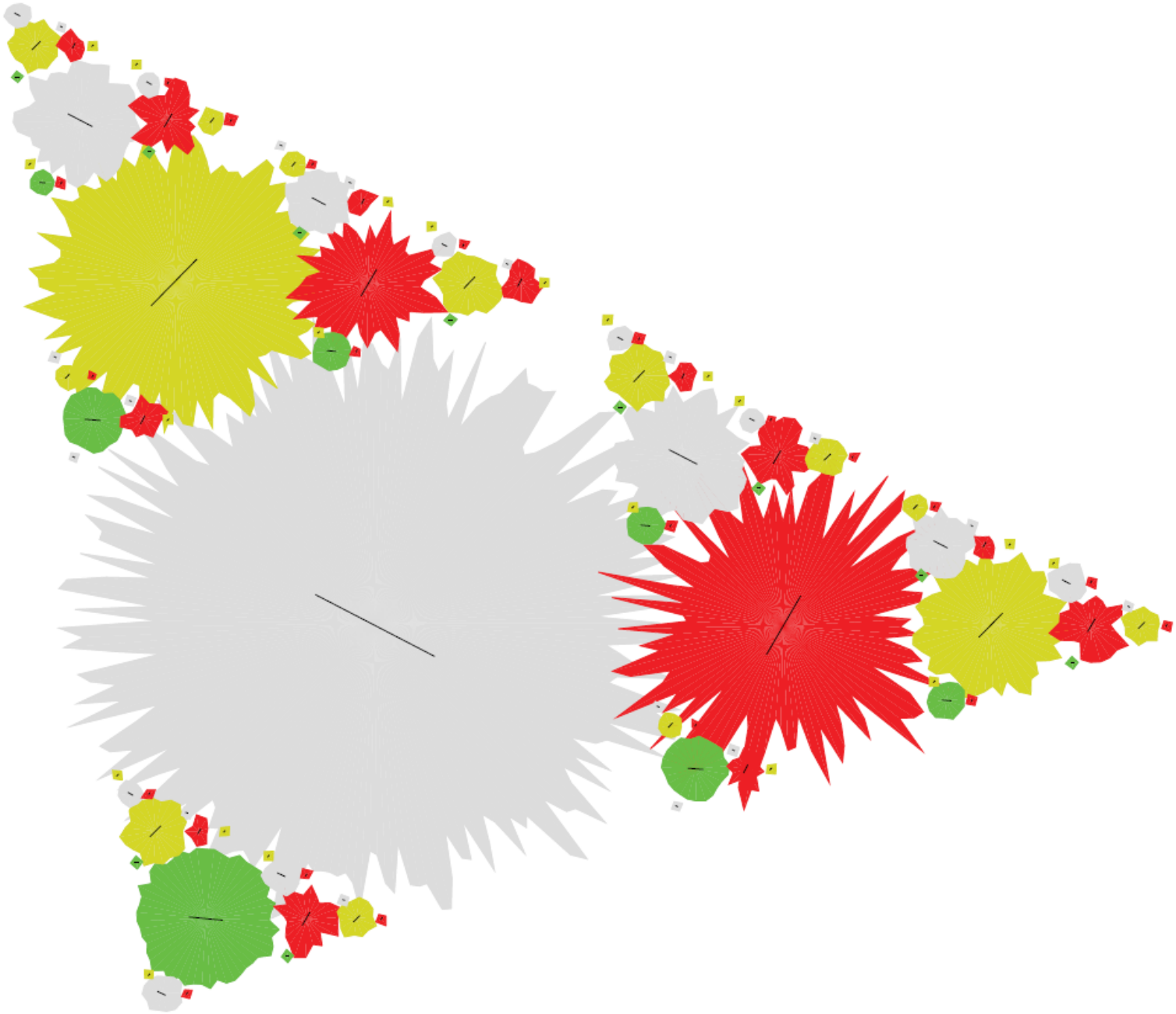
- The *Trellis display* method is similar to the dimensional stacking. The name of the method is derived from the Latin word ‘tri-liceum’ which means a frame of lattice-work used for climbing plants.
- At first two features are selected. They are called *axis variables*. The ranges of the values of these features are divided into non-overlapping subranges.
- Other features are called *conditioning variables*.
- The panel plots are drawn for each pair of subranges. The panel plots can be scatter, bar, surface plots, etc.

Auto MPG Data Visualized by Trellis Display

- The axis variables are weight and miles per gallon (MPG).
- Origin and number of cylinders are conditioning variables.



Fractal Foam for Iris Data



Multidimensional Data Visualization

Linear Projection Methods

Dimensionality Reduction

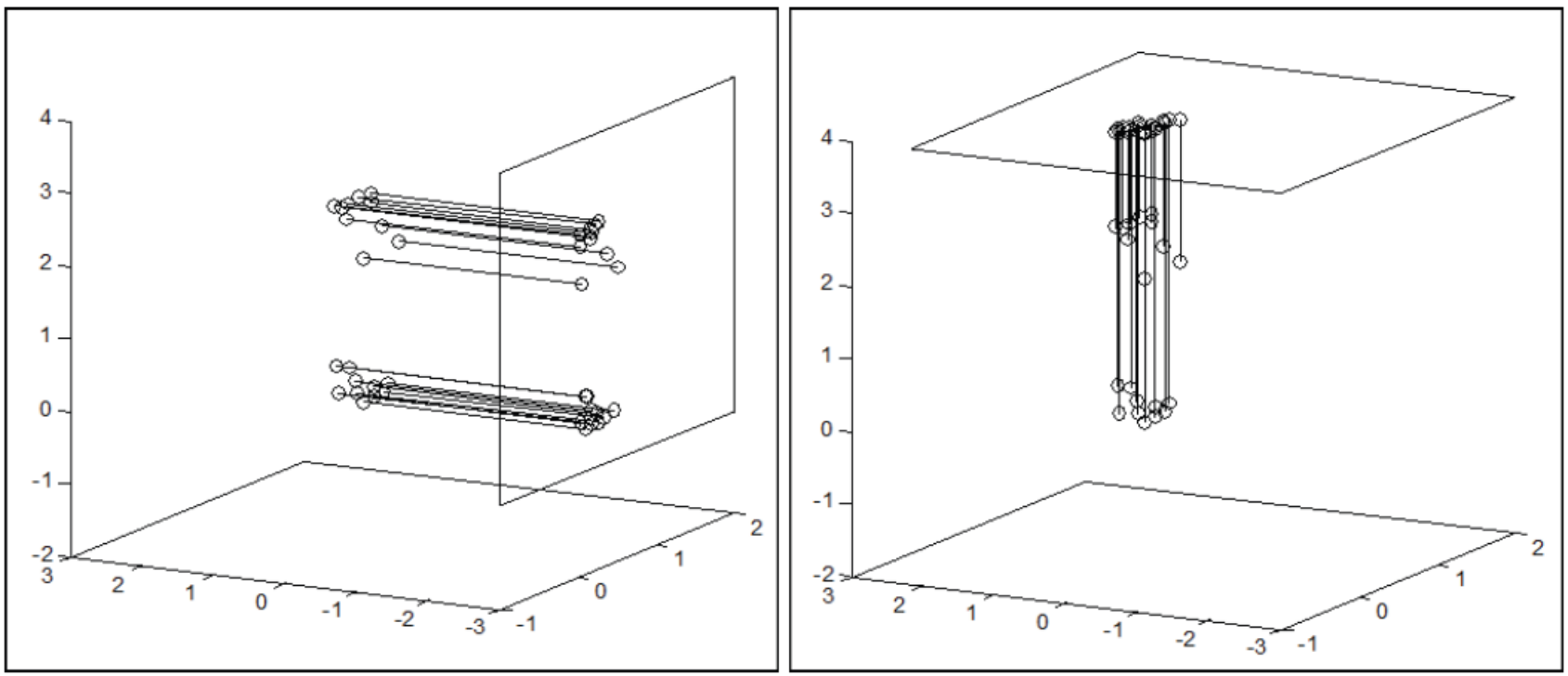
- It is difficult to perceive the data structure using the direct visualization methods, particularly when we deal with large data sets or data of high dimensionality.
- Projection methods are based on reduction of the dimensionality of data.
- Their advantage is that each n-dimensional object is represented as a point in the space of low-dimensionality d, $d < n$, usually $d = 2$.
- There exists a lot of methods that can be used for reducing the dimensionality.
- The aim of these methods is to represent the multidimensional data in a low-dimensional space so that certain properties (such as distances, topology or other proximities) of the data set were preserved as faithfully as possible.
- These methods can be used to visualize the multidimensional data, if a small enough resulting dimensionality is chosen.

Projection Methods

- Methods that allow us to represent multidimensional data from $\mathbb{R}^n$ in a low-dimensional space $\mathbb{R}^d$, $d < n$, are called projection (dimensionality reduction) methods.
- If the dimensionality of the *projection space* is small enough ($d = 2$ or $d = 3$), these methods may be used to visualize the multidimensional data.
- In such a case, the projection space can be called a *display*, *embedding* or *image space*.
- These methods usually invoke formal mathematical criteria by which the projection distortion is minimized.
  1. Linear projection methods:
     a) principal component analysis,
     b) linear discriminant analysis,
     c) projection pursuit.
  2. Nonlinear projection methods:
     a) multidimensional scaling,
     b) locally linear embedding,
     c) isometric feature mapping,
     d) principal curves.

Example of Projection of Three-Dimensional Points

- Figure shows two possible ways of projections of the three-dimensional points ($n = 3$) onto a plane ($d = 2$). We can see two clusters of points on the projection plane on the left and only one cluster on the projection plane on the right.



Example of Projections

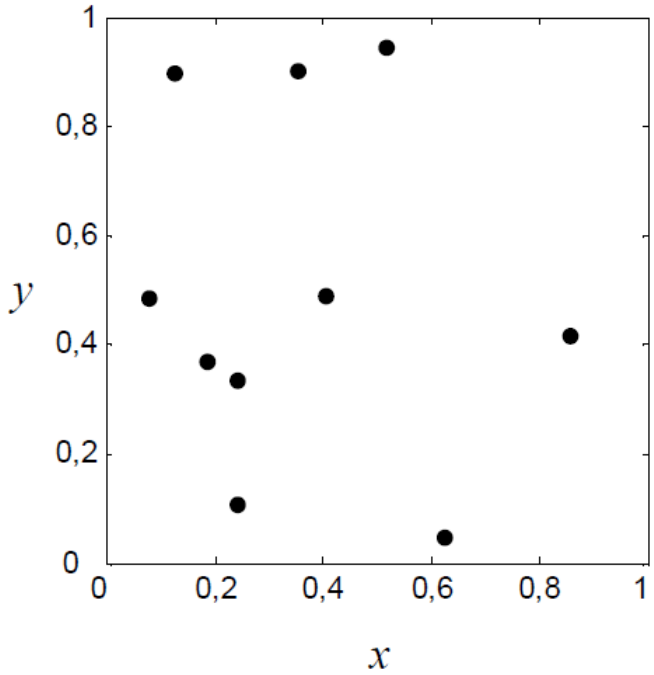
- The example demonstrates that different projections of the same data can reveal different aspects of the data structure (clusters, outliers, etc.).
- Indeed, some projections can fail to reveal any structure.
- Therefore, the proper choose of projection is an important problem.
- When visualizing multidimensional data, we confront with two often contradictory aims.
- On one hand, we want to reduce the dimensionality of data in the simplest way.
- On the other hand, we want to preserve the original information as much as possible.

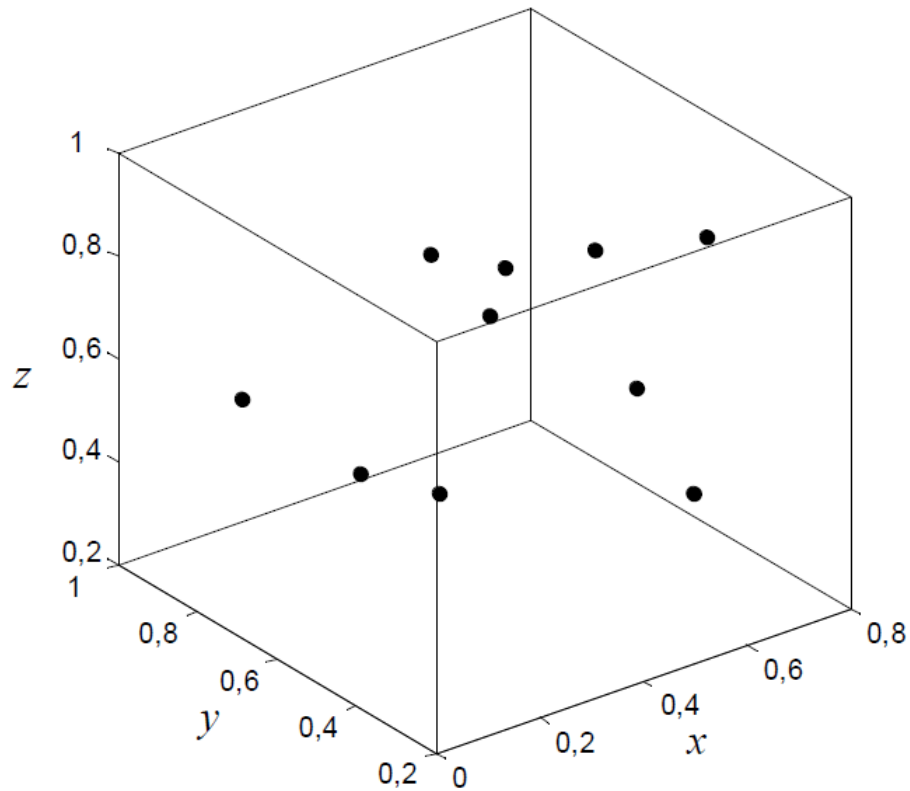
Projection Methods

- The projection methods are used for *transformation* of multidimensional data to a low-dimensional space.
- The aim of these methods is to represent the multidimensional data in a low-dimensional space so that certain properties of the data set were preserved as faithfully as possible.
- These methods can be used to visualize the multidimensional data, if a sufficiently small dimensionality of the projection space $\mathbb{R}^d$ is chosen ($d = 2$ or $d = 3$).
- We call the space $\mathbb{R}^d$ as a display or image space, since its points can be observed visually.

Scatter Plots

- *Scatter plots* are one of the most commonly used techniques for data representation on a plane $\mathbb{R}^2$ or space $\mathbb{R}^3$. Points are displayed in the classic (x, y) or (x, y, z) format.





Projection Methods

- Suppose that the multidimensional data set is defined by a matrix

$$X = \{X_1, X_2, \ldots, X_m\} = \left\{x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\right\}.$$

  Here m is the number of objects (n-dimensional points $X_i \in \mathbb{R}^n$, where $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$). x_{ij} is the jth coordinate, corresponding to the jth feature.
- One needs to find a transformation of the points $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, into points $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, $i = 1, \ldots, m$, that are on a low-dimensional space $\mathbb{R}^d$, $d < n$. One-dimensional space ($d = 1$) can also be used, however more information can be preserved when observing points on a plane ($d = 2$) or a 3D space ($d = 3$).

Criteria of the Projection Quality

- There are some formal mathematical criteria of the projection quality. These criteria are optimized in order to get the optimal projection of multidimensional data onto a low-dimensional space.
- The main goal is to preserve the proportions of distances or estimations of other proximities between the multidimensional points in the image space as well as to preserve, or even to highlight other characteristics of the multidimensional data (for example, clusters).
- Let us remind that the proximity is the general term of *similarity* and *dissimilarity*. A high value of similarity indicates that the objects X_i and X_j are very similar. For dissimilarities, a small value indicates that the objects are very similar, e.g. dissimilarity may be measured using the Euclidean (or other) distances.

Proximity of Data

- A (dis)similarity is a proximity that indicates how two objects X_i and X_j are (dis)similar. The (dis)similarity is denoted by δ_{ij}.
- If δ_{ij} is a similarity, a high δ_{ij} value indicates that the objects X_i and X_j are very similar.
- For dissimilarities, a small δ_{ij} value indicates that the objects are very similar.
- When the proximities are known, the visualization of objects $X_1, X_2, \ldots, X_m$ may be carried out using the matrix of their proximities $\Delta = \{\delta_{ij}, i, j = 1, \ldots, m\}$. The advantage is that the dimensionality n can be unknown. This often happens, for example, in psychological tests.

Proximity Measures

- A proximity matrix can be obtained from matrix X applying some proximity measure, too.
- Often the proximity is measured using the Euclidean distance, which belongs to the group of Minkowski distances. The Minkowski distance between two objects $X_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ and $X_l = (x_{l1}, x_{l2}, \ldots, x_{ln})$ is defined by the formula:

$$d_q(X_k, X_l) = \left\{ \sum_{j=1}^{n} |x_{kj} - x_{lj}|^q \right\}^{\frac{1}{q}}.$$

- Some other proximity measures are also possible: Canberra distance, Bray-Curtis dissimilarity, correlation, etc.

Minkowski Distances

- The following Minkowski distances may be derived for different q:
- City-block or Manhattan distance, $q = 1$:

$$d_1(X_k, X_l) = \sum_{j=1}^{n} |x_{kj} - x_{lj}|.$$

- Euclidean distance, $q = 2$:

$$d_2(X_k, X_l) = \sqrt{\sum_{j=1}^{n} |x_{kj} - x_{lj}|^2}.$$

- Chebyshev distance, $q = \infty$:

$$d_\infty(X_k, X_l) = \max_j \left| x_{kl} - x_{lj} \right|.$$

Linear Transformation

- There are linear and nonlinear projection methods.
- Linear projection methods pursue a linear transformation of data. There are various linear transformations: rotation, shearing, reflection, scaling, etc.
- A *linear transformation* may be described by linear equations
$$Y_i = X_i A + B.$$
- If $d = n$, i.e. $Y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$ and $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, then A is a square matrix, consisting of n rows and n columns. The matrix A is called a *transformation matrix*.
- If a linear transformation is used for dimensionality reduction, then $d < n$, $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, $i = 1, \ldots, m$, and A is a matrix, consisting of n rows and d columns.

Example of Linear Transformation

- Let us analyze a simple case of a linear transformation, when $n = d = 2$. Let us have a point $X_i = (x_{i1}, x_{i2})$. Transform it linearly to a point $Y_i = (y_{i1}, y_{i2})$ using a matrix

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}.$$

- In the case of rotation, the elements of the matrix A can be expressed using trigonometric functions:

$$A = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix},$$

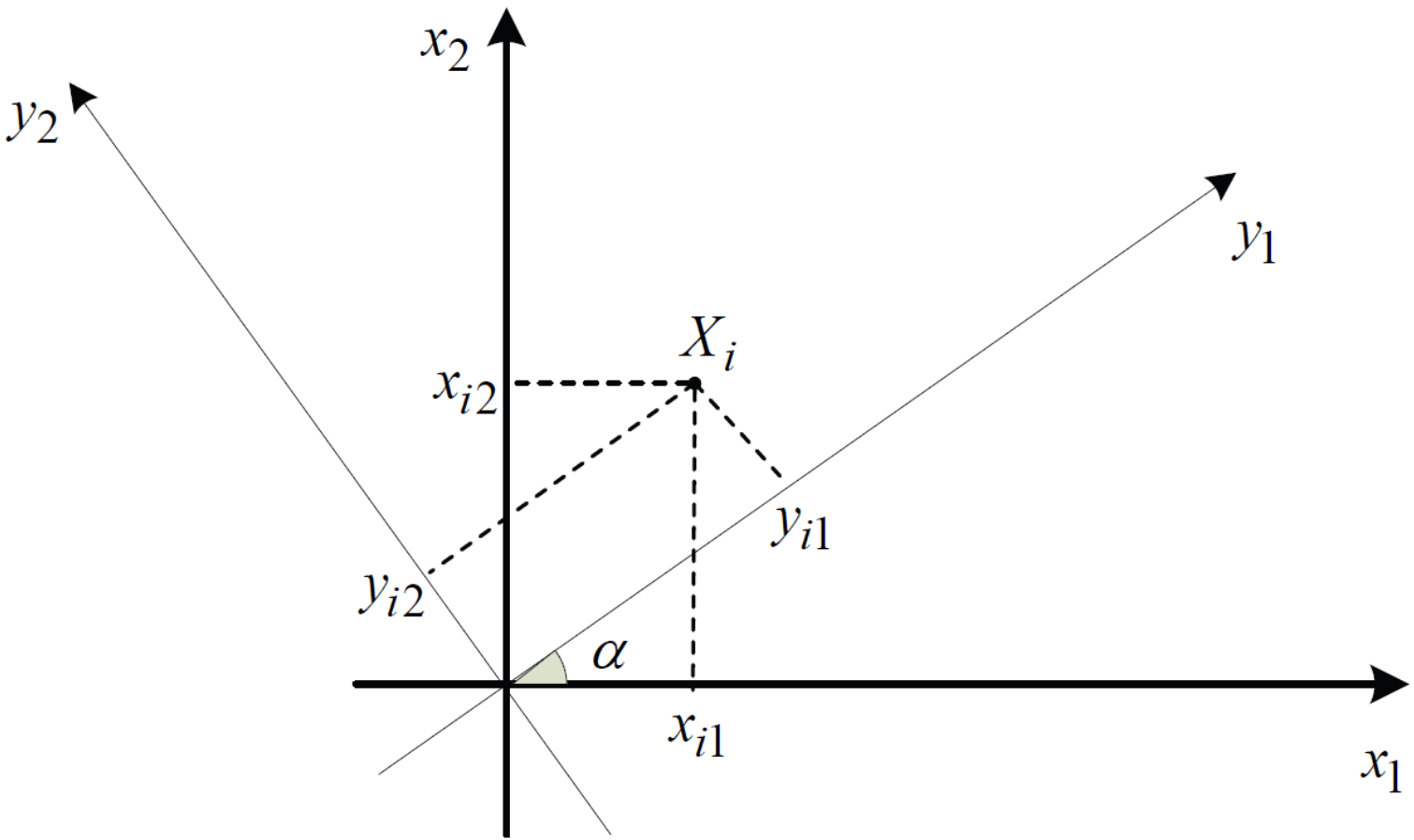
  where α is the rotation angle between the axes x_1 and y_1, as well as between the axes x_2 and y_2.
- The coordinate system (x_1, x_2) is rotated around the origin $(0, 0)$ counterclockwise by α to get a coordinate system (y_1, y_2).
- Such a matrix A is called a *rotation matrix*.

Example of Linear Transformation

- The coordinates of the point Y_i may be expressed as:

$$\begin{aligned} y_{i1} &= x_{i1}\cos(\alpha) + x_{i2}\sin(\alpha), \\ y_{i2} &= x_{i2}\cos(\alpha) - x_{i1}\sin(\alpha). \end{aligned}$$

- Actually (y_{i1}, y_{i2}) is a linear transformation of the point X_i to the coordinate system (y_1, y_2).



Nonlinear Transformation

- A nonlinear transformation is such that cannot be expressed in the linear form.
- The nonlinear transformation may be described as follows:

$$Y = f(X),$$

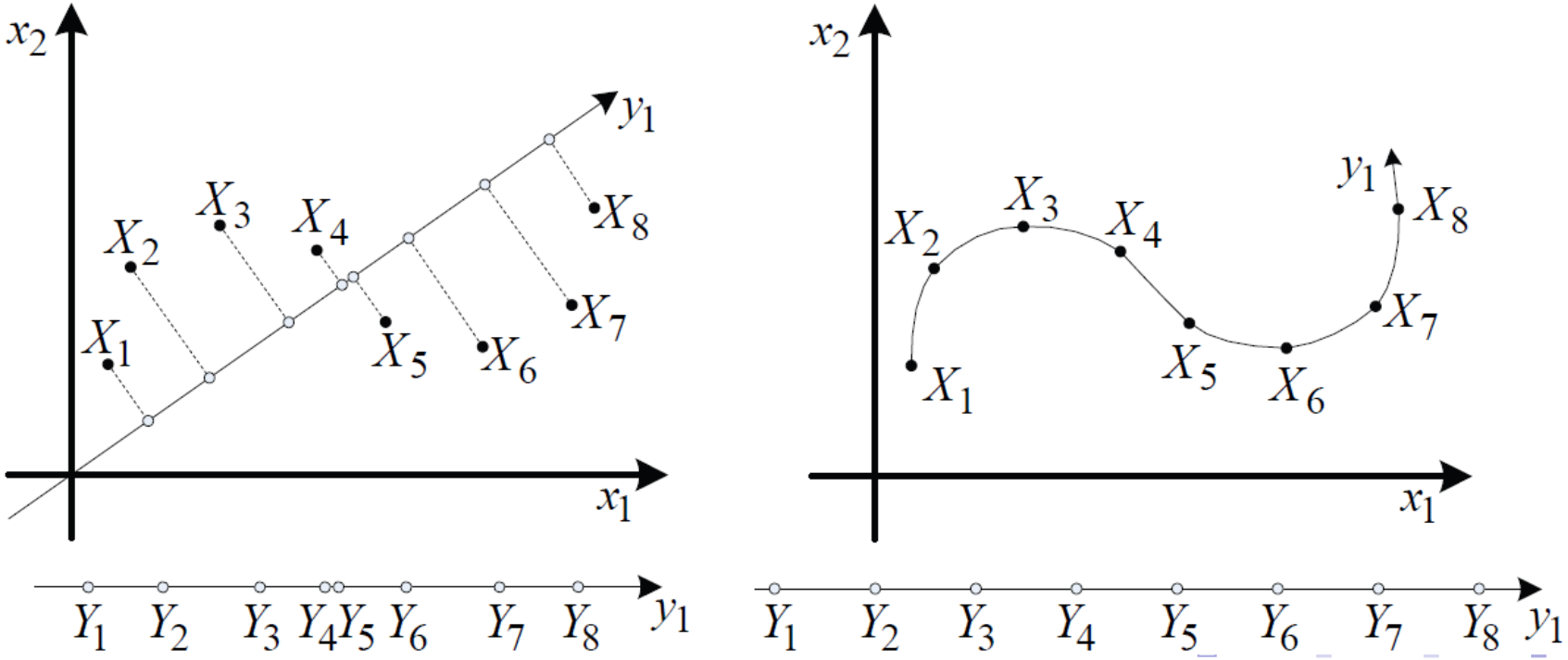
where f is a nonlinear function and

$$Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\}.$$

- The nonlinear transformation is more complicated than the linear one and requires more time-consuming computations. However, such a transformation allows us to preserve the characteristics of multidimensional data better as compared with the linear transformation if $d < n$, i.e. the data are projected to a lower-dimensional space.

Linear and Nonlinear Projection

- Let the two-dimensional points $X_1, X_2, \ldots, X_8$ be spread so that the distances between the nearest points are equal.
- If we project to the one-dimensional space using the linear projection (to the line y_1), equal distances between the nearest points are not preserved. However, in the case of the nonlinear projection, when the proper transformation is found, the distances between the nearest points remain equal.



Principal Component Analysis

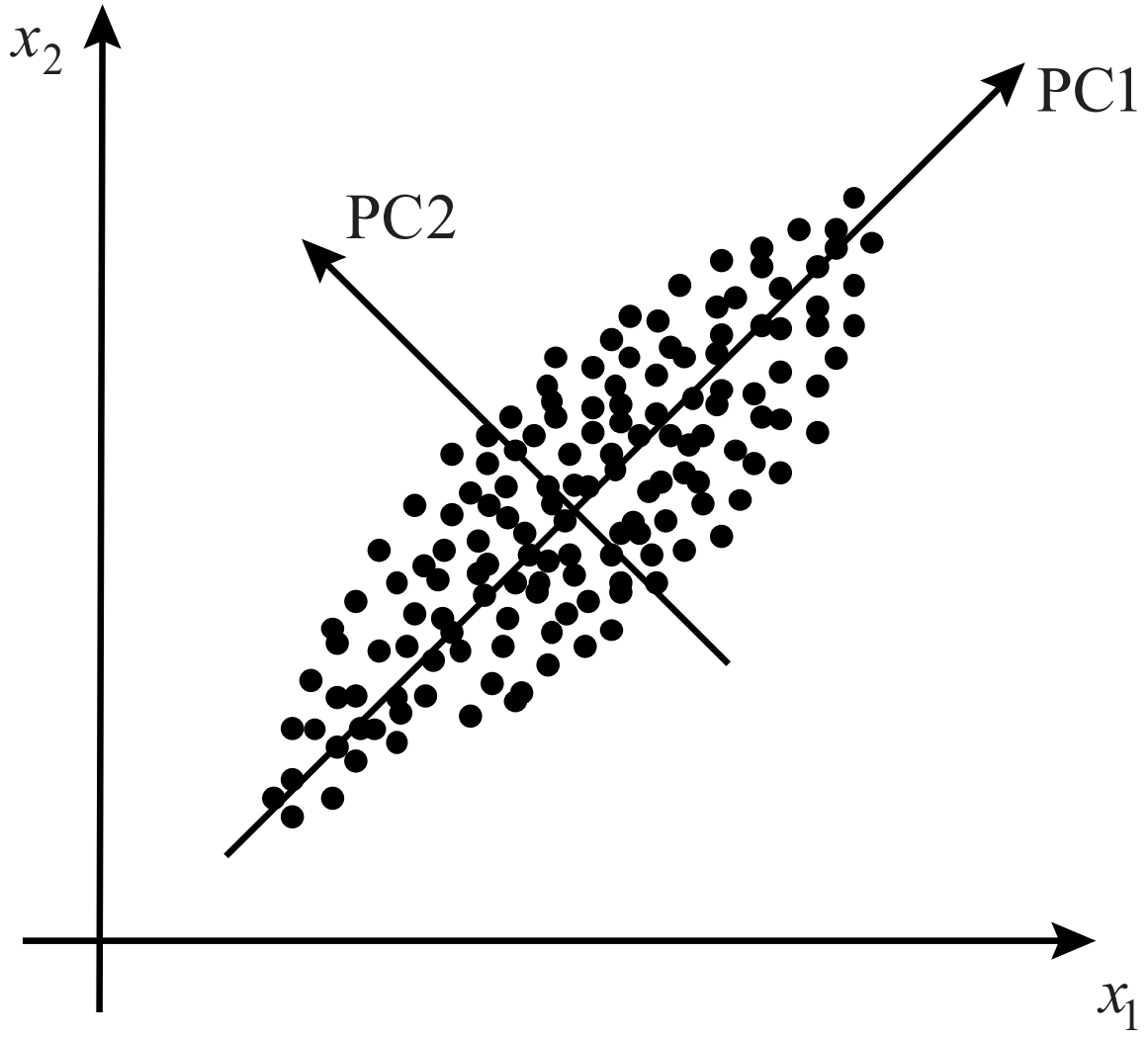
- The principal component analysis (PCA) is a well-known data analysis technique invented in 1901 by Pearson.
- It is a way of linear transforming a set X of n-dimensional points $X_1, X_2, \ldots, X_m$ into another set Y of n-dimensional points $Y_1, Y_2, \ldots, Y_m$.
- The property of the set is that the largest part of its information content is stored in the first few coordinates (components) of points Y_i, $i = 1, \ldots, m$.
- The principal component analysis is often used to reduce the dimensionality of multidimensional points X_i, $i = 1, \ldots, m$, by discarding some of the components of the points Y_i and by leaving only the first (principal) d ones.
- The principal component analysis projects the data linearly into a low-dimensional space preserving the variance of the data best.

Principal Component Analysis

- The main idea of PCA is to reduce the dimensionality of data by performing a linear transformation and rejecting a part of the components, variances of which are the smallest ones.
- When analyzing the data set X, a direction in $\mathbb{R}^n$ with the maximal variance is found.
- This direction defines the first principal component.
- Other principal components maximize the variance of a data set in the directions orthogonal to the previous principal components.
- So, the principal components are uncorrelated and ordered by decreasing variances.

Illustration of Principal Component Analysis

- Figure illustrates a two-dimensional case with two principal components PC1 and PC2.



Principal Component Analysis

- The principal component analysis needs a correlation or covariance matrix of features.
- Suppose we have a data matrix X:

$$X = \{X_1, X_2, \ldots, X_m\} = \left\{x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\right\}.$$

- The rows of this matrix correspond to the objects $X = \{X_1, X_2, \ldots, X_m\}$ and the columns correspond to the features $x_1, x_2, \ldots, x_n$ characterizing the objects.

Correlation

- A *correlation* is a number that describes the degree of relationship between two features.
- The *correlation coefficient* r_{kl} between the features x_k and x_l is computed by the formula:

$$r_{kl} = \frac{\sum_{i=1}^{m}(x_{ik} - \bar{x}_k)(x_{il} - \bar{x}_l)}{\sqrt{\sum_{i=1}^{m}(x_{ik} - \bar{x}_k)^2 \sum_{i=1}^{m}(x_{il} - \bar{x}_l)^2}},$$

where

$$\bar{x}_k = \frac{1}{m}\sum_{i=1}^{m} x_{ik} \text{ and } \bar{x}_l = \frac{1}{m}\sum_{i=1}^{m} x_{il}.$$

- The *correlation matrix* $R = \{r_{kl},\ k, l = 1, \ldots, n\}$ consists of the correlation coefficients. The diagonal elements $r_{kk},\ k = 1, \ldots, n$, are equal to 1. This matrix is symmetric.

Covariance

- The *covariance coefficient* c_{kl} between the features x_k and x_l is computed by the formula:

$$c_{kl} = \frac{1}{m-1} \sum_{i=1}^{m} (x_{ik} - \bar{x}_k)(x_{il} - \bar{x}_l).$$

- If $k = l$, the expression is a variance formula, i.e. c_{kk} is the variance of feature x_k. The *covariance matrix* C consists of the covariance coefficients:

$$C = \{c_{kl},\ k, l = 1, \ldots, n\}.$$

- It follows that the correlation coefficient is equal to:

$$r_{kl} = \frac{c_{kl}}{\sqrt{c_{kk} c_{ll}}}.$$

- If the features x_k and x_l are not correlated, their covariance coefficient is equal to zero: $c_{kl} = c_{lk} = 0, k \neq l$.

Principal Component Matrix

- Let us describe the eigenvector and the eigenvalue of the covariance matrix.
- The *eigenvalue* λ_k and the *eigenvector* E_k corresponding to λ_k are solutions of the equation $CE_k = \lambda_k E_k$. Here E_k is a vector-column. The value of λ_k is found from the characteristic equation $|C - \lambda_k I| = 0$, where I is an identity matrix of the same order as the matrix C and $|.|$ denotes a determinant of the matrix.
- The number of eigenvectors is equal to n.
- Let us sort the eigenvectors E_k, $k = 1, \ldots, n$, in descending order of the corresponding eigenvalues $(\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n)$.
- The matrix $A = (E_1, E_2, \ldots, E_n)$ is called a principal component matrix. The columns of this matrix are the eigenvectors E_k, $k = 1, \ldots, n$, corresponding to the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n$.
- Each column of the matrix A is orthogonal to any other column. Usually E_k, $k = 1, \ldots, n$ of unit length are used.

Principal Component Transformation

- Let us transform the points X_i, $i = 1, \ldots, m$, to points Y_i, $i = 1, \ldots, m$, by the formula:

$$Y_i = (X_i - \bar{X})A, \ i = 1, \ldots, m,$$

where $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $\bar{X} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$, $A = (E_1, E_2, \ldots, E_n)$, A is a transformation matrix.

- $Y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$, obtained by the formula, are points in the new coordinate system $(y_1, y_2, \ldots, y_n)$.
- The eigenvectors E_k, $k = 1, \ldots, n$ represent the basis set of this system.
- The covariance matrix of components $y_1, y_2, \ldots, y_n$ of the points Y_i, $i = 1, \ldots, m$ is equal to:

$$\begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}.$$

Principal Component Transformation with Dimensionality Reduction

- We may use only a few first eigenvectors for transforming multidimensional data instead of all the eigenvectors of the covariance matrix.
- Suppose that the matrix A_d consists of the first d eigenvectors. Then it is possible to define a transformation

$$Y_i = (X_i - \bar{X})A_d, \ i = 1, \ldots, m.$$

- In this way, a projection of the point X_i to the d-dimensional space is derived.

Properties of Eigenvalues

- Some properties of the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ are as follows:
  1. $\sum_{k=1}^{n} \lambda_k = \sum_{k=1}^{n} c_{kk}$.
  2. $\lambda_1 \geq \max_k c_{kk}$.
  3. $\lambda_n \leq \min_k c_{kk}$.
- It follows from the second property that the first eigenvector E_1 describes the first principal component y_1, the variance of which is the highest one among $\lambda_1, \lambda_2, \ldots, \lambda_n$. The second eigenvector E_2 describes the second principal component y_2, the variance of which is the second one according to the value.

Principal Components in Data Visualization

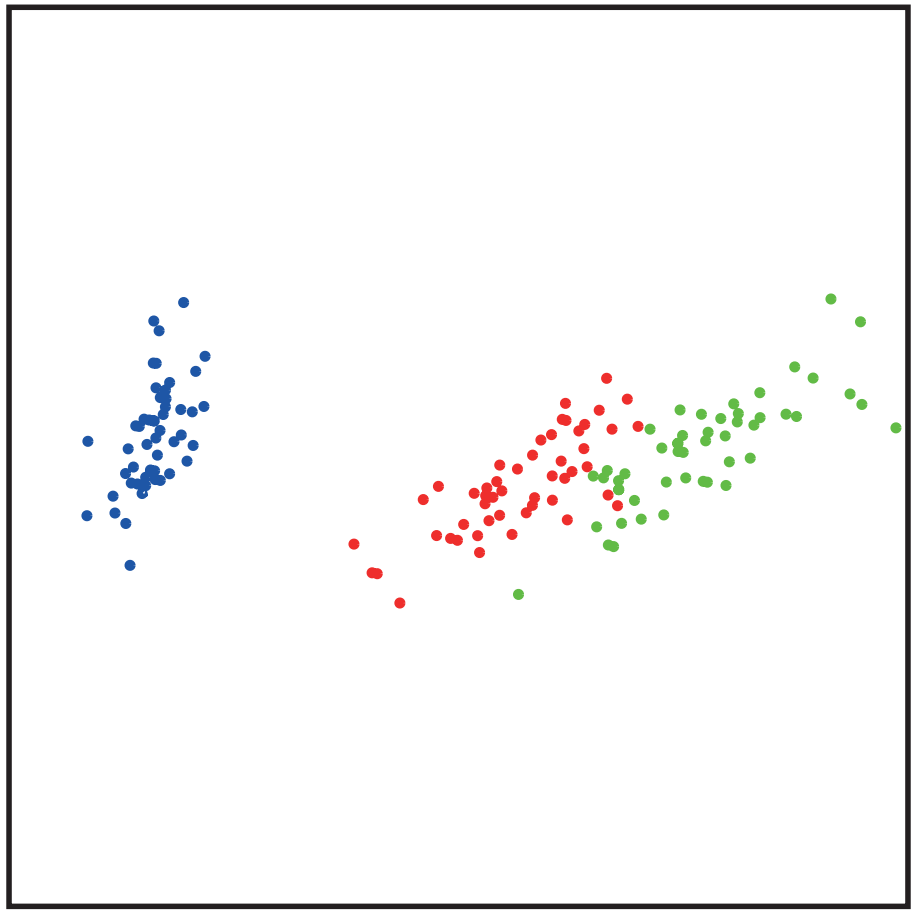
- It follows from the third property that the last eigenvector E_n describes the principal component y_n, the variance of which is smallest. Therefore, if only the first d eigenvectors are used, the components with the smallest variances will be rejected.
- In order to derive the principal components, it suffices to find the highest d eigenvalues and the corresponding eigenvectors of matrix C. This matrix has specific properties: it is symmetric and non-negative definite, therefore, special fast algorithms are used.
- The advantage of the principal component analysis is the simplicity of its idea. This fact influences its popularity and wide application.

Application of Principal Component Analysis in Visualization of Multidimensional Data

- The examples of application of principal component analysis in visualization of multidimensional data are presented.
- The Iris and the Breast Cancer data sets are visualized by two principal components.
- We do not present labels and units for both axes in the figure, because we are interested in observing the interlocation of points on a plane only.
- In figures we can observe clusters of points, corresponding to particular classes of n-dimensional objects.

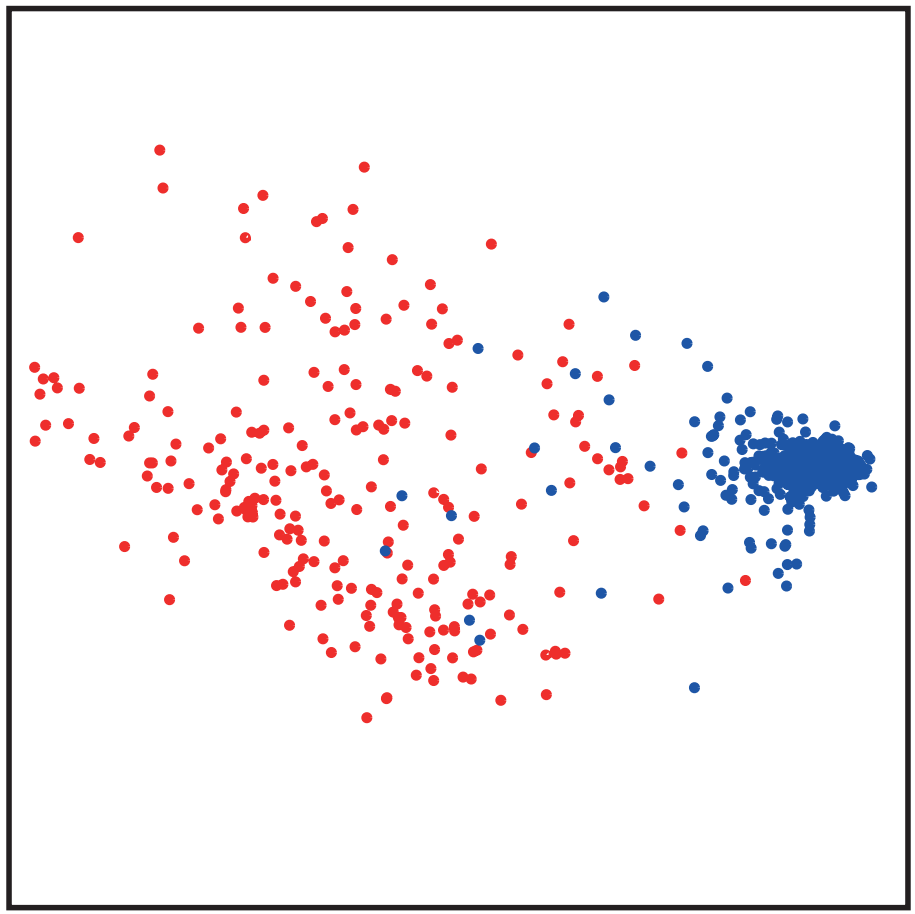
Iris Data Set Visualized Using PCA

- Setosa irises (marked in blue) are faraway from Versicolor (red) and Virginica (green) irises. There is no exactly expressed boundary between these two species.



Breast Cancer Data Set Visualized Using PCA

- A large amount of the points, corresponding to the benign tumor data (blue points), are concentrated in one area, and the other points, corresponding to the malignant tumor data (red), are spread widely.



Principal Component Analysis

- In the literature, some authors prefer to define the principal components using the correlation matrix instead of the covariance one. The correlation between a pair of features is equivalent to the covariance divided by the product of the standard deviations of two features.
- Although PCA is widely used for multidimensional data visualization, it has some shortcomings. It is not good for data of nonlinear structures, consisting of arbitrarily shaped clusters or curved manifolds.
- During the past 50 years many works have appeared proposing extensions of the principal components to data with a nonlinear structure. *Principal curves* are a nonlinear generalization of principal components. The principal curve provides a nonlinear summary of the data. The idea of the principal curves can be extended to principal surfaces.

Linear Discriminant Analysis

- In contrast to most other dimensionality reduction methods, a *linear discriminant analysis* (LDA) is a supervised method.
- The method is often called Fisher's discriminant analysis.
- In a supervised strategy, some known properties of data (for example, belonging of the objects to one of classes) are applied.
- LDA transforms multidimensional data to a low-dimensional space, maximizing the linear separability between objects belonging to different classes.

Linear Discriminant Analysis

- Suppose that the data matrix X

$$X = \{X_1, X_2, \ldots, X_m\} = \left\{x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\right\}.$$

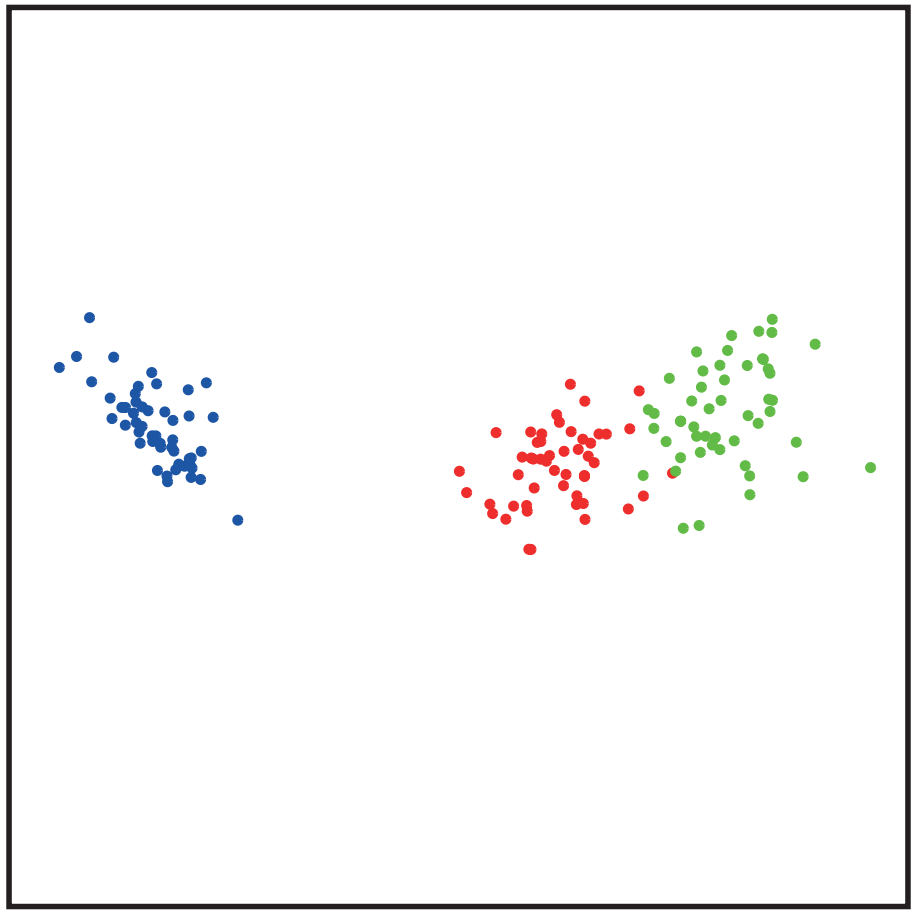
  consists of k submatrices $X^{(1)}, X^{(2)}, \ldots, X^{(k)}$, where k is the number of classes.
- The rows of $X_i^{(j)}$, $i = 1, \ldots, m_j$, of $X^{(j)}$ correspond to objects that belong to the jth classes. Here m_j is the number of objects in the jth class. The number of all objects $m = \sum_{j=1}^{k} m_j$.

Scheme of Linear Discriminant Analysis

1. Covariance matrix C of the whole data set X and covariance matrices $C^{(j)}$, $j = 1, \ldots, k$, of each class are computed. The within-class scatter is defined: $S_w = \sum_{j=1}^{k} p_j C^{(j)}$, where $p_j = \frac{m_j}{m}$. The between-class scatter is defined: $S_b = C - S_w$.
2. The eigenvectors and eigenvalues of the matrix $S = S_w^{-1} S_b$ are computed. The eigenvectors are sorted in descending order of the corresponding eigenvalues. d eigenvectors, corresponding to the highest eigenvalues, are selected (under the requirement that $d < k$).
3. The transformation $Y_i = (X_i - \bar{X})A_d$, $i = 1, \ldots, m$, is performed, where A_d is a d-column matrix consisting of the eigenvectors, corresponding to the highest d eigenvalues of matrix S. Here $\bar{X} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ is the vector of averages of the features.

Iris Data Set Visualized Using LDA

- The difference between LDA and PCA is that, in addition, the known classes of objects are applied.



Projection Pursuit

- A projection pursuit is a type of statistical technique which involves finding the most “interesting” projection in multidimensional data.
- The aim of the projection pursuit, as many other projection methods, is to find such linear combinations of components (commonly two or three dimensional) that the transformed data preserve a structure of the initial data.
- Suppose, we have the data matrix X, rows of which are the n-dimensional vectors. A projection of points of the space $\mathbb{R}^n$ to the space $\mathbb{R}^d$ can be expressed as follows: $Y = XA$, where
  $X = \{X_1, X_2, \ldots, X_m\} = \{x_{ij}, i = 1, \ldots, m, j = 1, \ldots, n\}$, A is a matrix consisted of n rows and d columns,
  $Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij}, i = 1, \ldots, m, j = 1, \ldots, d\}$ is a matrix of the data obtained after projection. A question arises how to choose the matrix A.

Projection Pursuit

- At first, we have to decide what kind of feature (property) we wish to detect (highlight) in a visualization process. After that, one needs to define a measure that reflects this property. Call this measure $I(Y)$. It is sometimes called the index function.
- Suppose we wish to highlight data clusters. In such a case, a statistical clustering measure – mean nearest neighbor distance could be used. Lower values of this measure indicate greater clustering.
- Let's $I(Y)$ is the mean nearest neighbor distance. The expression $I(Y)$ can be written in the form $I(XA)$. The problem of projection choice can be formulated as an optimization problem: it is necessary to choose such a matrix A that the value of the function $I(XA)$ was minimal.
- If we are interested in a presence of outliers, then the problem is changed so that the maximal mean nearest neighbor distance were derived. More complex index functions could be used too.

Multidimensional Data Visualization

Nonlinear Projection Methods

Dimensionality Reduction

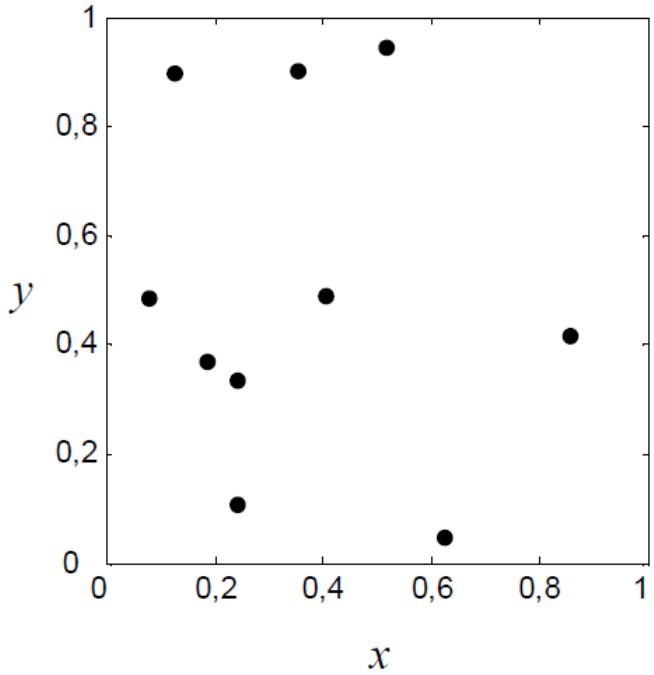
- It is difficult to perceive the data structure using the direct visualization methods, particularly when we deal with large data sets or data of high dimensionality.
- Projection methods are based on reduction of the dimensionality of data.
- Their advantage is that each n-dimensional object is represented as a point in the space of low-dimensionality d, $d < n$, usually $d = 2$.
- There exists a lot of methods that can be used for reducing the dimensionality.
- The aim of these methods is to represent the multidimensional data in a low-dimensional space so that certain properties (such as distances, topology or other proximities) of the data set were preserved as faithfully as possible.
- These methods can be used to visualize the multidimensional data, if a small enough resulting dimensionality is chosen.

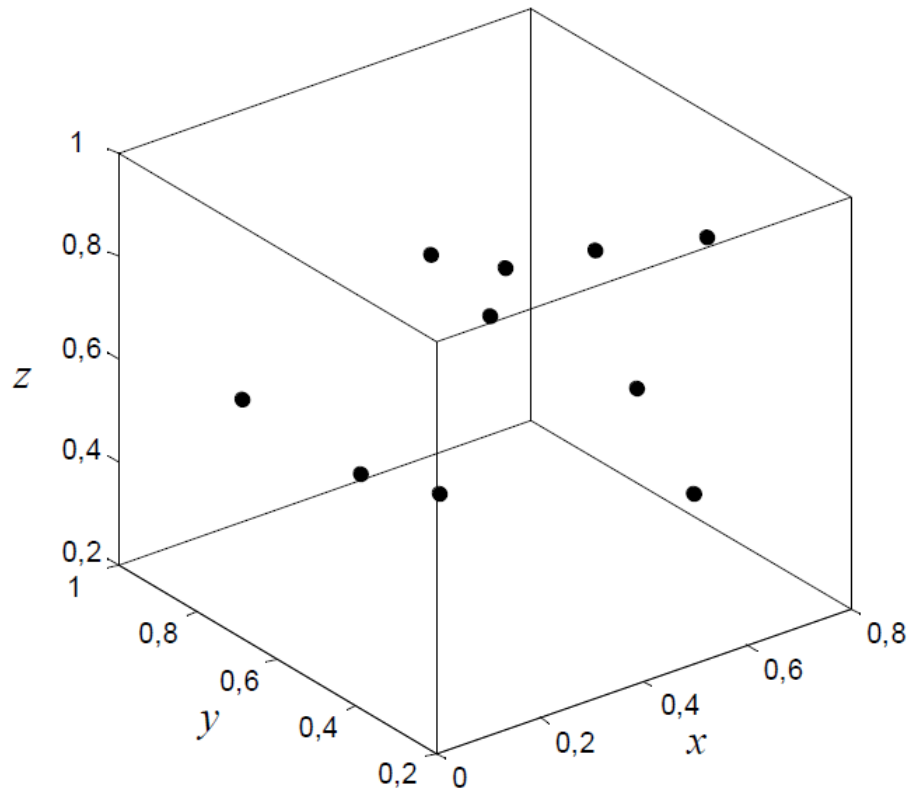
Projection Methods

- The projection methods are used for *transformation* of multidimensional data to a low-dimensional space.
- The aim of these methods is to represent the multidimensional data in a low-dimensional space so that certain properties of the data set were preserved as faithfully as possible.
- These methods can be used to visualize the multidimensional data, if a sufficiently small dimensionality of the projection space $\mathbb{R}^d$ is chosen ($d = 2$ or $d = 3$).
- We call the space $\mathbb{R}^d$ as a display or image space, since its points can be observed visually.
- These methods usually invoke formal mathematical criteria by which the projection distortion is minimized.

Scatter Plots

- *Scatter plots* are one of the most commonly used techniques for data representation on a plane $\mathbb{R}^2$ or space $\mathbb{R}^3$. Points are displayed in the classic (x, y) or (x, y, z) format.





Projection Methods

- Suppose that the multidimensional data set is defined by a matrix

$$X = \{X_1, X_2, \ldots, X_m\} = \left\{x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\right\}.$$

  Here m is the number of objects (n-dimensional points $X_i \in \mathbb{R}^n$, where $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$). x_{ij} is the jth coordinate, corresponding to the jth feature.
- One needs to find a transformation of the points $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, into points $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, $i = 1, \ldots, m$, that are on a low-dimensional space $\mathbb{R}^d$, $d < n$. One-dimensional space ($d = 1$) can also be used, however more information can be preserved when observing points on a plane ($d = 2$) or a 3D space ($d = 3$).

Criteria of the Projection Quality

- There are some formal mathematical criteria of the projection quality. These criteria are optimized in order to get the optimal projection of multidimensional data onto a low-dimensional space.
- The main goal is to preserve the proportions of distances or estimations of other proximities between the multidimensional points in the image space as well as to preserve, or even to highlight other characteristics of the multidimensional data (for example, clusters).

Nonlinear Transformation

- There are linear and nonlinear projection methods.
- Linear projection methods pursue a linear transformation of data. A *linear transformation* may be described by linear equations

$$Y_i = X_i A.$$

- A nonlinear transformation may be described as follows:

$$Y = f(X),$$

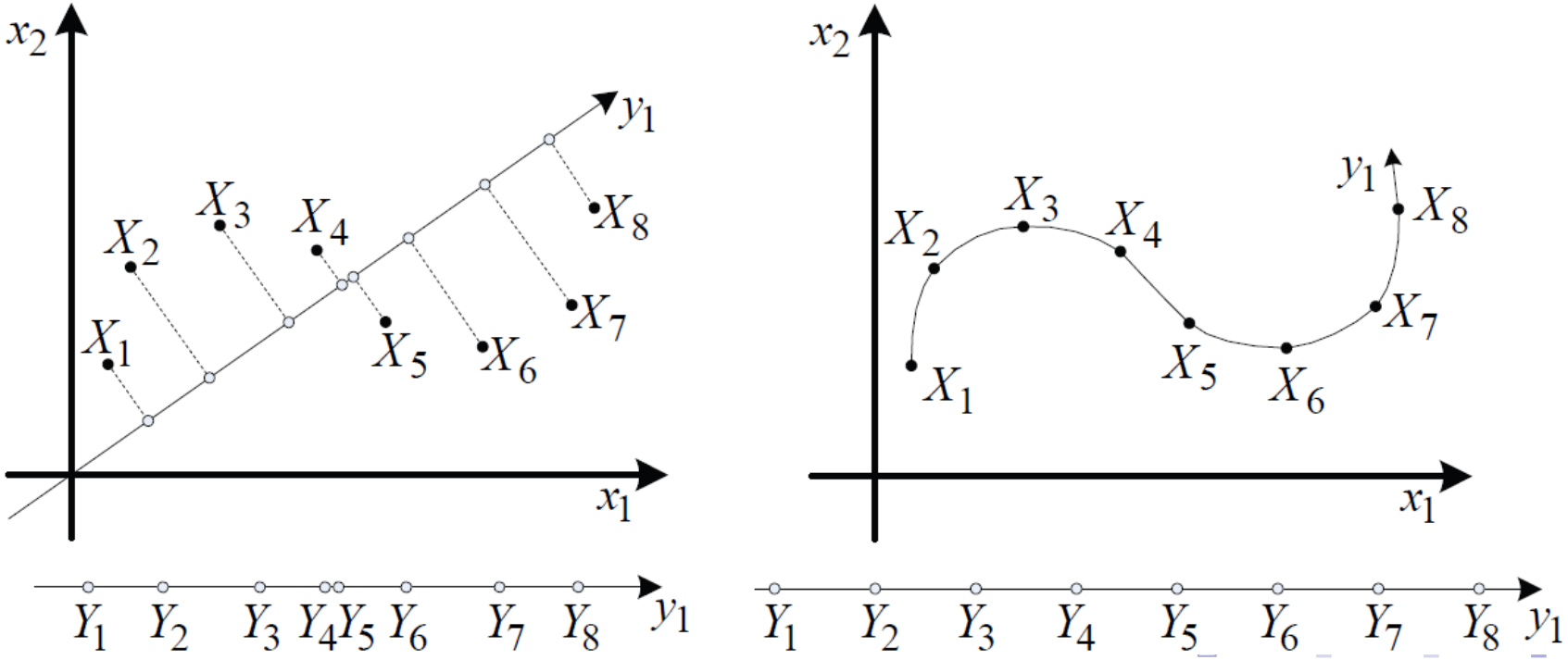
where f is a nonlinear function and

$$Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\}.$$

- The nonlinear transformation is more complicated than the linear one and requires more time-consuming computations. However, such a transformation allows us to preserve the characteristics of multidimensional data better as compared with the linear transformation if $d < n$, i.e. the data are projected to a lower-dimensional space.

Linear and Nonlinear Projection

- Let the two-dimensional points $X_1, X_2, \ldots, X_8$ be spread so that the distances between the nearest points are equal.
- If we project to the one-dimensional space using the linear projection (to the line y_1), equal distances between the nearest points are not preserved. However, in the case of the nonlinear projection, when the proper transformation is found, the distances between the nearest points remain equal.



Nonlinear Projection Methods

- multidimensional scaling,
- locally linear embedding,
- isometric feature mapping,
- principal curves.

Proximity Measures

- The aim of projection methods is to transform multidimensional data to a low-dimensional space so that the proximity of the data was possibly preserved. Therefore, *proximity measures* should be defined.
- Often the proximity is measured using the Euclidean distance, which belongs to the group of Minkowski distances. The Minkowski distance between two objects $X_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ and $X_l = (x_{l1}, x_{l2}, \ldots, x_{ln})$ is defined by the formula:

$$d_q(X_k, X_l) = \left\{ \sum_{j=1}^{n} |x_{kj} - x_{lj}|^q \right\}^{\frac{1}{q}}.$$

- Some other proximity measures are also possible: Canberra distance, Bray-Curtis dissimilarity, correlation, etc.

Minkowski Distances

- City-block or Manhattan distance, $q = 1$:

$$d_1(X_k, X_l) = \sum_{j=1}^{n} |x_{kj} - x_{lj}|.$$

- Euclidean distance, $q = 2$:

$$d_2(X_k, X_l) = \sqrt{\sum_{j=1}^{n} |x_{kj} - x_{lj}|^2}.$$

- Chebyshev distance, $q = \infty$:

$$d_\infty(X_k, X_l) = \max_j \left| x_{kl} - x_{lj} \right|.$$

Distances between two objects X_k and X_l satisfy

- $d(X_k, X_l)$ is a nonnegative real number;
- $d(X_k, X_k) = 0$;
- $d(X_k, X_l) = d(X_l, X_k)$, i.e. the distance from object X_k to object X_l is equal to the distance from object X_l to object X_k;
- $d(X_k, X_l) \leq d(X_k, X_j) + d(X_j, X_l)$, i.e. the distance between any two objects X_k and X_l cannot be larger than a sum of distances between objects X_k, X_j and X_l, X_j (triangle inequality).

Multidimensional scaling

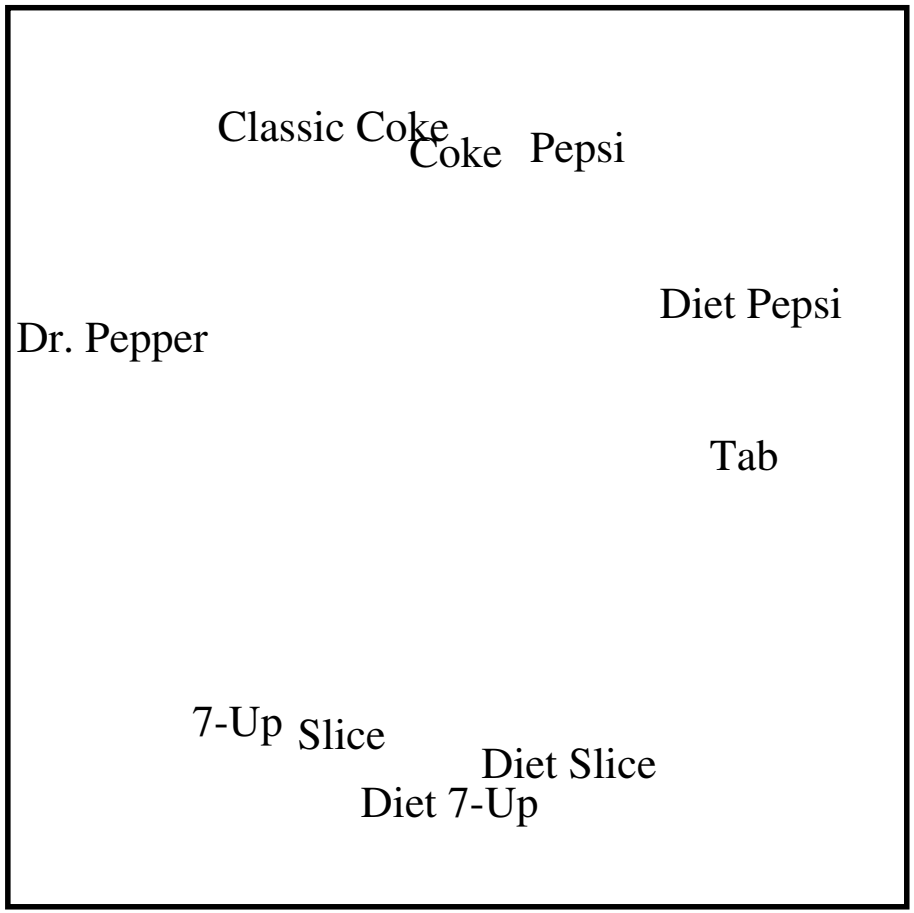
- *Multidimensional scaling* (MDS) refers to a group of methods that are widely used for dimensionality reduction and visualization of multidimensional data.
- The data for MDS is a matrix consisting of pairwise proximities of the objects.
- Let us denote the pairwise proximity of the ith and jth objects by δ_{ij}. If the objects are defined by the multidimensional points $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, m$, the proximity can be measured by the distance between points: $\delta_{ij} = d(X_i, X_j)$, where various distances can be used. The distance between points in a low-dimensional space Y_i and Y_j corresponding to the ith and jth objects is denoted by $d(Y_i, Y_j)$.

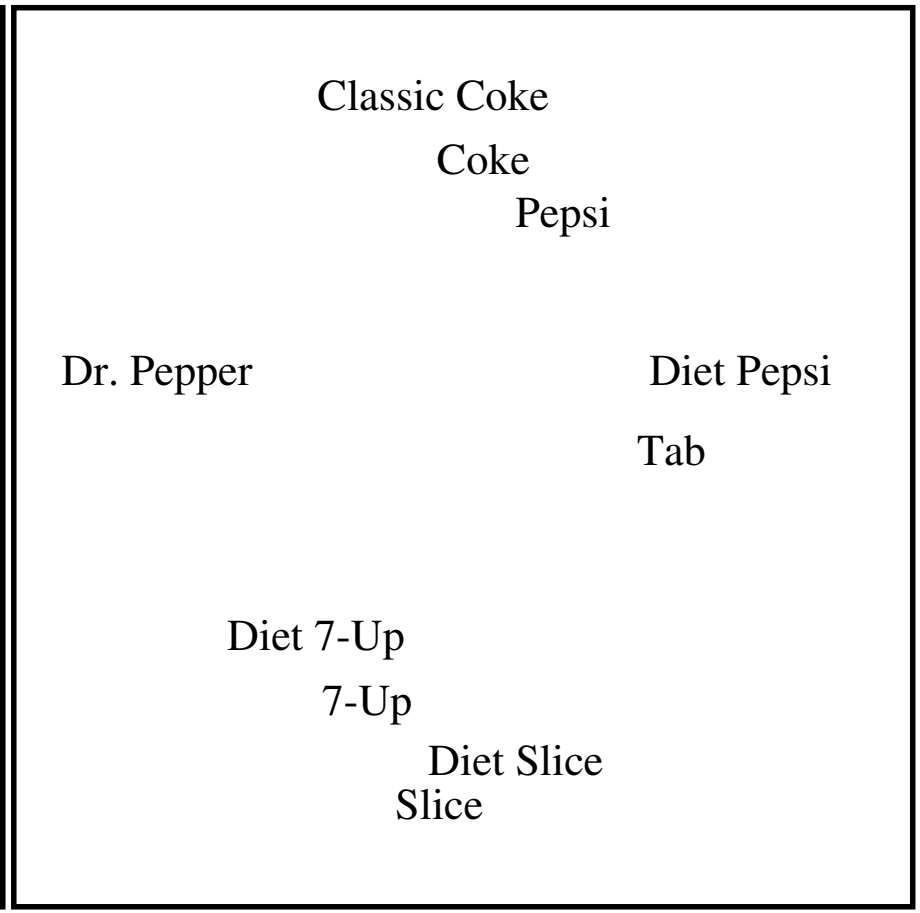
Cola data set

- Cola data set is based on experimental testing of several soft drinks.
- 38 students have tested ten ($m = 10$) different brands of soft drinks.
- Each pair was judged on its dissimilarity in a nine-point scale (1 – very similar, 9 – completely different).
- Accumulated dissimilarities form the data set.

	1	2	3	4	5	6	7	8	9	10
1. Pepsi	0	127	169	204	309	320	286	317	321	238
2. Coke	127	0	143	235	318	322	256	318	318	231
3. Classic Coke	169	143	0	243	326	327	258	318	318	242
4. Diet Pepsi	204	235	243	0	285	288	259	312	317	194
5. Diet Slice	309	318	326	285	0	155	312	131	170	285
6. Diet 7-Up	320	322	327	288	155	0	306	164	136	281
7. Dr Pepper	286	256	258	259	312	306	0	300	295	256
8. Slice	317	318	318	312	131	164	300	0	132	291
9. 7-Up	321	318	318	317	170	136	295	132	0	297
10. Tab	238	231	242	194	285	281	256	291	297	0

Cola data set visualized using multidimensional scaling





Stress Function

- The goal of multidimensional scaling is to find low-dimensional points $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, such that the distances between the points in the low-dimensional space were as close to the proximities as possible. The least-squares objective function (*raw Stress*) to be minimized can be written as

$$\sigma_r(Y) = \sum_{i<j} w_{ij}(d(Y_i, Y_j) - \delta_{ij})^2,$$

  where $Y = \{Y_1, Y_2, \ldots Y_m\}$, w_{ij} are non-negative weights.
- The *normalized Stress* is defined as follows:

$$\sigma_n(Y) = \frac{\sum_{i<j} w_{ij}(d(Y_i, Y_j) - \delta_{ij})^2}{\sum_{i<j} w_{ij}\delta_{ij}^2}.$$

  The normalization using the parameter $\sum_{i<j} w_{ij}\delta_{ij}^2$ gives a clear interpretation of the visualization quality that depends less on the number of objects m and the scale of proximities.

Relative Error

- The *relative error* is defined as follows:

$$E(Y) = \sqrt{\sigma_n(Y)} = \sqrt{\frac{\sum_{i<j} w_{ij}(d(Y_i, Y_j) - \delta_{ij})^2}{\sum_{i<j} w_{ij}\delta_{ij}^2}}.$$

  The reason for using $E(Y)$ rather than the normalized error $\sigma_n(Y)$ is that $\sigma_n(Y)$ is almost always very small in practice, so $E(Y)$ values are easier to discriminate.

- Often $w_{ij} = 1$, then previous formula becomes as follows:

$$E(Y) = \sqrt{\frac{\sum_{i<j} (d(Y_i, Y_j) - \delta_{ij})^2}{\sum_{i<j} \delta_{ij}^2}}.$$

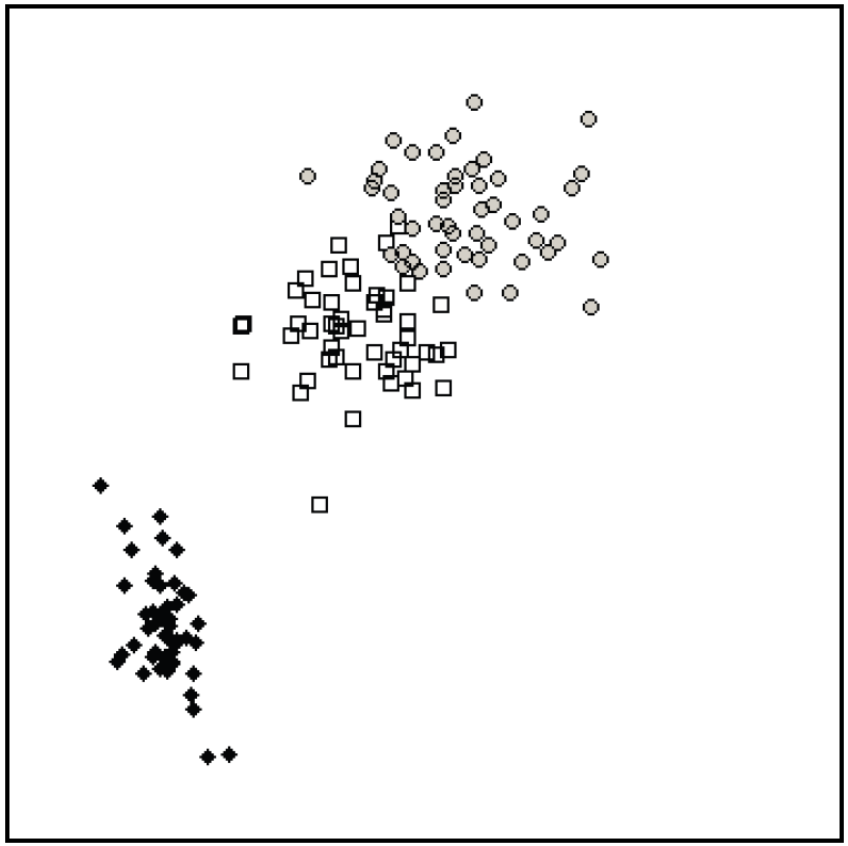
Stress-1 and Other Variants

- *Stress-1* is defined by the formula:

$$\sigma_1(Y) = \sqrt{\frac{\sum_{i<j} (d(Y_i, Y_j) - \delta_{ij})^2}{\sum_{i<j} (d(Y_i, Y_j))^2}}.$$

- There exists a multitude of variants of MDS with different weights and optimization algorithms. Various local optimization strategies will be discussed in this lecture.

Iris Data Set Visualized Using Multidimensional Scaling



SMACOF Algorithm

- The multidimensional scaling Stress function can be minimized in the majorization way. The idea of majorization is to replace iteratively the original complicated function $f(x)$ by an auxiliary function $g(x, z)$, where z is some fixed value. The function $g(x, z)$ has to meet some requirements to be called a majorizing function of $f(x)$. The auxiliary function $g(x, z)$ should be
  - simpler to minimize than $f(x)$,
  - not smaller than the original function, i.e. $f(x) \leq g(x, z)$,
  - touch $f(x)$ at the so-called supporting point z, i.e. $f(z) = g(z, z)$.

Relative Mapping

- *Relative mapping* can be used for mapping new objects, when some objects had been mapped before. It may be of interest to see where new objects are visualized among the already mapped objects.
- Such an optimization problem has a smaller number of variables and takes much shorter computing time. This is achieved by modifying the Stress function:

$$E_R(Y_{\hat{m}+1}, \ldots, Y_m) = \sum_{i,j=\hat{m}+1, i<j}^{m} w_{ij}(\delta_{ij} - d(Y_i, Y_j))^2 + \sum_{i=\hat{m}+1}^{m} \sum_{j=1}^{\hat{m}} w_{ij}(\delta_{ij} - d(Y_i, \hat{Y}_j))^2,$$

where $\hat{m}$ is the number of previously mapped points, $\hat{Y}_j$ are previously mapped points. The number of variables is $(m - \hat{m})d$ instead of md.

Relative Multidimensional Scaling

- The relative mapping may be used in the *relative multidimensional scaling method* to visualize large data sets. The visualization process is divided into three steps:
  1. The basic objects are chosen.
  2. The basic objects are visualized by the MDS algorithm.
  3. The remaining objects are visualized using the relative mapping.
- If objects are defined by multidimensional points, then the basic objects may be chosen according to k-means clustering.
- The visualization results are very dependent on the selected set of the basic objects. The basic objects should be selected so that they were distributed as uniformly as possible all over the data set, which yields better results of the obtained visualization.

Sammon's Mapping

- *Sammon's mapping* is one of the MDS methods.
- The Stress function of Sammon's mapping is as follows:

$$E_S(Y) = \frac{1}{\sum_{k<l} \delta_{kl}} \sum_{i<j} \frac{(\delta_{ij} - d(Y_i, Y_j))^2}{\delta_{ij}}.$$

- Sammon's Stress $E_S(Y)$ is coincident with the function $\sigma_r(Y)$, if

$$w_{ij} = \frac{1}{\sum_{k<l} \delta_{kl}\delta_{ij}}.$$

- Due to the normalization (division by δ_{ij}), the preservation of small values of proximities is emphasized.

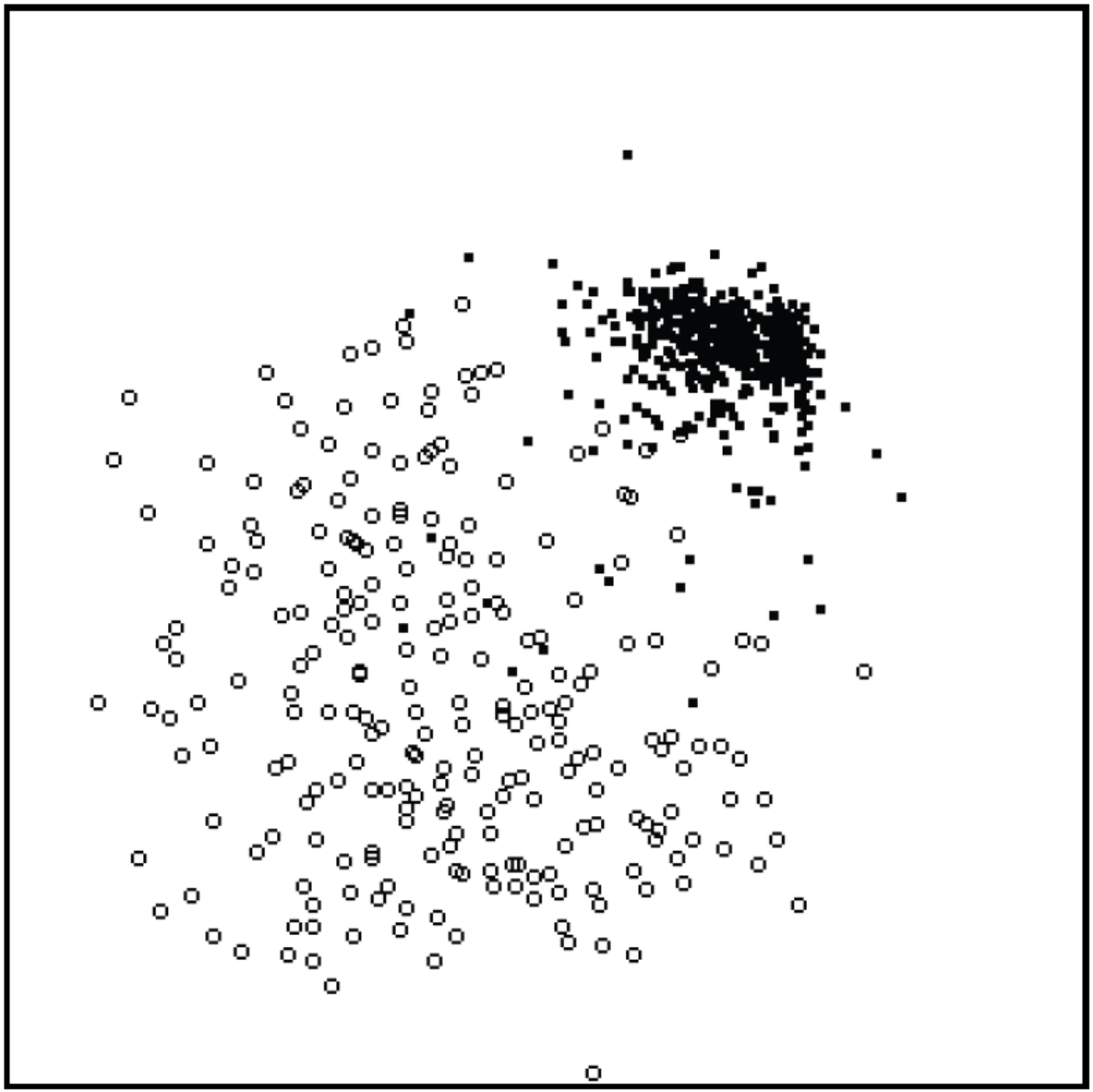
Sammon's Method

- Various optimization methods could be used to minimize the function $E_S(Y)$ when projecting multidimensional objects on the plane ($d = 2$).
- The coordinates y_{ik}, $i = 1, \ldots, m$, $k = 1, 2$, of the two-dimensional points $Y_i \in \mathbb{R}^2$ are computed by the iteration formula:

$$\begin{aligned} y_{ik}(t+1) &= y_{ik}(t) - \eta \triangle(t), \\ \triangle &= \frac{\frac{\partial E_S}{\partial y_{ik}}}{\left| \frac{\partial^2 E_S}{\partial y_{ik}^2} \right|}, \end{aligned}$$

  where t denotes the order number of iteration, η is an optimization step parameter. The coordinates of all m points $Y_i \in \mathbb{R}^2$, $i = 1, \ldots, m$, are recomputed during every iteration.
- The results of $E_S(Y)$ minimization depend on η and on the initial coordinates of points $Y_1, Y_2, \ldots, Y_m$. $\eta \in [0.3, 0.4]$ is recommended.

Breast Cancer Data Set Visualized Using Sammon's Mapping



Manifold-Based Visualization

- Most of real-life data are multidimensional, but they are not truly high-dimensional.
- Multidimensional points just lie on a low-dimensional manifold embedded into a high-dimensional space.
- A *manifold* is an abstract topological space, in which the neighborhood of each point is a subset of the Euclidean space, however the global structure of a manifold may be more complicated.
- A line and a curve are one-dimensional manifolds. The neighborhood of each point on the one-dimensional manifold is a line segment.
- A plane, the surface of a ball, and a toroid are two-dimensional manifolds, etc. The neighborhood of each point on the two-dimensional manifold is a flat region.
- The surface of the Earth is also a two-dimensional manifold.
- A manifold is a smooth low-dimensional surface embedded in a higher dimensional space.

Manifold-Based Visualization

- Multidimensional data can have meaningful hidden low-dimensional structures in the sense of lying on or near to a smooth low-dimensional manifold.
- The intrinsic dimensionality $d \ll n$ of multidimensional data is defined as the minimal number of parameters or latent variables necessary to describe the data.
- An important property of a manifold is its topology, i.e. neighborhood relationships between the subregions of the manifold.
- Nonlinear manifold learning methods are topology preserving methods. The key purpose of such methods is to preserve neighborhood relationships between points. If multidimensional points are close to each other, the points representing them in the low-dimensional space should also be close. In some cases, it is like unfolding a nonlinear manifold.

Manifold-Based Visualization Methods

- A large number of nonlinear manifold learning methods have been proposed over the last decade: locally linear embedding (LLE), isometric feature mapping (ISOMAP), Laplacian eigenmaps (LE), Hessian LLE (HLLE), etc.
- These methods are supposed to overcome the difficulties experienced with other classical nonlinear approaches. They are able to recover the intrinsic geometric structure of nonlinear multidimensional data.
- Local approaches (e.g., LLE, Laplacian eigenmaps) attempt to preserve the topology (the local geometry) of the data.
- In addition, global approaches (e.g. ISOMAP) attempt to preserve geometry at all scales: nearby multidimensional points are projected to nearby points in a low-dimensional space, and faraway multidimensional points to faraway low-dimensional points.

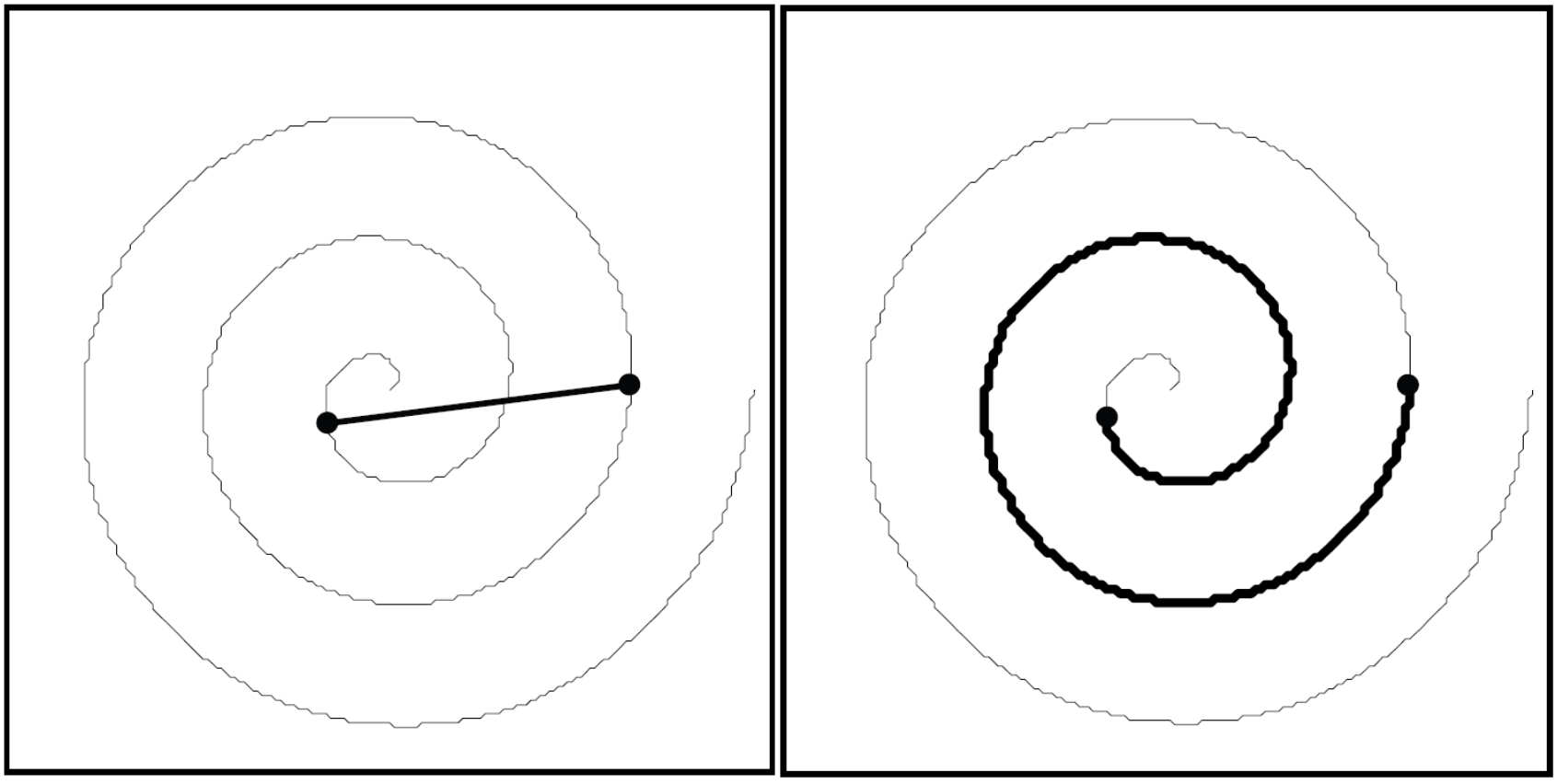
Isometric feature mapping (ISOMAP)

- *Isometric feature mapping* (ISOMAP) can be assigned to the group of multidimensional scaling. ISOMAP is designed for dimensionality reduction as well as for visualization of multidimensional data. An assumption is made that the multidimensional points are located on a lower-dimensional manifold. Therefore geodesic distances are used as a measure of proximity between the multidimensional points.
- Usually Euclidean distances between the points (as a proximity measure) are used in multidimensional scaling. In this case, the existence of a manifold is not taken into consideration.
- In ISOMAP, the geodesic distance is a proximity measure between the multidimensional points. A *geodesic distance* is the length of the shortest path between two points along the surface of a manifold.

Geodesic Distances

- In order to compute the geodesic distances between n-dimensional points from $\{X_1, X_2, \ldots, X_m\}$, it is necessary to build a weighted graph over the points that are vertices of the graph. The vertices, corresponding to the neighboring points, are connected using edges.
- The neighborhood of the point X_i can be defined:
  1. by a fixed number of the nearest points,
  2. by all the points within some fixed distance from X_i.
- The weights of edges are Euclidean distances between the corresponding points.
- Using one of the algorithms for the shortest path in the graph, for example Dijkstra's algorithm, the shortest path length between the pair of points is computed. This length is an estimate of the geodesic distance between the points.
- The matrix of geodesic distances between all multidimensional points is formed. This matrix defines dissimilarities between the objects. It can be used as data for multidimensional scaling.

Euclidean and Geodesic Distances



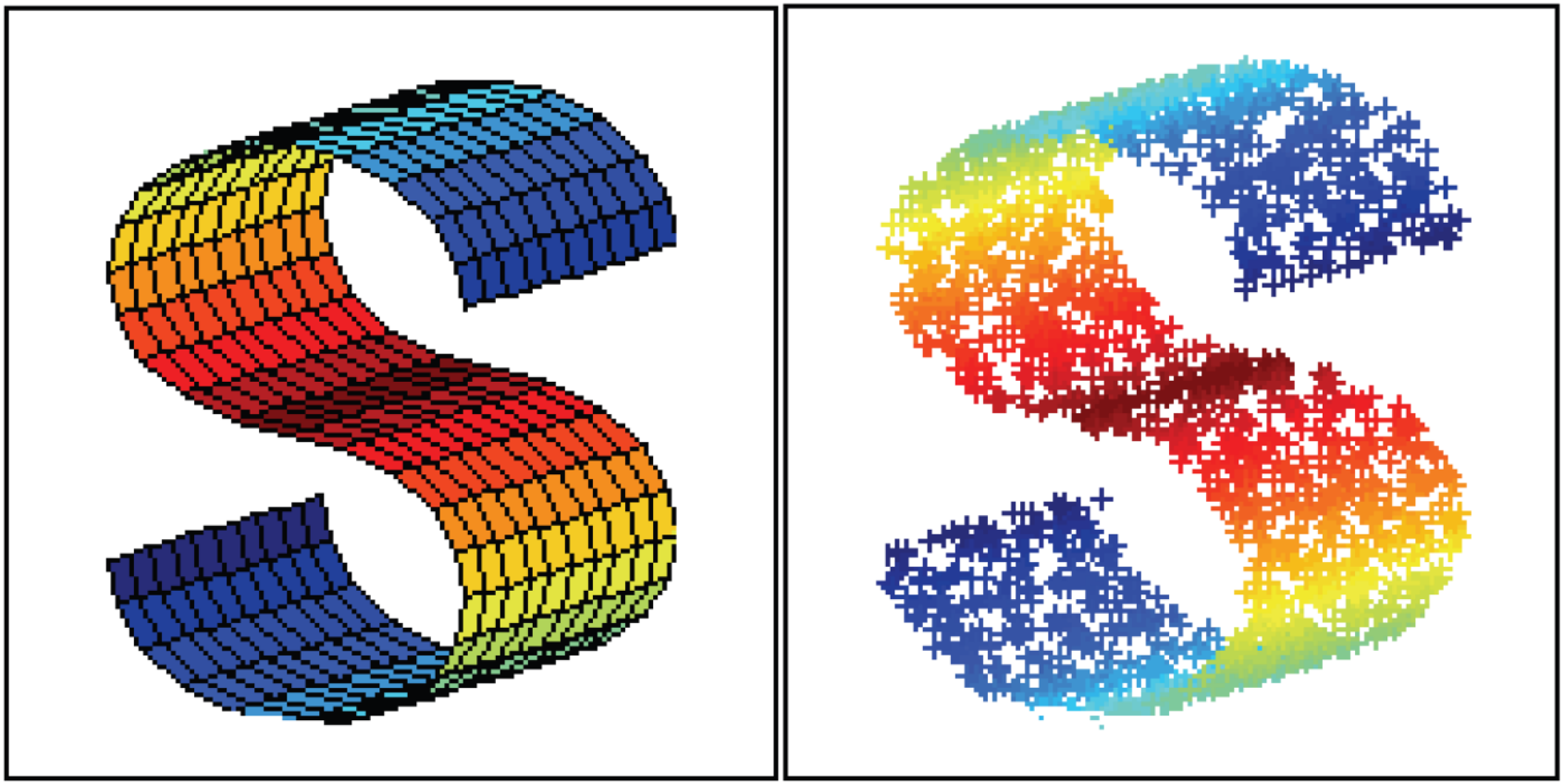
ISOMAP Algorithm

- The ISOMAP algorithm can be summarized as follows:

1. The neighbors of each multidimensional point are chosen from $\{X_1, X_2, \ldots, X_m\}$.
2. A weighted graph is constructed.
3. The geodesic distances between the pairs of all points are computed; a dissimilarity matrix is formed.
4. The projection of multidimensional points to a low-dimensional space (projection space) is obtained by multidimensional scaling.

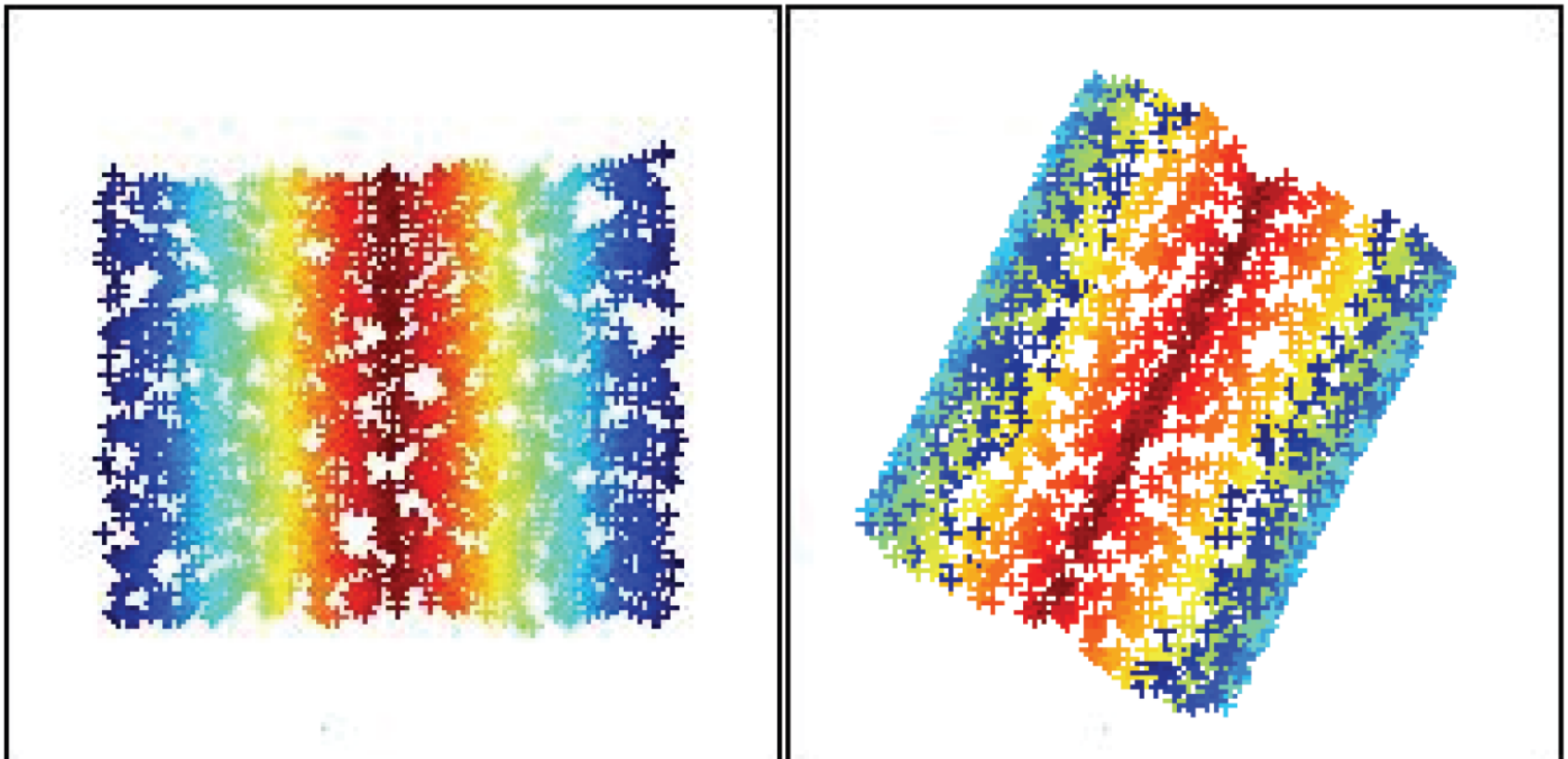
S-manifold

- The S-manifold is presented, $n = 3$. The points on the manifold are also shown, $m = 1000$.



ISOMAP and MDS Projections of S-manifold

- The structure of the manifold is well preserved by ISOMAP, because the S-manifold is unfolded: the farthest points on the manifold remain the farthest ones on the projection.
- The farthest points obtained by MDS are pale blue. These points are the farthest in multidimensional space in the sense of Euclidean distances, but they are not farthest in the sense of geodesic distances on the S-manifold.

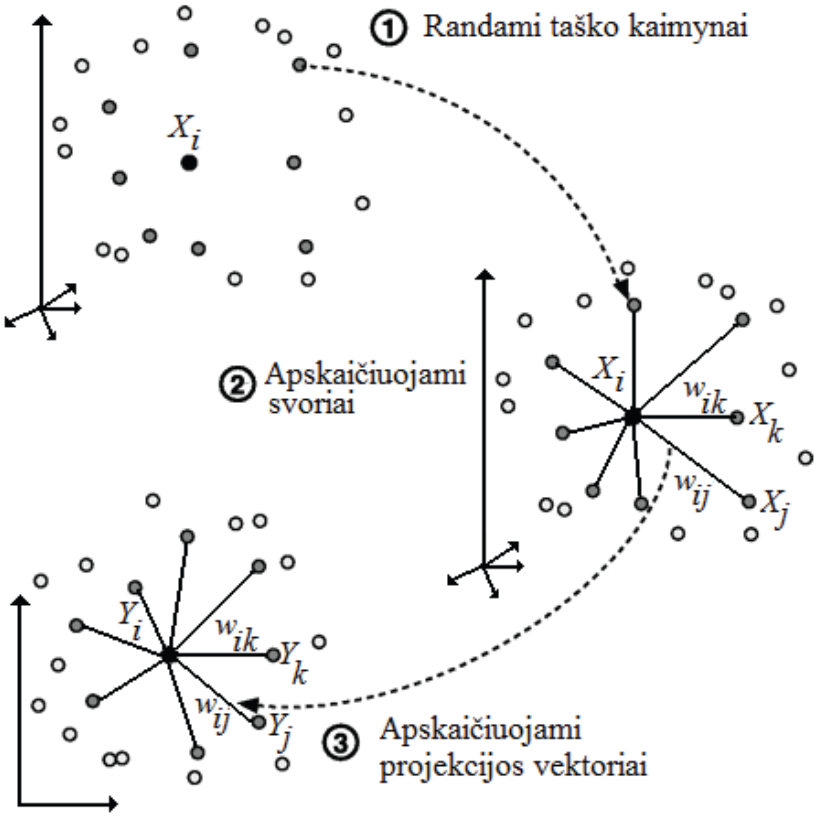


Locally Linear Embedding

- *Locally linear embedding* (LLE) is a nonlinear method for dimensionality reduction and manifold learning.
- Given a set of data points distributed on a manifold in a multidimensional space, LLE is able to project the data to a low-dimensional space by unfolding the manifold.
- LLE works by assuming that the manifold is well sampled, i.e. there are enough data, each data point and its neighbors lie on or close to a locally linear patch.
- Therefore, a data point can be approximated as a weighted linear combination of its neighbors. The basic idea of LLE is that such a linear combination is invariant under linear transformations (translation, rotation, and scaling) and, therefore, it should remain unchanged after the manifold has been unfolded to a low-dimensional space.
- The low-dimensional configuration of data points is obtained by solving two least squares optimization problems.

Scheme of Locally Linear Embedding

- The LLE algorithm transforms the set X of n-dimensional points X_i, $i = 1, \ldots, m$ ($X_i \in \mathbb{R}^n$) to a set Y of d-dimensional points Y_i, $i = 1, \ldots, m$ ($Y_i \in \mathbb{R}^d$).



LLE Algorithm

- The LLE algorithm consists of three steps.
- In the *first step*, we identify k neighbors of each data point X_i.
- As the neighbors, k-nearest points from the set X may be chosen according to the Euclidean distance.

LLE Algorithm: Second Step

- In the *second step*, we compute the weights w_{ij} that reconstruct each n-dimensional point X_i best from its neighbors minimizing the following error function:

$$E_{LLE}(W) = \sum_{i=1}^{m} \left\| X_i - \sum_{j=1}^{m} w_{ij} X_j \right\|^2 .$$

where $W = \{w_{ij},\ i, j = 1, \ldots, m\}$; $w_{ij} = 0$, if X_i and X_j are not neighbors; $\sum_{j=1}^{m} w_{ij} = 1$; $\|\cdot\|$ is the Euclidean distance. This is a typical least squares optimization problem, the minimum of which can be found by solving a linear system of equations.

LLE Algorithm: Third Step

- In the *third step*, we map each data point X_i to a low-dimensional point Y_i, which best preserves a multidimensional neighborhood geometry, represented by the weights w_{ij}. So, the weights are fixed and coordinates of low-dimensional points are sought by minimizing the following function:

$$E_{LLE}(Y) = \sum_{i=1}^{m} \left\| Y_i - \sum_{j=1}^{m} w_{ij} Y_j \right\|^2$$

subject to:

$$\sum_{i=1}^{m} Y_i = 0 \text{ and } \frac{1}{m} \sum_{i=1}^{m} Y_i^T Y_i = I,$$

where I is the identity matrix consisting of d rows and d columns.

LLE Algorithm: Third Step

- The most straightforward method for computing the d-dimensional coordinates ($d < n$) is to find the bottom $d + 1$ eigenvectors of the sparse matrix

$$\bar{M} = (I - W)^T (I - W)^T ,$$

  where I is the identity matrix, consisting of m rows and m columns.
- These eigenvectors are associated with the $d + 1$ smallest eigenvalues of $\bar{M}$. The bottom eigenvector, the eigenvalue of which is closest to zero, is the unit vector with all equal components and it is discarded. The remaining d eigenvectors form the d embedding coordinates of points Y_i.

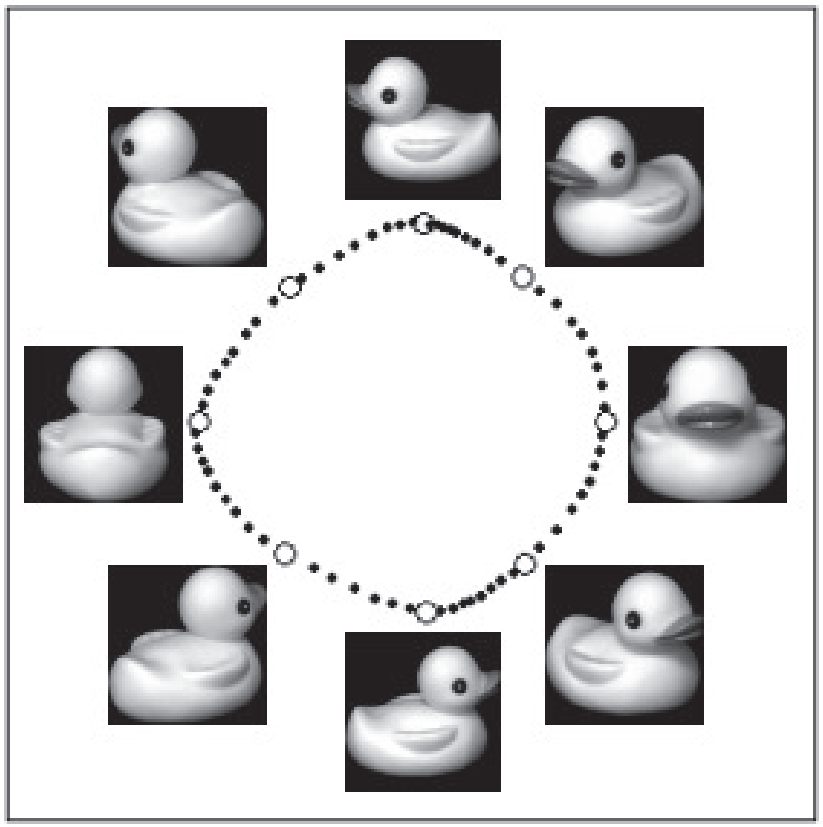
The Number of Nearest Neighbors

- The most important step to success of LLE is to choose the proper number k of the nearest neighbors for each data point. The mapping quality is rather sensitive to this parameter.
- If k is too small, a continuous manifold can falsely be divided into disjoint sub-manifolds. In this way, the mapping does not reflect any global properties.
- The large number of the nearest neighbors k causes smoothing or elimination of small-scale structures in the manifold, the mapping looses its specific character, and behaves like PCA and MDS.

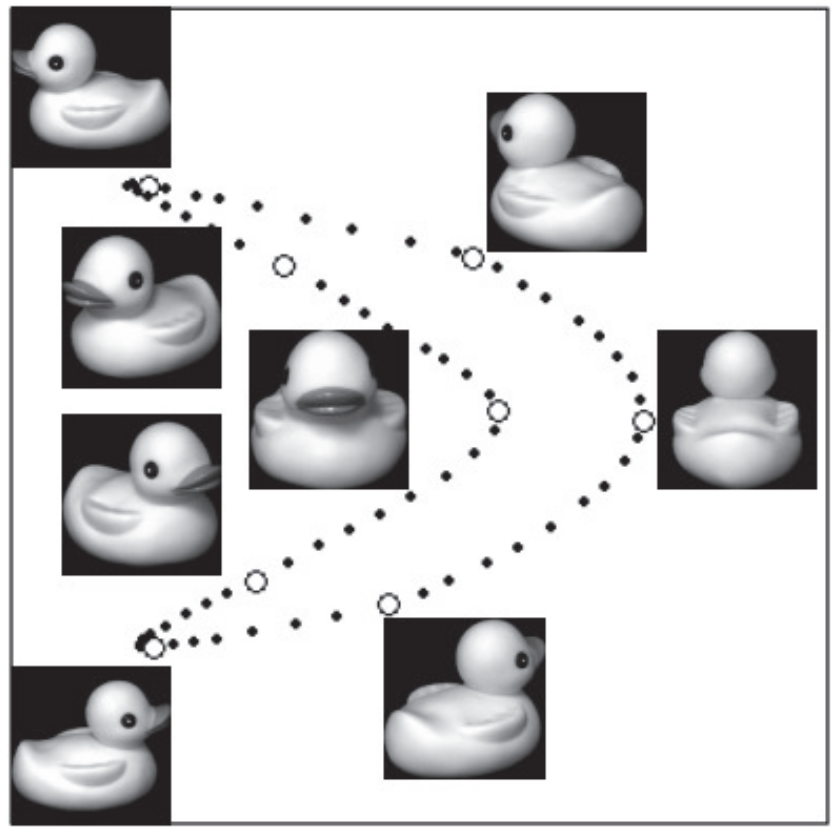
Application of LLE

- One of the applications of the LLE method is the analysis of images. For example, the data set consists of pictures of the same moving object. Features of pictures are color parameters of pixels. Since the number of pixels in the picture is usually large, the dimensionality of the analyzed data is very large.
- For example, a set of uncolored pictures, obtained by gradually (by 5°) rotating a duckling around, was analyzed. The number of pictures is $m = 72$. The pictures consist of 128×128 gray-scale pixels, therefore the dimensionality of data is $n = 16384$. Intuitively, one would expect multidimensional data that represent these pictures, to lie on a manifold parameterized by a rotation angle.
- The results of LLE are presented. Since the duckling was gradually turned around, a better representation is obtained as $k = 4$, because, in this case, we see the points on a circle.

Visualization of Pictures of a Rotating Duckling by LLE



$k = 4$

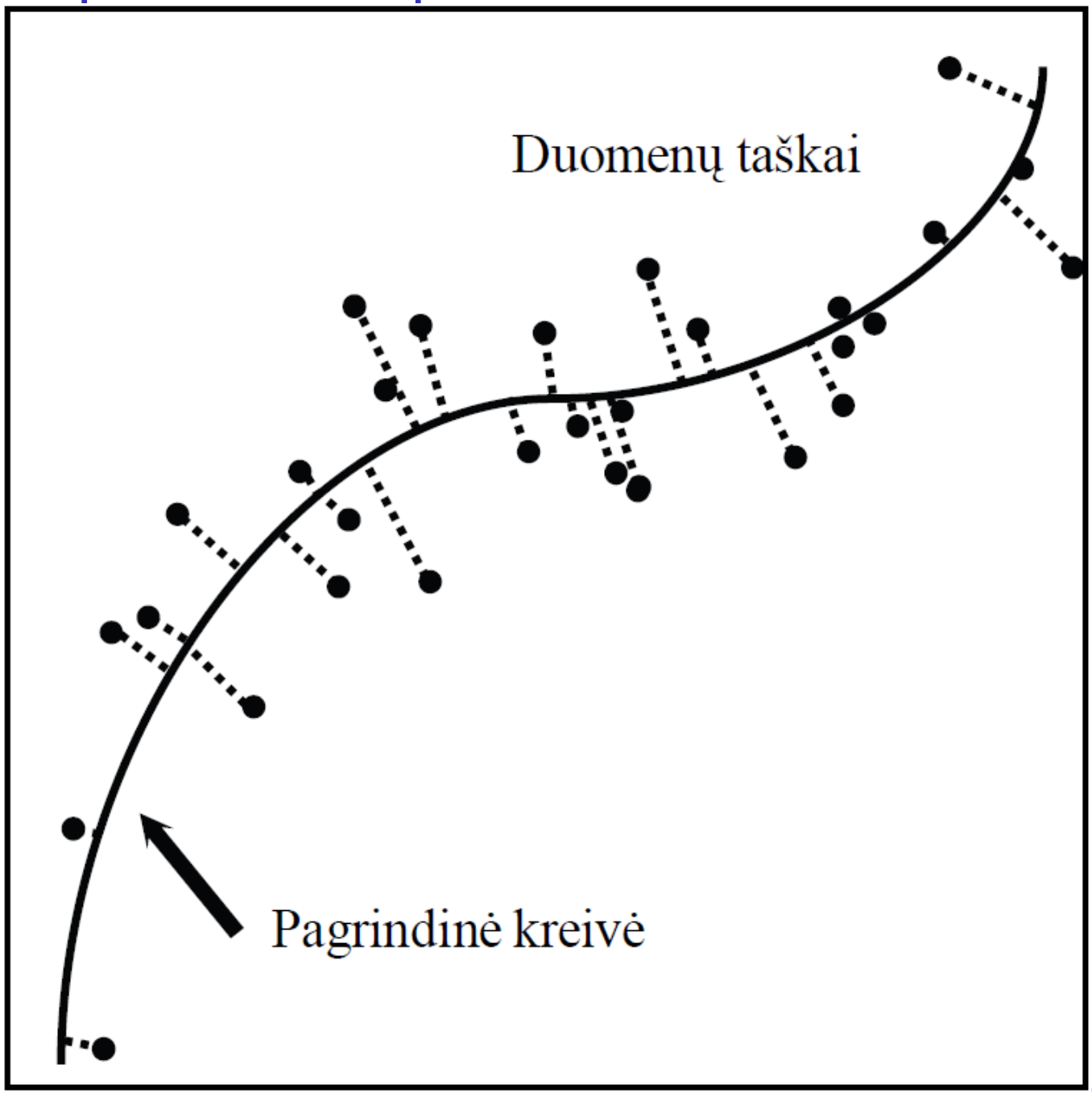


$k = 9$

Principal Curves

- Principal curves are a nonlinear generalization of principal components.
- The idea of principal curves is to find a nonlinear fit to the localized centers of some given data in n-dimensional space.
- Principal curve is a smooth one-dimensional curve that passes through the middle of the n-dimensional data: every point in the curve is the mean of the points that project onto this point.
- Principal curve minimizes the distance from the points, and provides a non-linear summary of the data.
- The principal curves approximate the data points better than PCA or linear regression.
- The idea of the principal curves can be extended to principal surfaces.

Example of Principal Curve



Multidimensional Data Visualization

Optimization-Based Visualization

Multidimensional scaling (MDS) – a technique for exploratory analysis of multidimensional data

- Pairwise dissimilarities between n objects are given by a matrix (δ_{ij}), $i, j = 1, \ldots, n$, it is supposed that $\delta_{ij} = \delta_{ji}$.
- The points representing objects in an m-dimensional embedding space $\mathbf{x}_i \in \mathbb{R}^m$, $i = 1, \ldots, n$ should be found whose inter-point distances fit the given dissimilarities.
- The problem is reduced to minimization of a fitness criterion, e.g. so called *Stress* function

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(d\left(\mathbf{x}_i, \mathbf{x}_j\right) - \delta_{ij} \right)^2,$$

where $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$; $d(\mathbf{x}_i, \mathbf{x}_j)$ denotes the distance between the points $\mathbf{x}_i$ and $\mathbf{x}_j$; weights $w_{ij} > 0, i, j = 1, \ldots, n$.

MDS is a difficult global optimization problem

- Although *Stress* function is defined by an analytical formula which seems rather simple, it normally has many local minima.
- The problem is high dimensional: $\mathbf{x} \in \mathbb{R}^N$ and the number of variables is equal to $N = n \times m$.
- *Stress* function is invariant with respect to translation, rotation and mirroring.
- Smoothness of *Stress* function depends on distances $d(\mathbf{x}_i, \mathbf{x}_j)$, however, non-differentiability normally cannot be ignored. Minkowski distances

$$d_r(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^r \right)^{1/r}.$$

Global optimization

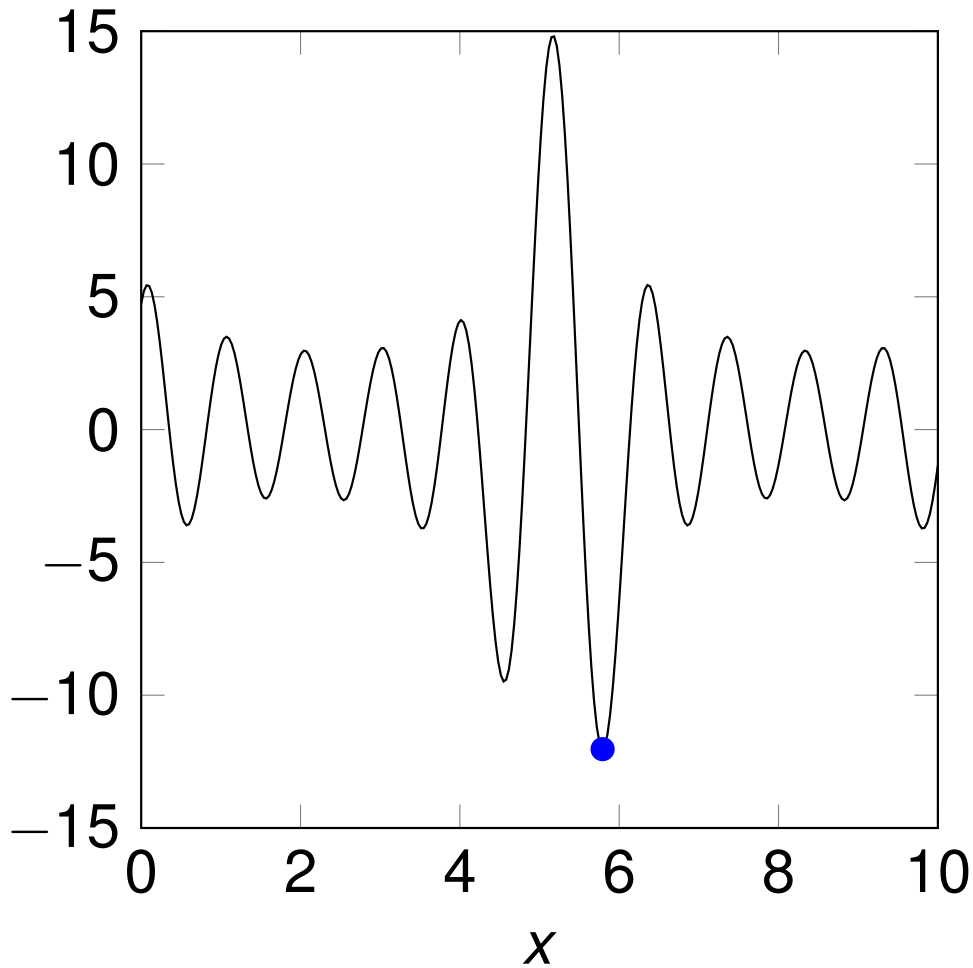
$$\text{Find } f^* = \min_{\mathbf{x}\in\mathbb{A}} f(\mathbf{x}) \text{ and } \mathbf{x}^* \in \mathbb{A},\ f(\mathbf{x}^*) = f^*, \text{ where } \mathbb{A} \subseteq \mathbb{R}^n.$$

Example:

- $n = 1$;
- $\mathbb{A} = [0, 10]$;
- Objective function

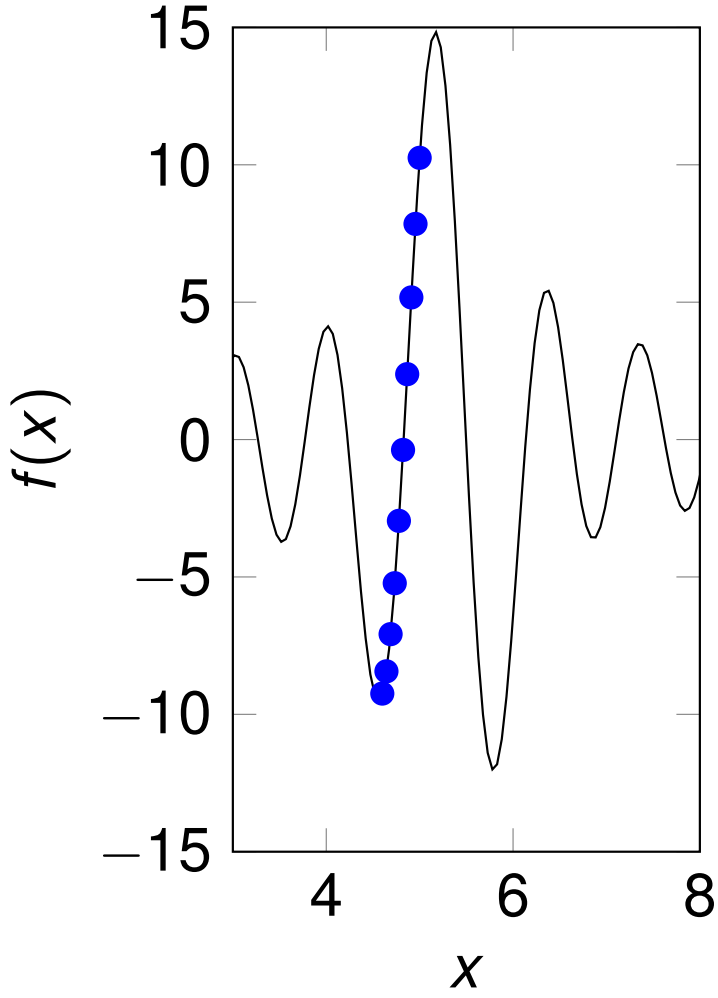
$$f(x) = -\sum_{j=1}^{5} j \sin((j+1)x+j);$$

- $f^* = -12.0312$;
- $x^* = 5.7918$.



Local optimization

- A point $\mathbf{x}^*$ is a local minimum point if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for $\mathbf{x} \in N$, where N is a neighborhood of $\mathbf{x}^*$.
- A local minimum point can be found stepping in the direction of steepest descent.
- Without additional information one cannot say if the local minimum is global.
- How do we know if we are in the deepest hole?



Algorithms for global optimization

- With guaranteed accuracy:
  - Lipschitz optimization,
  - Branch and bound algorithms.
- Randomized algorithms and heuristics:
  - Random search,
  - Multi-start,
  - Clustering methods,
  - Generalized descent methods,
  - Simulated annealing,
  - Evolutionary algorithms (genetic algorithms, evolution strategies),
  - Swarm-based optimization (particle swarm, ant colony).

Criteria of performance of global optimization algorithms

- Speed:
  - Time of optimization,
  - Number of objective function (and sometimes gradient, bounding, and other functions) evaluations,
  - Both criteria are equivalent when objective function is “expensive” – its evaluation takes much more time than auxiliary computations of an algorithm.
- Best function value found.
- Reliability – how often problems are solved with prescribed accuracy.

Parallel global optimization

- Global optimization problems are classified difficult in the sense of the algorithmic complexity theory. Global optimization algorithms are computationally intensive.
- When computing power of usual computers is not sufficient to solve a practical global optimization problem, high performance parallel computers and computational grids may be helpful.
- An algorithm is more applicable in case its parallel implementation is available, because larger practical problems may be solved by means of parallel computations.

Criteria of performance of parallel algorithms

- Efficiency of the parallelization can be evaluated using several criteria taken into account the optimization time and the number of processors.
- A commonly used criterion of parallel algorithms is the speedup:

$$s_p = \frac{t_1}{t_p},$$

  where t_p is the time used by the algorithm implemented on p processors.
- The speedup divided by the number of processors is called the efficiency:

$$e_p = \frac{s_p}{p}.$$

Covering methods

- Detect and discard non-promising sub-regions using
  - interval arithmetic $\left\{ f\left(X\right) | X \in \underline{X}, \overline{X} \in \left[\mathbb{R}, \mathbb{R}\right]^n \right\} \subseteq \overline{f}\left(\overline{X}\right)$,
  - Lipschitz condition $|f(x) - f(y)| \leq L||x - y||$,
  - convex envelopes,
  - statistical estimates,
  - heuristic estimates,
  - ad hoc algorithms.
- May be implemented using branch and bound algorithms.
- May guarantee accuracy for some classes of problems: $f^* \leq \min_{x \in D} f(x) + \epsilon$.

Branch and bound

- An iteration of a classical branch and bound processes a node in the search tree representing a not yet explored subspace.
- Iteration has three main components: selection of the node to process, branching and bound calculation.
- The bound for the objective function over a subset of feasible solutions should be evaluated and compared with the best objective function value found.
- If the evaluated bound is worse than the known function value, the subset cannot contain optimal solutions and the branch describing the subset can be pruned.
- The fundamental aspect of branch and bound is that the earlier the branch is pruned, the smaller the number of complete solutions that will need to be explicitly evaluated.

General algorithm for partition and branch-and-bound

Cover D by $L = \{L_j | D \subseteq \bigcup L_j, j = \overline{1, m}\}$ using **covering rule.**
while $L \neq \emptyset$ **do**
  Choose $l \in L$ using **selection rule**, $L \leftarrow L \backslash \{l\}$.
  if bounding rule is satisfied for l **then**
    Partition l into p subsets l_j using **subdivision rule.**
    for $j = 1, \ldots, p$ **do**
      if bounding rule is satisfied for l_j **then**
        $L \leftarrow L \bigcup \{l_j\}$.
      end if
    end for
  end if
end while

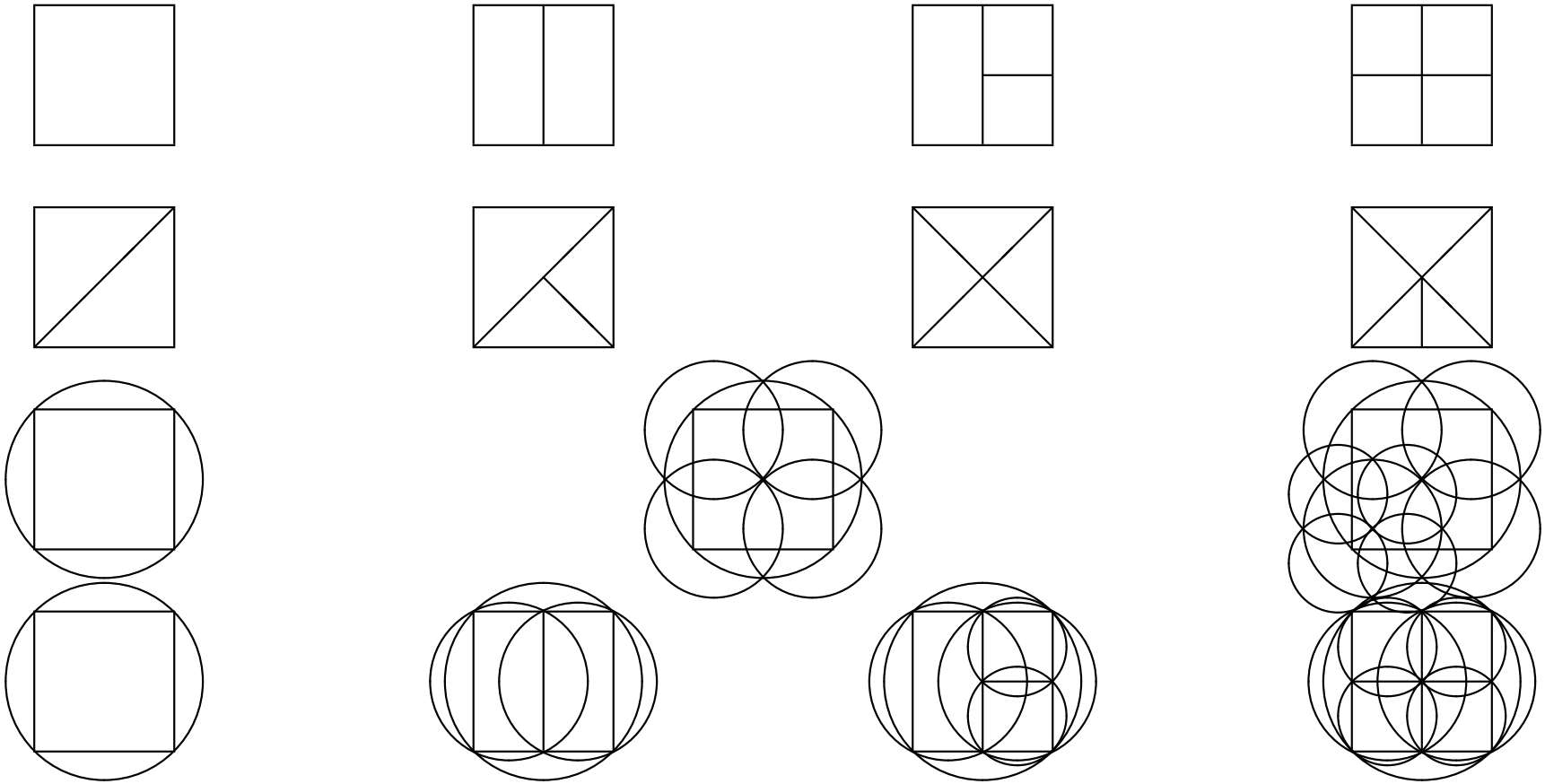
Rules of branch and bound algorithms for global optimization

- The **rules** of **covering** and **subdivision** depend on type of partitions used: hyper-rectangular, simplicial, etc.
- **Selection rules**:
  - Best first, statistical – the best candidate: priority queue.
  - Depth first – the youngest candidate: stack or without storing.
  - Breadth first – the oldest candidate: queue.
- **Bounding rule** describes how the bounds for minimum are found. For the upper bound the best currently found function value may be accepted. The lower bound may be estimated using
  - Convex envelopes of the objective function.
  - Lipschitz condition $|f(x) - f(y)| \leq L||x - y||$.
  - Interval arithmetic $\left\{ f\left(X\right) | X \in \underline{X}, \overline{X} \in [\mathbb{R}, \mathbb{R}]^n \right\} \subseteq \underline{\bar{f}}\left(\overline{X}\right)$.

Generalized branch and bound template

- Standard parts of branch and bound algorithms are implemented in the package, only specific parts should be implemented by the user.
- Suitable for implementation of branch and bound algorithms for combinatorial optimization and for covering methods of continuous global optimization.
- Parallelization tools include master-slave and distributed versions of parallel branch and bound.

Rules of covering rectangular feasible region and branching



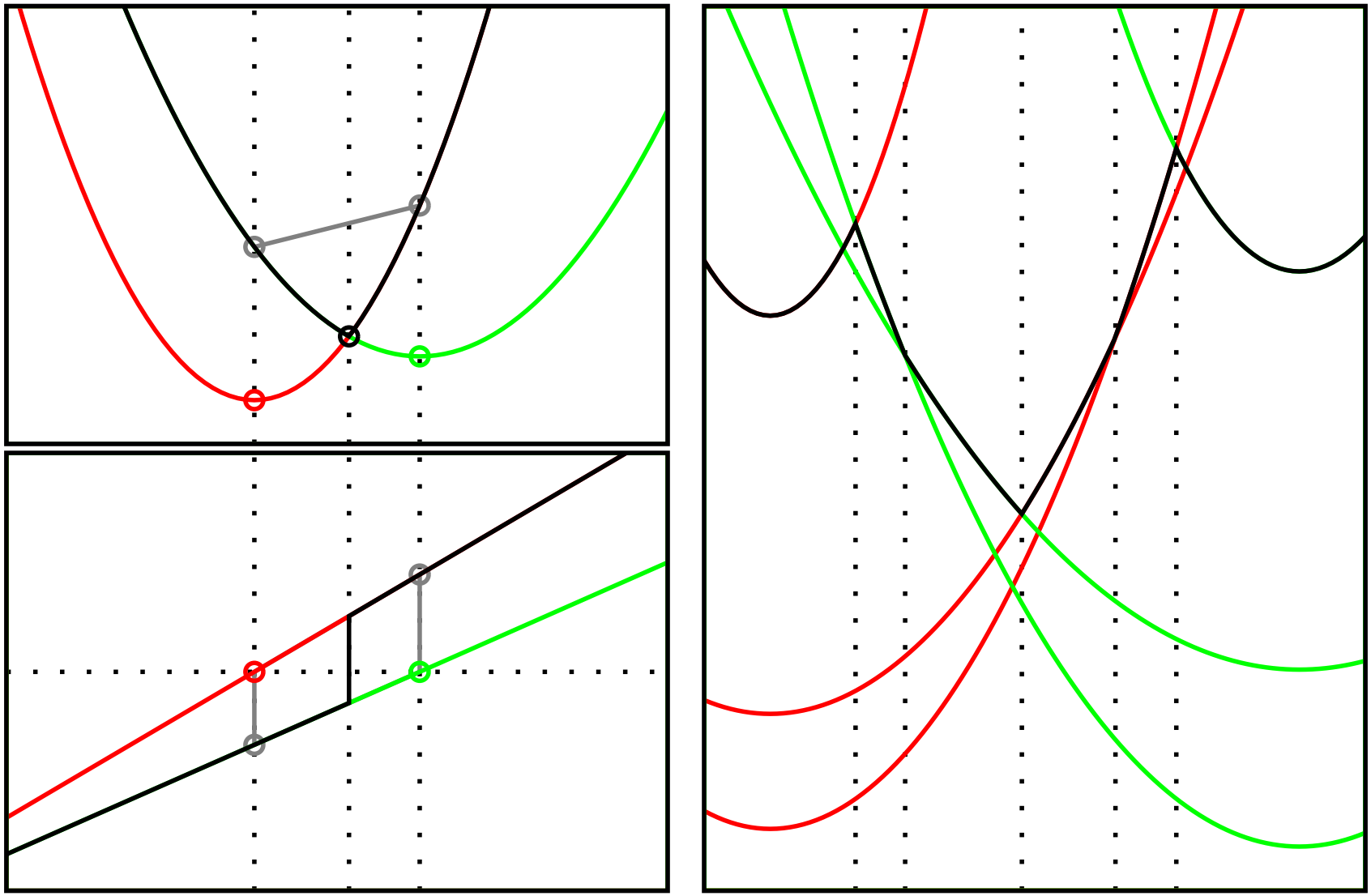
MDS with city-block distances

- If city-block distances $d_1(\mathbf{x}_i, \mathbf{x}_j)$ are used, *Stress* can be redefined as

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right| - \delta_{ij} \right)^2 .$$

- In the case of city-block distances and $m \geq 2$ *Stress* can be non-differentiable even at a minimum point. With this respect the case of city-block distances is different from the other cases of Minkowski distances when positiveness of distances $d(\mathbf{x}_i^*, \mathbf{x}_j^*)$, $i, j = 1, \ldots, n$ at a local minimum point $\mathbf{x}^*$ implies differentiability of *Stress*.
- However *Stress* with city-block distances is piecewise quadratic, and such a structure can be exploited for tailoring of ad hoc global optimization algorithms.

Piecewise quadratic function non-differentiable at a minimum point



Decomposition of the original optimization problem into a set of quadratic programming problems

- Let $A(\mathbf{P})$ denote a set such that

$$A(\mathbf{P}) = \left\{\mathbf{x} | x_{ik} \leq x_{jk} \text{ for } p_{ki} < p_{kj},\ i, j = 1, \ldots, n,\ k = 1, \ldots, m\right\},$$

where $\mathbf{P} = (p_{11}, p_{12}, \ldots, p_{mn})$, $\mathbf{p}_k = (p_{k1}, p_{k2}, \ldots, x_{kn})$ are m permutations of $1, \ldots, n$.

- For $\mathbf{x} \in A(\mathbf{P})$,

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(\sum_{k=1}^{m} \left(x_{ik} - x_{jk}\right) z_{kij} - \delta_{ij} \right)^2,$$

where

$$z_{kij} = \begin{cases} 1, & p_{ki} > p_{kj}, \\ -1, & p_{ki} < p_{kj}. \end{cases}$$

Since function $S(\mathbf{x})$ is quadratic over polyhedron $\mathbf{x} \in A(\mathbf{P})$ the minimization problem

$$\min_{\mathbf{x} \in A(\mathbf{P})} S(\mathbf{x})$$

is a quadratic programming problem. It is equivalent to

$$\min \left[- \sum_{k=1}^{m} \sum_{i=1}^{n} x_{ik} \sum_{j=1}^{n} w_{ij} \delta_{ij} z_{kij} + \right.$$

$$\left. \frac{1}{2} \sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{i=1}^{n} \left(x_{ik} x_{il} \sum_{t=1, t \neq i}^{n} w_{it} z_{kit} z_{lit} - \sum_{j=1, j \neq i}^{n} x_{ik} x_{jl} w_{ij} z_{kij} z_{lij} \right) \right]$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ik} = 0, \ k = 1, \ldots, m,$$

$$x_{\{j | p_{kj} = i+1\}, k} - x_{\{j | p_{kj} = i\}, k} \geq 0, \ k = 1, \ldots, m, \ i = 1, \ldots, n-1.$$

Quadratic programming problem

- The definition of quadratic programming problem in matrix form:

$$\min \left(-\mathbf{d}^T \mathbf{x} + \frac{1}{2} \mathbf{x}^T \mathbf{D} \mathbf{x} \right)$$

$$\begin{aligned} \text{s.t. } \mathbf{A}^0 \mathbf{x} &= 0, \\ \mathbf{A}^k \mathbf{x} &\geq 0, \ k = 1, \ldots, m, \end{aligned}$$

- where
  - $\mathbf{d}$ is a vector of dimensionality $(nm \times 1)$,
  - $\mathbf{D}$ is an $(nm \times nm)$ matrix,
  - $\mathbf{A}^0$ is an $(m \times nm)$ matrix,
  - $\mathbf{A}^k$, $k = 1, \ldots, m$, are $((n-1) \times nm)$ matrices.

Definition of quadratic programming problem

$$
\begin{aligned}
d_{kn-n+i} &= \sum_{j=1, j\neq i}^{n} w_{ij}\delta_{ij}z_{kij} \;\Bigg|\; \begin{array}{c} k=1,\ldots,m, \\ i=1,\ldots,n. \end{array} \\
D_{kn-n+i,ln-n+j} &= \begin{cases} \sum_{t=1,t\neq i}^{n} w_{it}z_{kit}z_{lit}, & i=j, \\ -w_{ij}z_{kij}z_{lij}, & i\neq j, \end{cases} \;\Bigg|\; \begin{array}{c} k,l=1,\ldots,m, \\ i,j=1,\ldots,n. \end{array} \\
A^{0}_{kj} &= \begin{cases} 1, & j=kn-n+1,\ldots,kn, \\ 0, & \text{otherwise}, \end{cases} \;\Bigg|\; \begin{array}{c} k=1,\ldots,m, \\ j=1,\ldots,mn. \end{array} \\
A^{k}_{ij} &= \begin{cases} 1, & p_{k,j-kn+n}=i+1, \\ -1, & p_{k,j-kn+n}=i, \\ 0, & \text{otherwise}, \end{cases} \;\Bigg|\; \begin{array}{c} k=1,\ldots,m, \\ i=1,\ldots,n-1, \\ j=1,\ldots,mn. \end{array}
\end{aligned}
$$

Two level MDS method with city-block distances

- Taking into account the structure of the minimization problem a two level minimization method can be applied: to solve a combinatorial optimization problem at the upper level, and to solve a quadratic programming problem at the lower level:

$$\min_{\mathbf{P}} S(\mathbf{P}), \text{ s.t. } S(\mathbf{P}) = \min_{\mathbf{x} \in A(\mathbf{P})} S(\mathbf{x}) \sim$$

$$\sim \min \left(-\mathbf{c_P}^T \mathbf{x} + \frac{1}{2} \mathbf{x}^T \mathbf{Q_P} \mathbf{x} \right) \text{ s.t. } \mathbf{E}\mathbf{x} = \mathbf{0}, \ \mathbf{A_P}\mathbf{x} \geq \mathbf{0},$$

$$\mathbf{c_P} \in \mathbb{R}^{nm}, \ \mathbf{Q_P} \in \mathbb{R}^{nm \times nm}, \ \mathbf{E} \in \mathbb{R}^{m \times nm}, \ \mathbf{A_P} \in \mathbb{R}^{(n-1) \times nm}.$$

- For the lower level problem a quadratic programming method can be applied.

Upper level combinatorial problem

- The upper level objective function is defined over the set of m-tuple of permutations of $1, \ldots, n$.
- The number of feasible solutions is $(n!)^m$, avoiding mirrored solutions it can be reduced to approximately $(n!)^m/(2^m m!)$.
- Solution of combinatorial problem:
  - explicit enumeration of all feasible solutions;
  - branch and bound;
  - pure random search;
  - multistart;
  - evolutionary algorithm.

Local search based on quadratic programming

- A minimum point of a quadratic programming problem is not necessary a local minimizer of the initial problem of minimization of *Stress*, if it is on the boundary of polyhedron.
- Therefore local search may be continued:
  - Go to the neighbor polyhedron on the opposite side of the active inequality constrains.
  - If $i, \ldots, j$ inequality constraints $\mathbf{A_P x} \geq \mathbf{0}$ are active, $i \leq p_{kt} \leq j + 1$ should be updated to $i + j + 1 - p_{kt}$.
  - Perform quadratic programming.
  - Repeat while better values are found and some inequality constrains are active.

Parallel explicit enumeration

- Each processor runs the same algorithm generating feasible solutions which should be enumerated explicitly, but only each p-th is explicitly enumerated on a processor where p is the number of processors.
- The first processor explicitly enumerates the first, $(p+1)$ and so on generated solutions. The second processor enumerates 2-nd, $(p+2)$, ... The p-th processor enumerates p-th, $2p$-th, ...
- It is assumed that generation of the solutions to be explicitly enumerated requires much less computational time than the explicit enumeration which requires solution of the lower level quadratic programming problem.
- The results of different processors are collected when the generation of solutions and explicit enumeration are finished.
- The standardized message-passing communication protocol MPI is used for communication between parallel processors.

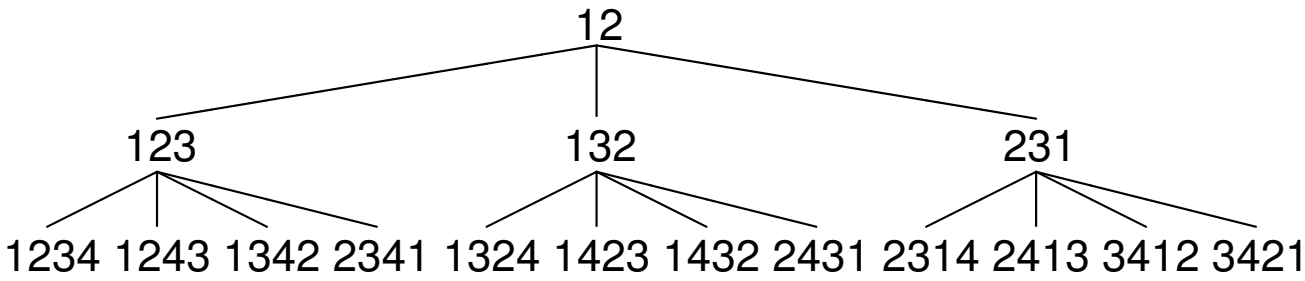
Branch and bound for MDS with city-block distances

- A subset of feasible solutions is represented by a partial solution defined by m-tuple $\overline{\mathbf{P}}$ of permutations of $1, \ldots, \overline{n}$ where only $\overline{n}$ of n objects are considered.
- Depth first selection strategy is used to avoid storing of candidate nodes.
- The lower bound is a partial *Stress* evaluated at the solution of the lower level quadratic programming problem for $\overline{n}$ objects over a polyhedron $A(\overline{\mathbf{P}})$:

$$\min_{\overline{\mathbf{x}} \in A(\overline{\mathbf{P}})} \sum_{i=1}^{\overline{n}} \sum_{j=1}^{\overline{n}} w_{ij} \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right| - \delta_{ij} \right)^2,$$

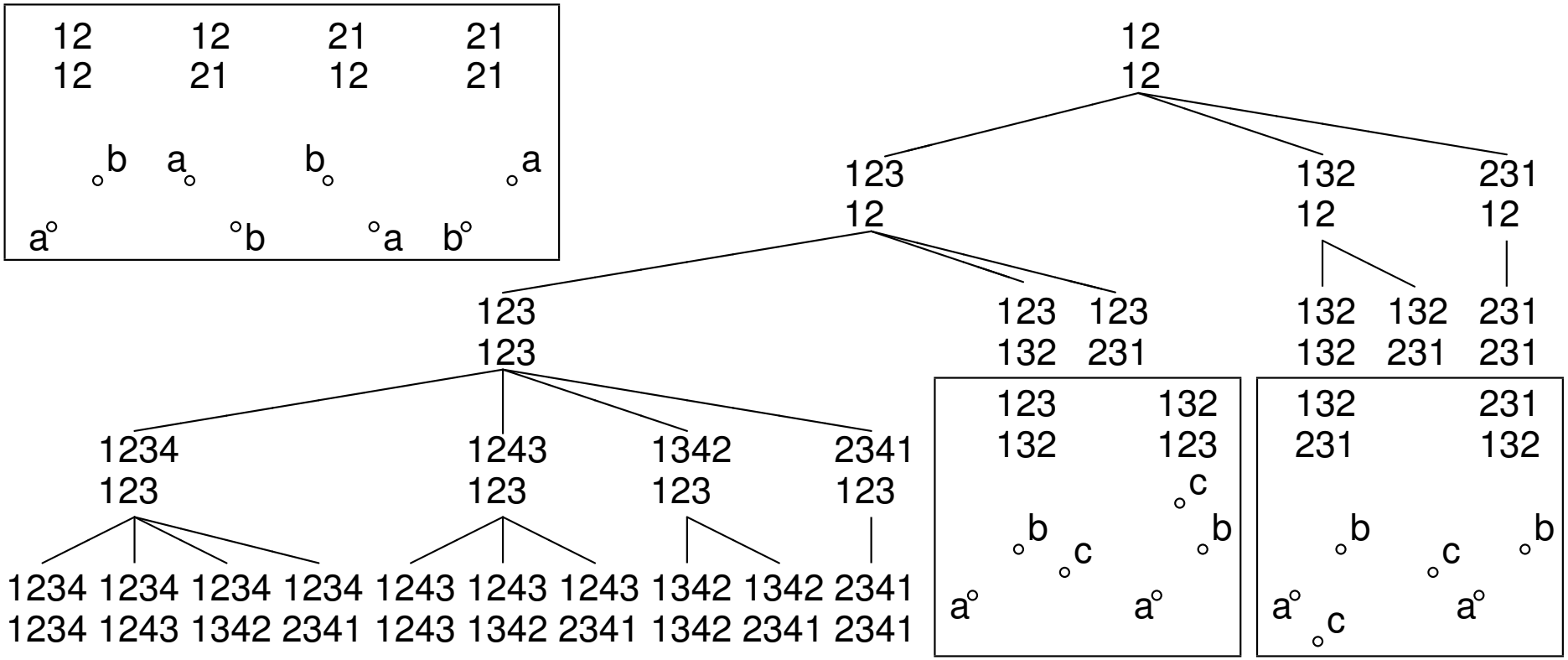
where $\overline{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_{\overline{n}})$.

A search tree for $m = 1$



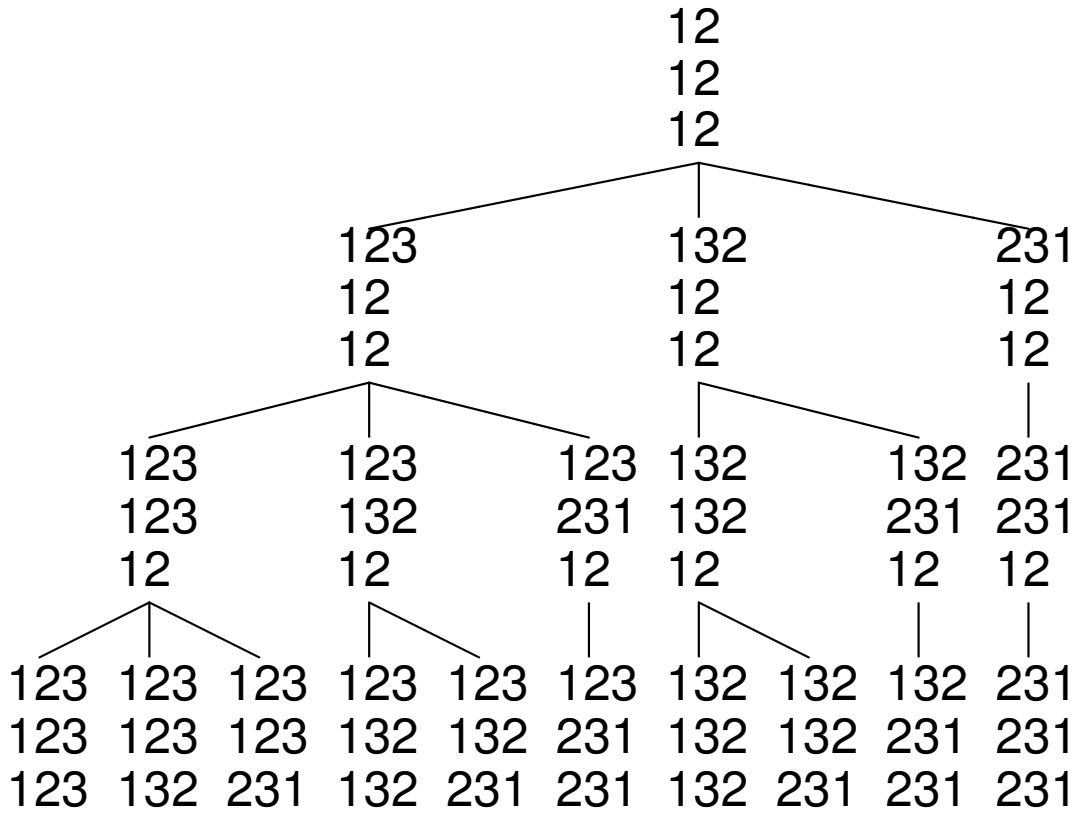
- Every numeral represents a value of p_{1i}, $i = 1, \ldots, \overline{n}$.
- To refuse mirrored solutions the search tree starts with a root node representing partial solution “12”. Although the sequence numbers of the first two objects may be changed (13 and 23) after assignment of the third object, their sequence is not changed ($1 < 2$, $1 < 3$ and $2 < 3$).
- The number of feasible solutions is $n!/2$.
- The number of nodes $\sum_{i=2}^{n} \frac{i!}{2}$.

A search tree for $m = 2$



- Every numeral represents p_{ki}, $k = 1, \ldots, m$, $i = 1, \ldots, \overline{n}$.
- The number of feasible solutions is $n!^2/8 + n!/4$.
- The lower bound for *STRESS* is evaluated only over the partial solutions defined by permutations of equal sizes.

A search tree for $m = 3$



- The number of feasible solutions is $n!^3/48 + n!^2/8 + n!/6$.

Parallel branch and bound algorithm

- Each process runs the same algorithm generating partial solutions of up to some level l: $\overline{n} \leq l$.
- Each p-th partial solution at the level $\overline{n} = l$ is evaluated and branched if needed, where p is the number of processes.
- The first process evaluates and branches the first, $(p + 1)$ and so on generated partial solutions. The second process evaluates 2-nd, $(p + 2)$, ... The p-th processor evaluates p-th, $2p$-th, ...
- The results of different processes are collected at the end of computation.
- The standardized message-passing communication protocol MPI is used for communication between parallel processes.

Evolutionary algorithm (n_p, t_c, N_{init}, p_{mut})

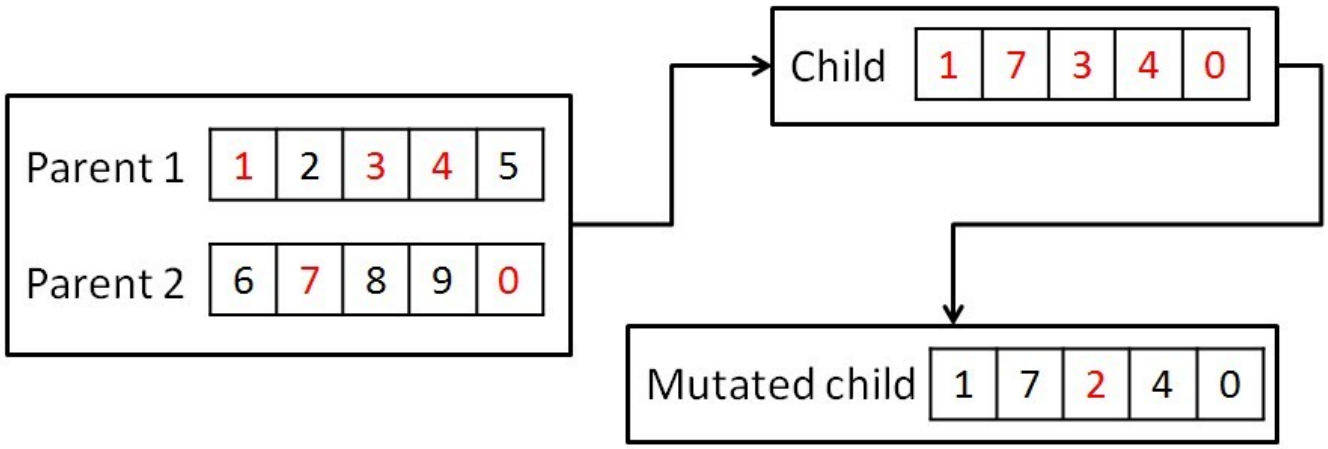
Generate N_{init} random feasible solutions.
Perform local search from n_p best solutions to form initial population.
while t_c time has not passed
    With probability p_{mut} mutate randomly chosen individual.
    Uniformly randomly generate two indices of parents i and j.
    Uniformly randomly generate two integers k and l from $1, \ldots, n$
    Compose permutations of descendant:
        elements $1, \ldots, k-1, l+1, \ldots, n$ are elements of $\mathbf{p}_i$,
        elements $k, \ldots, l$ are the missing numbers ordered in
            the same way as they are ordered in $\mathbf{p}_j$.
    Perform local search based on quadratic programming.
    If the offspring is more fitted than the worst individual,
        then the offspring replaces the latter.

Genetic operations

(1) Two parents are selected form population

(2) and combined to generate a child (Crossover).

(3) A small random change of each gene is made with probability $\frac{1}{d}$



Parallel version of evolutionary algorithm

- Multiple populations.
- Each processor runs the same evolutionary algorithm with different sequences of random numbers.
- The results of different processors are collected when search is finished after predefined time.
- Communications between processors are kept to minimum.

Multidimensional Data Visualization

Investigation of Optimization-Based Visualization

Multidimensional scaling (MDS) is a difficult global optimization problem

- The points representing objects should be found whose inter-point distances fit the given dissimilarities.
- The problem is reduced to minimization of a fitness criterion, e.g. so called *Stress* function

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(d\left(\mathbf{x}_i, \mathbf{x}_j\right) - \delta_{ij} \right)^2 .$$

- Although *Stress* function seems rather simple, it normally has many local minima.
- The problem is high dimensional: $\mathbf{x} \in \mathbb{R}^N$ and the number of variables is equal to $N = n \times m$.
- Non-differentiability normally cannot be ignored. Minkowski distances:

$$d_r(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^r \right)^{1/r} .$$

Experimental investigation of algorithms for MDS

- The accuracy of fit evaluated via minimum of $S(\mathbf{x})$ depends on n and $\delta_{ij}, i, j = 1, \ldots, n$. To reduce this undesirable impact, a relative error is used in the presentation of the results:

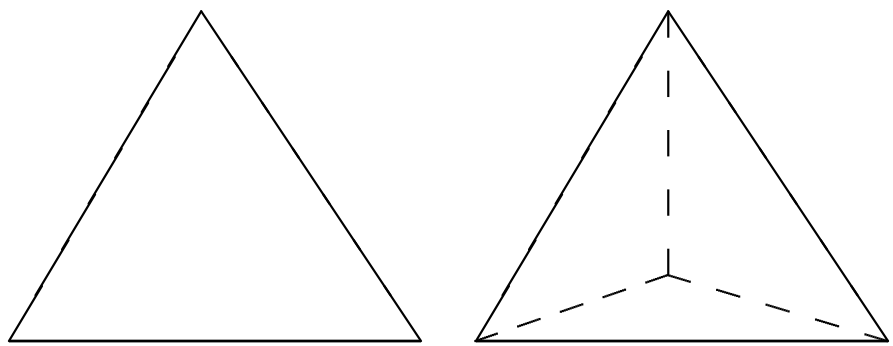
$$f(\mathbf{x}) = \sqrt{S(\mathbf{x}) \Big/ \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \delta_{ij}^2}\,.$$

- Performance is measured using
  - Time and number of quadratic programming problems solved;
  - Mean ($\overline{f^*}$) and standard deviation (s.d. f^*) of global minimum estimates;
  - Best estimate of global minimum (f^*) and percentage of runs (*perc*) when the estimate differs from f^* by less than 10^{-4}.
- Various data sets.

Data set: vertices of the standard simplex

- The distances between any two vertices are equal. $n = \dim + 1$.

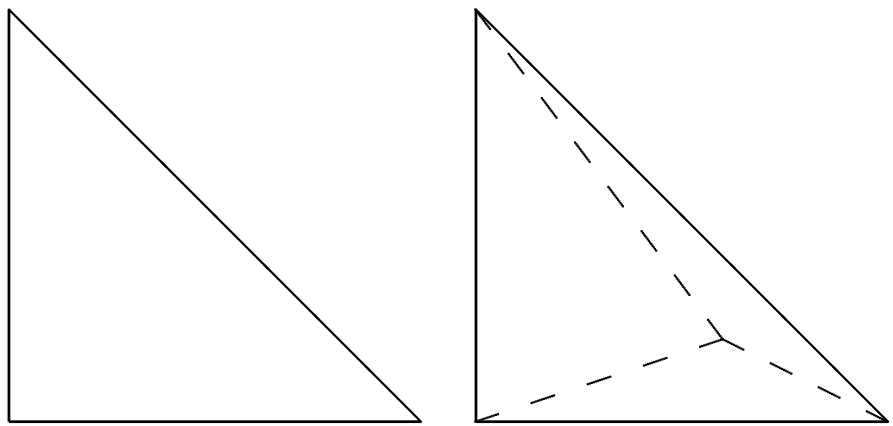
$$\Delta = \begin{pmatrix} 0 & 1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & 0 & 1 & 1 & & 1 & 1 \\ 1 & 1 & 0 & 1 & & 1 & 1 \\ 1 & 1 & 1 & 0 & & 1 & 1 \\ \vdots & & & & \ddots & & \vdots \\ 1 & 1 & 1 & 1 & \cdots & 0 & 1 \\ 1 & 1 & 1 & 1 & \cdots & 1 & 0 \end{pmatrix}.$$



Data set: vertices of the unit simplex

$$v_{ij} = \left\{ \begin{array}{ll} 1, & i = j+1, \\ 0, & \text{otherwise}, \end{array} \right. \Bigg| \; i = 1, \ldots, n, j = 1, \ldots, dim.\ n = \dim + 1.$$

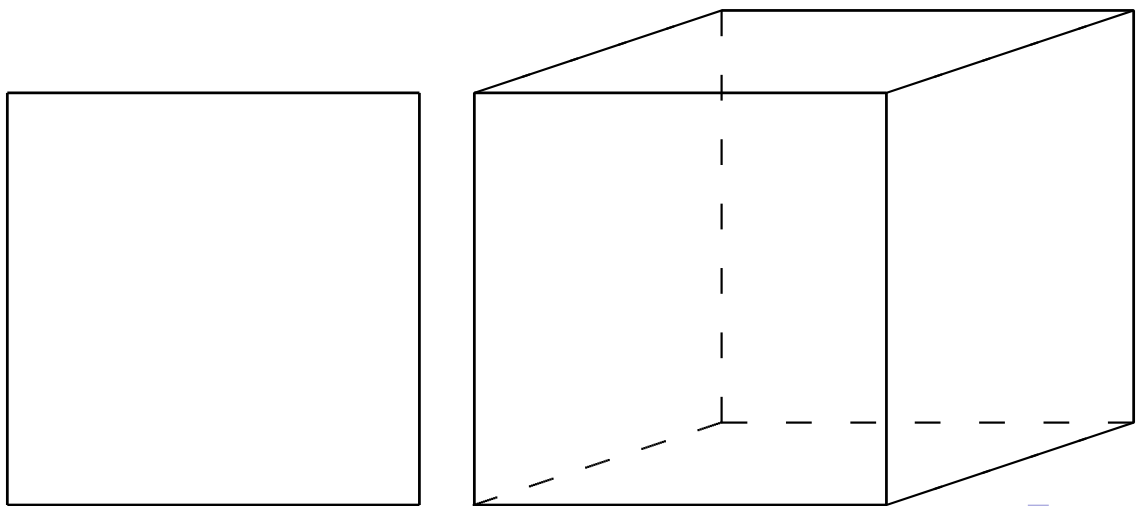
$$\Delta^1 = \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 2 & & 2 \\ 1 & 2 & 0 & & 2 \\ \vdots & & & \ddots & \vdots \\ 1 & 2 & 2 & \cdots & 0 \end{pmatrix}, \quad \Delta^2 = \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & 0 & \sqrt{2} & & \sqrt{2} \\ 1 & \sqrt{2} & 0 & & \sqrt{2} \\ \vdots & & & \ddots & \vdots \\ 1 & \sqrt{2} & \sqrt{2} & \cdots & 0 \end{pmatrix}.$$



Data set: vertices of multidimensional cube

- The coordinates of i-th vertex of a dim-dimensional cube are equal either to 0 or to 1, and they are defined by binary code of $i = 0, ..., n-1$. $n = 2^{\mathrm{dim}}$.

$$\Delta^1 = \begin{pmatrix} 0 & 1 & 1 & 2 & \cdots & n-1 & n \\ 1 & 0 & 2 & 1 & & n & n-1 \\ 1 & 2 & 0 & 1 & & n-2 & n-1 \\ 2 & 1 & 1 & 0 & & n-1 & n-2 \\ \vdots & & & & \ddots & & \vdots \\ n-1 & n & n-2 & n-1 & & 0 & 1 \\ n & n-1 & n-1 & n-2 & \cdots & 1 & 0 \end{pmatrix}.$$



Data set: error-perturbed distance data

- The data generated using uniformly distributed random points in m-dimensional space whose pairwise dissimilarities are computed by

$$\delta_{ij} = \sum_{k=1}^{m} \left| x_{ik}^{(e)} - x_{jk}^{(e)} \right|,$$

  where $x_{ik}^{(e)} = x_{ik} + N(0, e(x_{ik}))$, and $N(0, e)$ denotes the normally distributed random variable with mean zero and standard deviation e.
- Eight problems defined by all combinations of the parameters ($n = 10, 20$; $m = 2, 3$, $e(x) = 0.15x, 0.3x$) have been generated and they are referred as 'ghm'.

Empirical data sets

- A frequently used test problem for MDS algorithms is based on experimental testing of $n = 10$ soft drinks, where dissimilarities were measured by means of psychological experiment. This problem is referred as ‘cola’.
- Another frequently used test problem is confusion data for ($n = 36$) Morse codes. This problem is referred as ‘morsecodes’.

Pharmacological binding affinity data

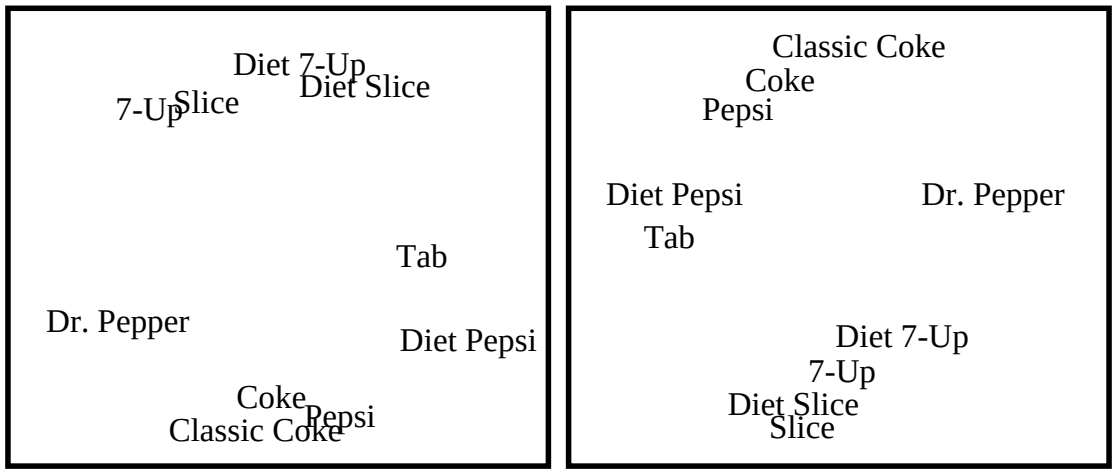
- Binding affinity data is represented through a matrix, one dimension formed by different ligands tested in a series of experiments while the other represents different proteins.
- Receptor proteins can be from different types or subtypes, or from different species, or engineered mutants of these.
- Ligands can be natural neurotransmitters or pharmacological drugs, an agonist activates, an antagonist blocks the receptor.
- Dissimilarities are computed as distances between vectors of the $\log_{10}$-transformed binding affinities.
  - Three human and five zebrafish α_2-adrenoceptor proteins, and 20 ligands (Ruuskanen et al., 2005);
  - human, rat, guinea pig and pig proteins (Uhlen et al., 1998);
  - wild type and mutant proteins (Hwa et al., 1995).

Pharmacological data

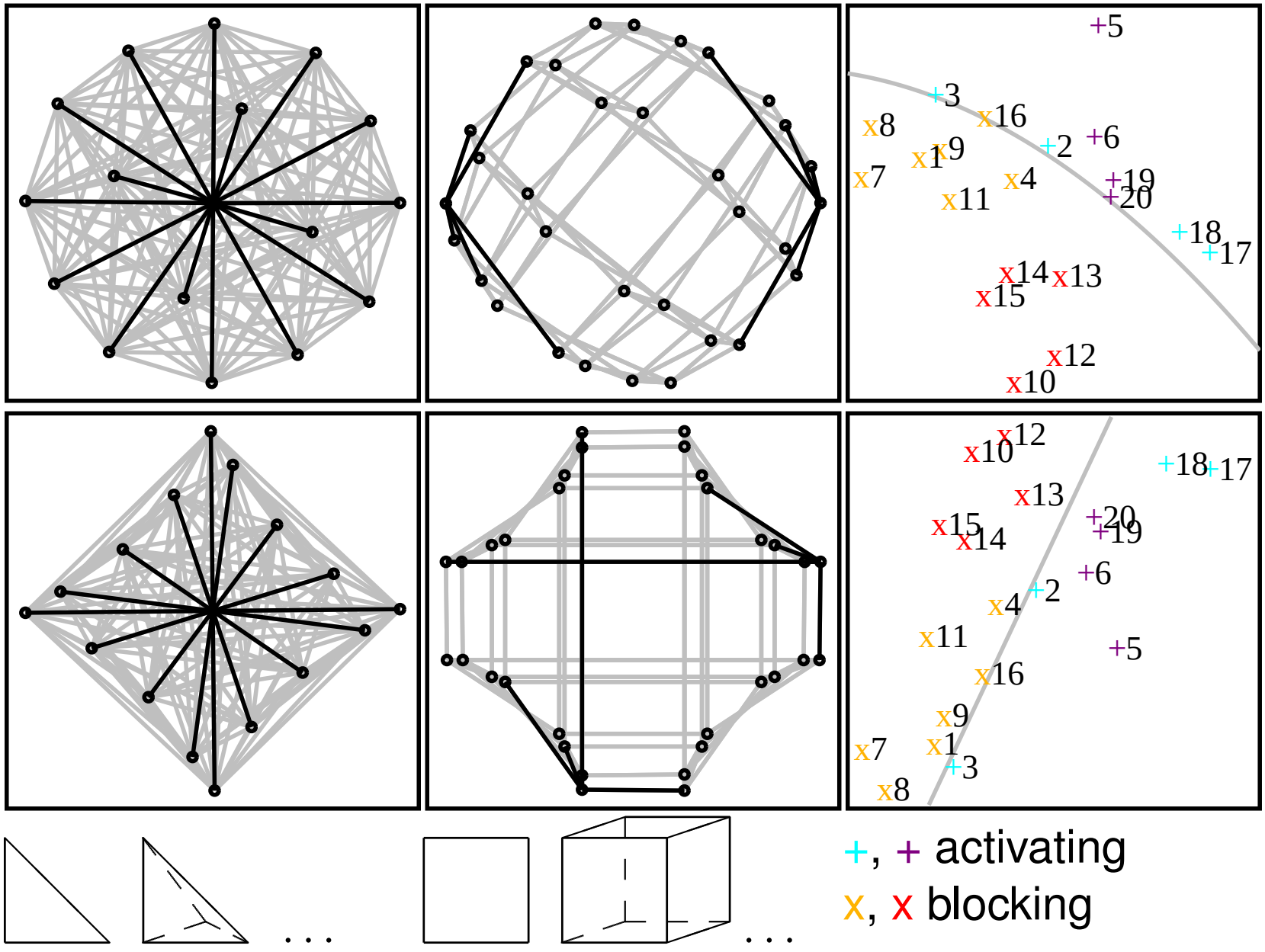
Ligand	hα_{2A}	zα_{2A}	hα_{2B}	zα_{2B}	hα_{2C}	zα_{2C}	zα_{2Da}	zα_{2Db}
1. Atipamezole	1.6	13	1.5	5.0	4.3	2.1	5.1	6.9
2. Clonidine	10	89	44	250	110	55	120	150
3. Dexmedetomidine	1.3	2.2	4.7	7.6	6.5	12	4.1	3.7
4. Idazoxan	17	85	24	40	17	17	52	94
5. Oxymetazoline	2.1	5.1	1100	1200	130	1300	1100	440
6. UK14,304	32	40	320	1200	190	700	260	280
7. L657.743	0.8	6.9	0.7	1.2	0.09	1.0	1.6	1.3
8. Rauwolscine	1.9	1.0	1.1	1.4	0.2	0.5	2.3	2.3
9. Yohimbine	5.9	5.2	7.5	9.3	4.6	3.4	6.4	4.0
10. Chlorpromazine	990	110	43	1.1	330	83	18	19
11. Clozapine	32	3.3	12	9.3	2.1	3.2	12	24
12. ARC239	2100	1800	9.6	36	66	280	55	44
13. Prazosin	1030	330	66	300	31	100	68	64
14. Spiperone	540	45	12	51	11	63	15	18
15. Spiroxatrine	320	150	2.4	93	3.1	35	11	11
16. WB-4101	5.4	11	60	51	1.9	19	31	16
17. 2-Amino-1-phenylethanol	1300	5400	4200	9400	8100	5100	3700	4000
18. Dopamine	2000	790	6300	4400	1200	3900	1300	1700
19. (–)-Adrenaline	150	140	710	910	130	1080	500	470
20. (–)-Noradrenaline	110	260	680	647	250	580	380	510

Experimental testing of soft drinks

Soft drinks	1	2	3	4	5	6	7	8	9	10
1. Pepsi	0	127	169	204	309	320	286	317	321	238
2. Coke	127	0	143	235	318	322	256	318	318	231
3. Classic Coke	169	143	0	243	326	327	258	318	318	242
4. Diet Pepsi	204	235	243	0	285	288	259	312	317	194
5. Diet Slice	309	318	326	285	0	155	312	131	170	285
6. Diet 7-Up	320	322	327	288	155	0	306	164	136	281
7. Dr Pepper	286	256	258	259	312	306	0	300	295	256
8. Slice	317	318	318	312	131	164	300	0	132	291
9. 7-Up	321	318	318	317	170	136	295	132	0	297
10. Tab	238	231	242	194	285	281	256	291	297	0



MDS with Euclidean and city-block distances



The best known estimates of global minimum

datasets		n	$\min f^*$, $m = 2$	$\min f^*$, $m = 3$
standard		8	**0.2825**	**0.1250**
simplices		12	**0.3300**	0.2013
		16	0.3525	0.2321
		20	0.3657	0.2508
unit		8	**0.2569**	**0.0992**
simplices		12	**0.3167**	0.1874
		16	0.3439	0.2243
		20	0.3595	0.2459
cubes		8	**0.2245**	**0.00**
		16	0.2965	0.1590
		32	0.3313	0.2078
	e%			
ghm	15	10	0.1293	0.0906
	30	10	0.2711	0.1298
	15	20	0.1868	0.1610
	30	20	0.2966	0.2284
cola		10	**0.1647**	0.0659
morsecodes		36	0.2944	0.1962

Reduced search space of problems exposing symmetries

- If exchange of some objects does not change dissimilarity data, exchange of points representing these objects does not change the value of *Stress* function.
- In continuous optimization: constrain sequence of first coordinate values of image points of symmetric objects.
- In combinatorial optimization: allow only some of permutations of first coordinate $\mathbf{p}_1$.
- For geometric data sets
  - standard simplex: $x_{11} \leq x_{21} \leq \ldots \leq x_{n1}$; $\mathbf{p}_1 = (1, 2, \ldots, n)$.
  - unit simplex: $x_{21} \leq x_{31} \leq \ldots \leq x_{n1}$;
    $\mathbf{p}_1 = (l, 1, 2, \ldots, l-1, l+1, \ldots, n)$.
  - hyper-cube: $x_{11} \leq x_{i1}, i = 2, \ldots n$; $p_{11} = 1$.

Explicit enumeration: standard simplices

	$m = 1$		$m = 2$		$m = 3$	
n	t, s (NQP)	f^*	t, s (NQP)	f^*	t, s (NQP)	f^*
3	0.00 (3)	0.3333	0.00 (6)	0.00	0.00 (10)	0.00
4	0.00 (12)	0.4082	0.00 (78)	0.00	0.01 (364)	0.00
5	0.00 (60)	0.4472	0.03 (1830)	0.1907	1.79 (37820)	0.00
6	0.00 (360)	0.4714	1.71 (64980)	0.2309	589.72 (7840920)	0.00
7	0.03 (2520)	0.4879	118.59 (3176460)	0.2621		
8	0.21 (20160)	0.5000	10229.00 (203222880)	0.2825		
9	2.24 (181440)	0.5092				
10	26.63 (1814400)	0.5164				
11	351.09 (19958400)	0.5222				
12	4702.00 (239500800)	0.5270				
3	0.00 (1)	0.3333	0.00 (3)	0.00	0.00 (6)	0.00
4	0.00 (1)	0.4082	0.00 (12)	0.00	0.00 (78)	0.00
5	0.00 (1)	0.4472	0.00 (60)	0.1907	0.09 (1830)	0.00
6	0.00 (1)	0.4714	0.01 (360)	0.2309	5.01 (64980)	0.00
7	0.00 (1)	0.4879	0.10 (2520)	0.2621	379.88 (3176460)	0.0945
8	0.00 (1)	0.5000	1.01 (20160)	0.2825	31681.00 (203222880)	0.1250
9	0.00 (1)	0.5092	11.89 (181440)	0.2991		
10	0.00 (1)	0.5164	153.88 (1814400)	0.3115		
11	0.00 (1)	0.5222	2121.56 (19958400)	0.3217		
12	0.00 (1)	0.5270	31170.00 (239500800)	0.3300		

Explicit enumeration: unit simplices

	$m = 1$		$m = 2$		$m = 3$	
n	t, s (NQP)	f^*	t, s (NQP)	f^*	t, s (NQP)	f^*
3	0.00 (3)	0.00	0.00 (6)	0.00	0.00 (10)	0.00
4	0.00 (12)	0.3651	0.00 (78)	0.00	0.01 (364)	0.00
5	0.00 (60)	0.4140	0.04 (1830)	0.00	2.02 (37820)	0.00
6	0.01 (360)	0.4554	2.05 (64980)	0.1869	661.11 (7840920)	0.00
7	0.02 (2520)	0.4745	137.12 (3176460)	0.2247		
8	0.23 (20160)	0.4917	11662.00 (203222880)	0.2569		
9	2.51 (181440)	0.5018				
10	29.78 (1814400)	0.5113				
11	378.45 (19958400)	0.5176				
12	5265.00 (239500800)	0.5236				
3	0.00 (2)	0.00	0.00 (4)	0.00	0.00 (7)	0.00
4	0.00 (2)	0.3651	0.00 (18)	0.00	0.01 (99)	0.00
5	0.00 (3)	0.4140	0.01 (108)	0.00	0.14 (2574)	0.00
6	0.00 (3)	0.4554	0.02 (720)	0.1869	8.49 (101160)	0.00
7	0.00 (4)	0.4745	0.25 (5760)	0.2247	695.19 (5446080)	0.00
8	0.00 (4)	0.4917	2.90 (50400)	0.2569	66686.00 (381049200)	0.0992
9	0.00 (5)	0.5018	37.16 (504000)	0.2759		
10	0.00 (5)	0.5113	560.84 (5443200)	0.2936		
11	0.00 (6)	0.5176	7813.00 (65318400)	0.3058		
12	0.00 (6)	0.5236	122360.00 (838252800)	0.3167		

Explicit enumeration: hyper-cubes

	$m = 1$		$m = 2$		$m = 3$	
n	t, s (NQP)	f^*	t, s (NQP)	f^*	t, s (NQP)	f^*
4	0.00 (12)	0.4082	0.00 (78)	0.00	0.02 (364)	0.00
8	0.24 (20160)	0.4787	12572.00 (203222880)	0.2245		
4	0.00 (6)	0.4082	0.00 (57)	0.00	0.01 (308)	0.00
8	0.06 (5040)	0.4787	5483.00 (88908120)	0.2245		

Explicit enumeration on SUN Fire E15k parallel computer: simplices, $n = 7$, $m = 2$

	standard simplex			unit simplex		
p	t, s	s_p	e_p	t, s	s_p	e_p
1	1037	1.00	1.00	1299	1.00	1.00
2	518	2.00	1.00	650	2.00	1.00
3	349	2.97	0.99	438	2.97	0.99
4	261	3.97	0.99	327	3.97	0.99
5	210	4.95	0.99	262	4.95	0.99
6	175	5.91	0.98	219	5.93	0.99
7	151	6.88	0.98	188	6.89	0.98
8	134	7.73	0.97	168	7.75	0.97
9	118	8.80	0.98	147	8.85	0.98
10	107	9.71	0.97	134	9.66	0.97
11	97	10.72	0.97	120	10.78	0.98
12	90	11.58	0.96	111	11.69	0.97
13	82	12.62	0.97	102	12.68	0.98
14	77	13.54	0.97	95	13.62	0.97
15	72	14.39	0.96	89	14.55	0.97
16	67	15.44	0.97	84	15.54	0.97
17	64	16.12	0.95	79	16.47	0.97
18	60	17.23	0.96	76	17.18	0.95
19	58	17.90	0.94	72	18.03	0.95
20	55	18.95	0.95	68	19.17	0.96
21	52	19.96	0.95	65	19.96	0.95
22	49	20.95	0.95	62	20.85	0.95
23	48	21.81	0.95	59	22.06	0.96
24	46	22.50	0.94	57	22.60	0.94

Worst case results of branch and bound: simplices, $m = 1$

	n	f^*	Enumeration: t, s (NQPP)	Branch and bound: t, s (NQPP)
standard	3	0.3333	0.00 (3)	0.00 (3)
simplices	4	0.4082	0.00 (12)	0.00 (14)
	5	0.4472	0.00 (60)	0.00 (73)
	6	0.4714	0.01 (360)	0.01 (432)
	7	0.4879	0.02 (2520)	0.05 (2951)
	8	0.5000	0.21 (20160)	0.24 (23110)
	9	0.5092	2.22 (181440)	2.47 (204549)
	10	0.5164	27.39 (1814400)	28.33 (2018948)
	11	0.5222	334.30 (19958400)	361.60 (21977347)
	12	0.5270	4687.0 (239500800)	4970.0 (261478146)
	13	0.5311	68762 (3113510400)	73714 (3374988545)
unit	3	0.00	0.00 (3)	0.00 (3)
simplices	4	0.3651	0.00 (12)	0.00 (14)
	5	0.4140	0.00 (60)	0.00 (73)
	6	0.4554	0.00 (360)	0.01 (432)
	7	0.4745	0.04 (2520)	0.03 (2951)
	8	0.4917	0.24 (20160)	0.32 (23110)
	9	0.5018	2.48 (181440)	2.77 (204549)
	10	0.5113	33.16 (1814400)	31.67 (2018948)
	11	0.5176	372.59 (19958400)	404.64 (21977347)
	12	0.5236	5208.0 (239500800)	5545.0 (261478146)
	13	0.5279	78579 (3113510400)	86436 (3374988545)

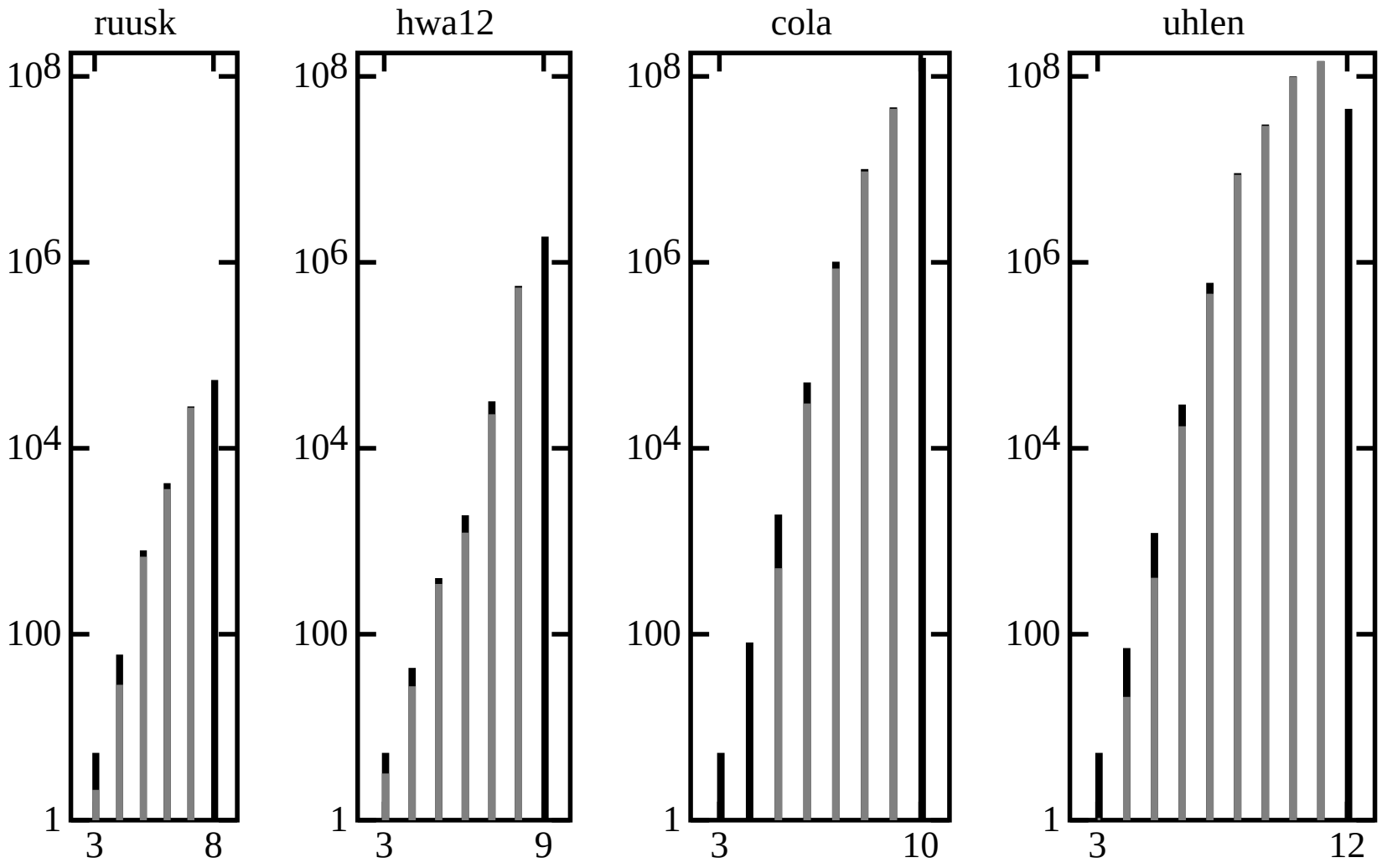
Pessimistic results of branch and bound: simplices, $m > 1$

	m	n	f^*	Enumeration: t, s (NQPP)	Branch and bound: t, s (NQPP)
standard	2	4	0.00	0.01 (78)	0.00 (63)
simplices	2	5	0.1907	0.05 (1830)	0.03 (1322)
	2	6	0.2309	1.73 (64980)	0.85 (27255)
	2	7	0.2621	113.77 (3176460)	59.61 (1655631)
	2	8	0.2825	10183 (203222880)	5107.0 (102073658)
	2	9	0.2991		502844 (3574743410)
	3	4	0.00	0.02 (364)	0.01 (133)
	3	5	0.00	1.79 (37820)	1.12 (23017)
	3	6	0.00	580.87 (7840920)	25.49 (335771)
	3	7	0.0945	301860 (2670344040)	11111 (92710201)
unit	2	4	0.00	0.00 (78)	0.00 (73)
simplices	2	5	0.00	0.05 (1830)	0.03 (662)
	2	6	0.1869	2.02 (64980)	0.51 (16076)
	2	7	0.2247	133.28 (3176460)	17.65 (422940)
	2	8	0.2569	11631 (203222880)	1675.0 (29943080)
	2	9	0.2759		134281 (1905072549)
	3	4	0.00	0.02 (364)	0.02 (313)
	3	5	0.00	2.02 (37820)	0.49 (9837)
	3	6	0.00	653.91 (7840920)	46.67 (578691)
	3	7	0.00	334788 (2670344040)	2652.0 (20674563)

Realistic results of branch and bound: cubes and empirical datasets

	m	n	f^*	Enumeration: t, s (NQPP)	Branch and bound: t, s (NQPP)
cubes	1	4	0.4082	0.00 (12)	0.00 (14)
	1	8	0.4787	0.23 (20160)	0.12 (11260)
	2	4	0.00	0.00 (78)	0.00 (73)
	2	8	0.2245	12518 (203222880)	124.68 (2157090)
	3	4	0.00	0.02 (364)	0.02 (353)
	3	8	0.00		6189 (35216122)
ruusk	1	8	0.2975	0.25 (20160)	0.02 (665)
	2	8	0.1096	12183 (203222880)	3.85 (82617)
	3	8	0.0188		838.68 (6381457)
hwa12	1	9	0.0107	3.06 (181440)	0.02 (2217)
	2	9	0.00		203.25 (2344833)
cola	1	10	0.3688	27.47 (1814400)	1.46 (65642)
	2	10	0.1647		14136 (163235214)
uhlen	1	12	0.2112	6413.0 (239500800)	0.62 (36559)
	2	12	0.0825		35951 (312924750)
hwa21	1	12	0.1790	6648.0 (239500800)	1.49 (71748)

Histograms of levels of solutions



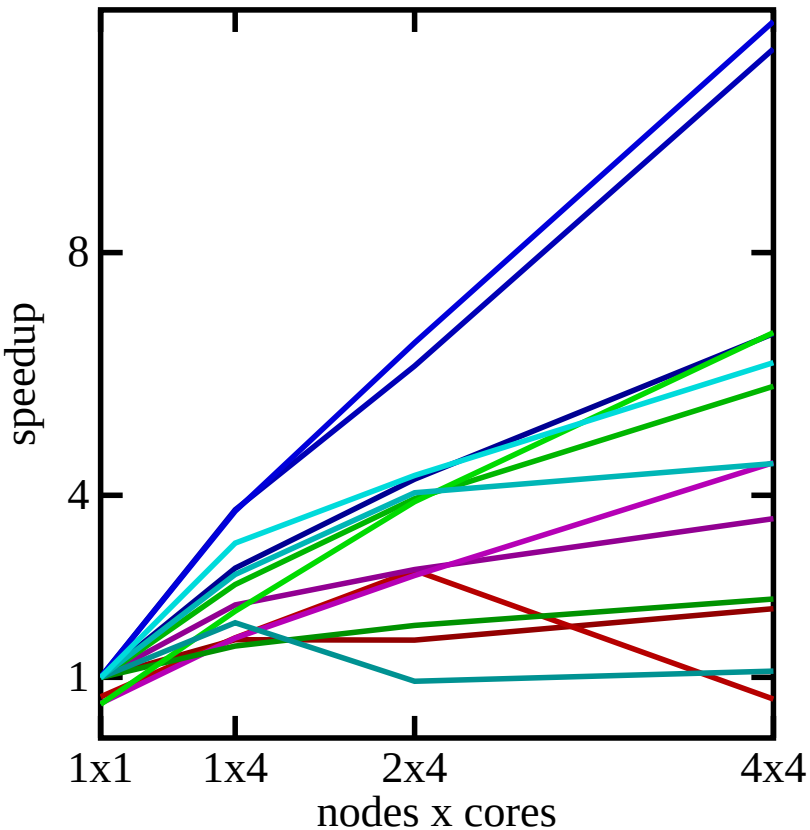
Results of parallel algorithm

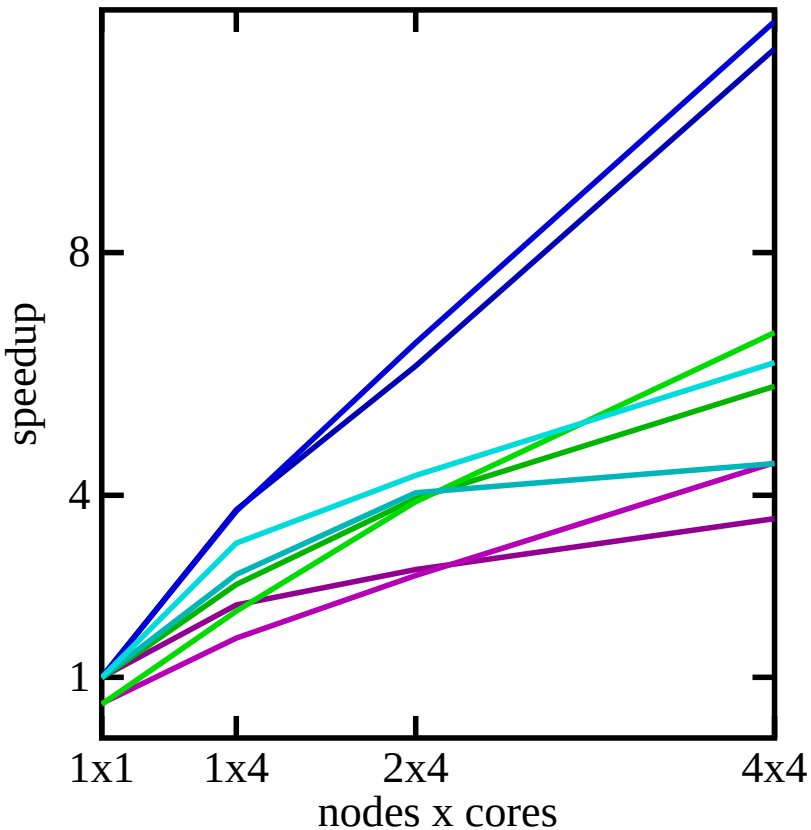
	1×1		1×4		2×4		4×4	
l	t, s	NQP	t, s	NQP	t, s	NQP	t, s	NQP
0	2.28	82617		ruusk1, $n = 8$, $m = 2$				
4	2.28	83630	1.41	180637	1.41	288547	1.07	475328
5	3.36	142800	1.38	191839	0.83	240849	3.56	302056
0	479.41	6381457		ruusk1, $n = 8$, $m = 3$				
4	479.69	6403810	218.62	10085145	172.68	14201943	132.61	21415601
5	840.87	13887913	291.15	16040461	179.02	18143183	105.64	20847563
0	120.95	2344833		hwa12, $n = 9$, $m = 2$				
4	120.84	2346237	79.52	5138135	64.96	6986355	52.61	10742968
5	121.89	2407591	47.64	3135785	30.55	3699097	20.79	5014227
6	216.78	5488888	57.66	5565997	30.95	5842364	18.04	6199196
7							612.00	208871963
0	9032	204022569		cola, $n = 10$, $m = 2$				
4	9011	204022487	3212	229324265	2108	270022713	1349	326420931
5	8991	204037437	2391	212954615	1466	226026528	792	244647590
6	8999	206189960	2405	212379122	1380	224377631	761	241643940
7	15189	396725753	4607	410841540	2700	433396504	1388	438645561
0	20515	312924750		uhlen1, $n = 12$, $m = 2$				
4	20494	312925348	10754	556642796	21847	1278648079	18532	2149661364
5	20428	312960596	7579	386113225	5054	508285836	4516	751368922
6	20522	315503838	6360	363849948	4721	461364157	3302	570468417
7	27674	506947703	17241	947746934	47190	2370684051	28521	3044562204

Speedup

$$s_p = \frac{t_1}{t_p},$$

t_p is the time used by the algorithm implemented on p cores, t_1 is the shortest time on one process.

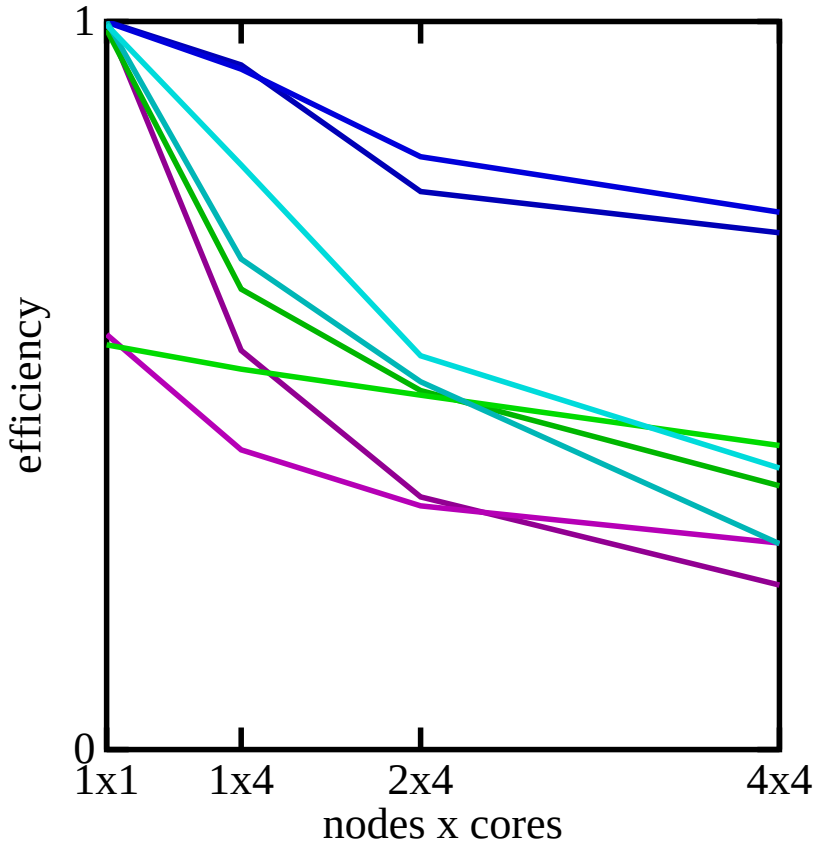


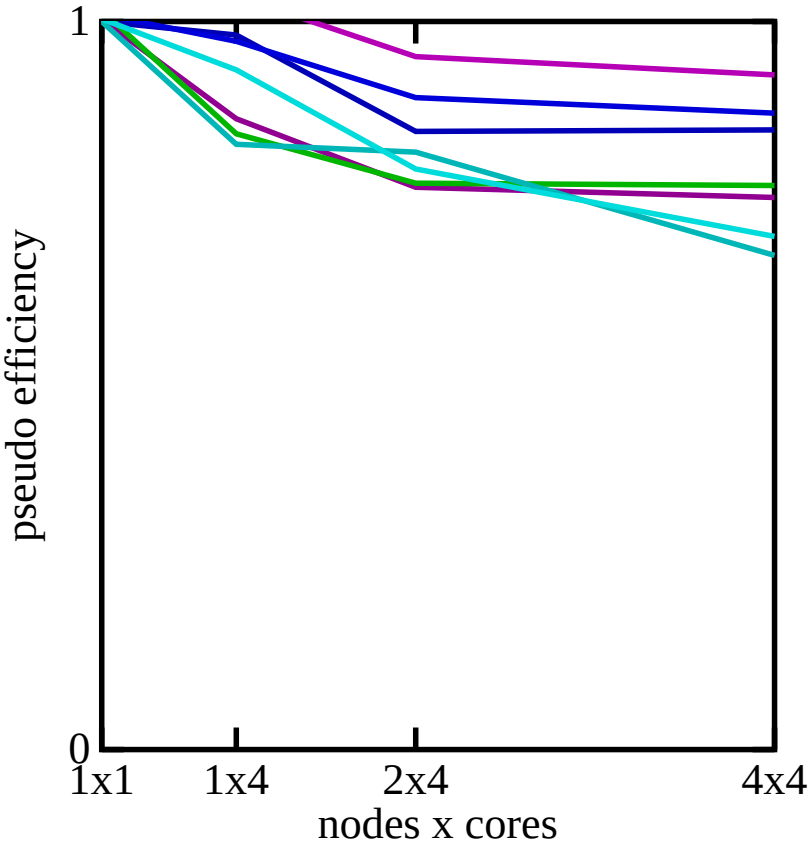


Efficiency of parallelization

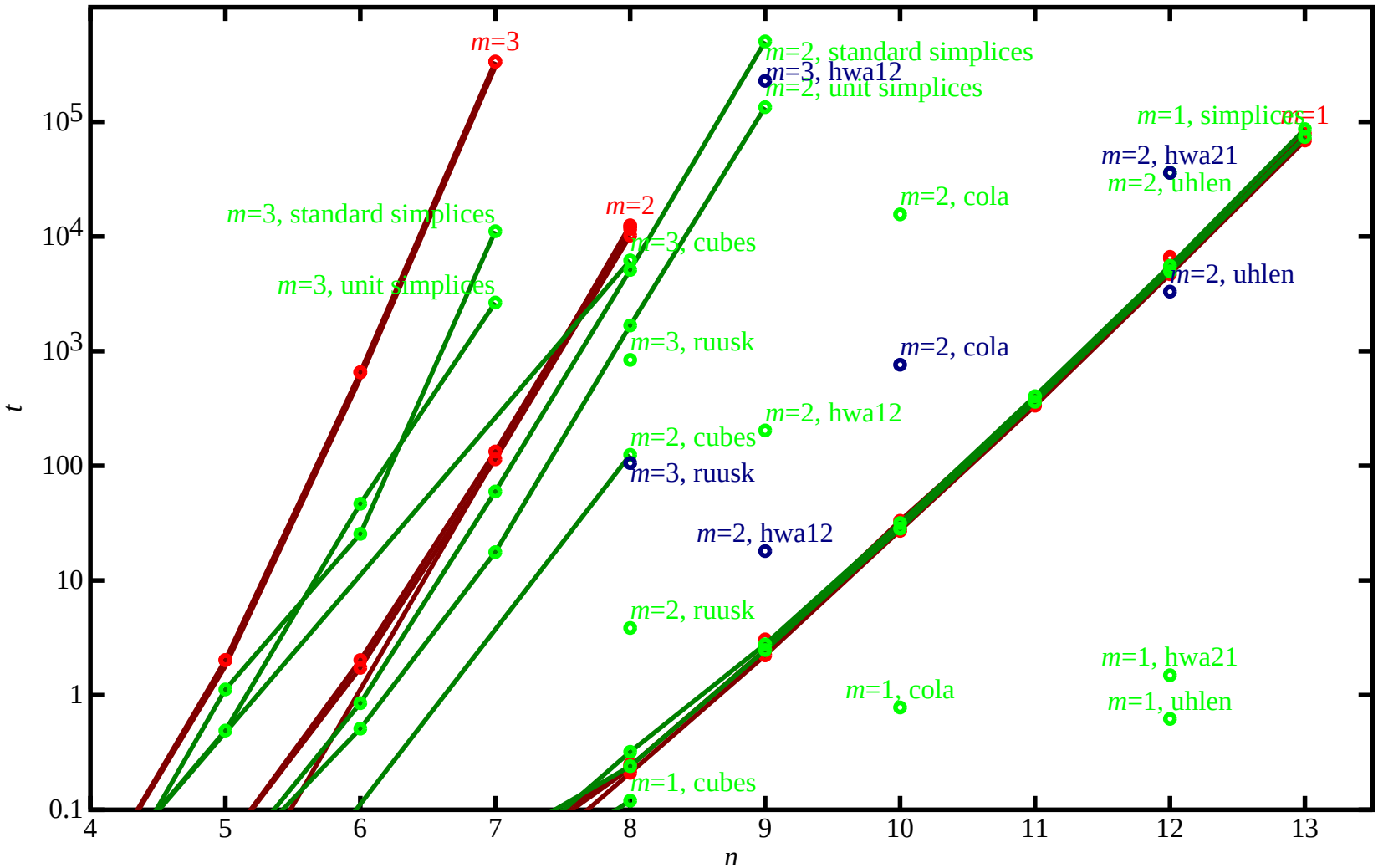
$$e_p = \frac{s_p}{p} = \frac{t_1}{p \times t_p}, \quad pe_p = \frac{t_1 / T_1}{p \times t_p / T_p},$$

T_p is the measure of amount of work done (NQP).





Results of explicit enumeration and branch and bound



Random search, multistart and evolutionary algorithms: $m = 2$, $t_c = 10$s

datasets			random search		multistart		evolutionary	
		n	$\overline{f^*}$	s.d. f^*	$\overline{f^*}$	s.d. f^*	$\overline{f^*}$	s.d. f^*
standard		8	**0.2825**	0.0000	**0.2825**	0.0000	**0.2825**	0.0000
simplices		12	0.3326	0.0008	0.3310	0.0004	**0.3301**	0.0002
		16	0.3575	0.0009	0.3550	0.0005	**0.3530**	0.0004
		20	0.3720	0.0011	0.3686	0.0004	**0.3663**	0.0003
unit		8	**0.2569**	0.0000	**0.2569**	0.0000	**0.2569**	0.0000
simplices		12	0.3218	0.0015	0.3168	0.0002	**0.3167**	0.0000
		16	0.3527	0.0016	0.3463	0.0006	**0.3440**	0.0002
		20	0.3701	0.0019	0.3627	0.0005	**0.3597**	0.0002
cubes		8	0.2304	0.0091	**0.2245**	0.0000	**0.2245**	0.0000
		16	0.3857	0.0095	0.3012	0.0021	**0.2966**	0.0002
		32	0.4753	0.0056	0.3508	0.0060	**0.3346**	0.0021
	e%							
ghm	15	10	0.1695	0.0083	**0.1293**	0.0000	**0.1293**	0.0000
	30	10	0.3084	0.0084	**0.2711**	0.0000	**0.2711**	0.0000
	15	20	0.3708	0.0166	0.1872	0.0005	**0.1868**	0.0000
	30	20	0.4282	0.0085	0.3034	0.0025	**0.2967**	0.0005
cola		10	0.1983	0.0074	0.1667	0.0012	**0.1648**	0.0003
morsecodes		36	0.4073	0.0040	0.3329	0.0063	**0.3125**	0.0048

Evolutionary algorithm: $m = 2$, $t_c = 10$s, $n_p = 60$

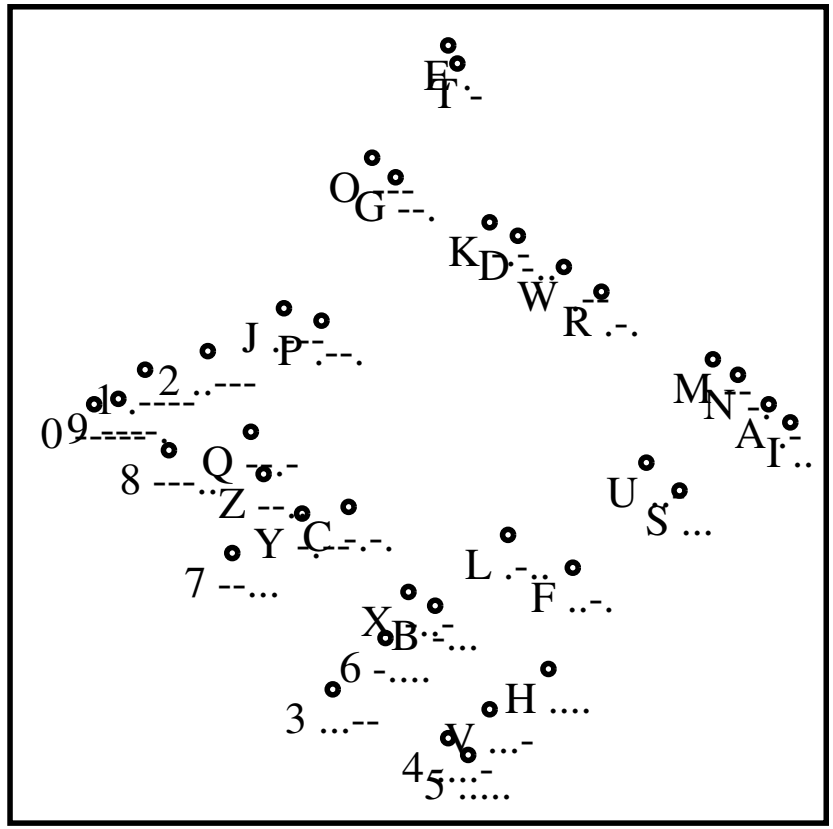
datasets			$N_{init} = 60$, $p_{mut} = 0.00$		$N_{init} = 100$, $p_{mut} = 0.00$		$N_{init} = 100$, $p_{mut} = 0.01$	
		n	$\overline{f^*}$	s.d. f^*	$\overline{f^*}$	s.d. f^*	$\overline{f^*}$	s.d. f^*
standard		8	**0.2825**	0.0000	**0.2825**	0.0000	**0.2825**	0.0000
simplices		12	0.3301	0.0002	**0.3300**	0.0002	**0.3300**	0.0001
		16	0.3530	0.0004	0.3529	0.0004	**0.3527**	0.0003
		20	0.3663	0.0003	**0.3661**	0.0002	**0.3661**	0.0003
unit		8	**0.2569**	0.0000	**0.2569**	0.0000	**0.2569**	0.0000
simplices		12	**0.3167**	0.0000	**0.3167**	0.0000	**0.3167**	0.0000
		16	**0.3440**	0.0002	**0.3440**	0.0001	**0.3440**	0.0001
		20	0.3597	0.0002	**0.3596**	0.0002	**0.3596**	0.0002
cubes		8	**0.2245**	0.0000	**0.2245**	0.0000	**0.2245**	0.0000
		16	**0.2966**	0.0002	**0.2966**	0.0002	**0.2966**	0.0001
		32	**0.3346**	0.0021	0.3354	0.0029	0.3355	0.0028
	e%							
ghm	15	10	**0.1293**	0.0000	**0.1293**	0.0000	**0.1293**	0.0000
	30	10	**0.2711**	0.0000	**0.2711**	0.0000	**0.2711**	0.0000
	15	20	**0.1868**	0.0000	**0.1868**	0.0000	**0.1868**	0.0000
	30	20	**0.2967**	0.0005	0.2968	0.0008	0.2970	0.0012
cola		10	**0.1648**	0.0003	0.1651	0.0006	0.1651	0.0006
morsecodes		36	0.3125	0.0048	0.3061	0.0027	**0.3057**	0.0028

Evolutionary algorithm and distance smoothing

datasets				evolutionary algorithm		distance smoothing	
	$e\%$	n	m	$\overline{f^*}$	s.d. f^*	$\overline{f^*}$	s.d. f^*
ghm	15	10	2	**0.1293**	0.0000	0.1457	0.0150
	30	10	2	**0.2711**	0.0000	0.2878	0.0113
	15	20	2	**0.1868**	0.0000	0.2071	0.0130
	30	20	2	**0.2970**	0.0012	0.3093	0.0076
cola		10	2	**0.1651**	0.0006	0.1823	0.0136
morsecodes		36	2	**0.3057**	0.0028	0.3106	0.0966
ghm	15	10	3	**0.0906**	0.0000	0.1116	0.0146
	30	10	3	**0.1298**	0.0000	0.1486	0.0086
	15	20	3	**0.1629**	0.0016	0.1761	0.0065
	30	20	3	**0.2394**	0.0031	0.2454	0.0063
cola		10	3	**0.0673**	0.0013	0.0914	0.0087
morsecodes		36	3	0.2231	0.0055	**0.2045**	0.0062

Comparison of evolutionary algorithm with simulated annealing: morsecodes, $m = 2$

min $S^*/2$	max $S^*/2$	time (s)
evolutionary algorithm		
153.5395	154.5550	1000
153.1380	154.0815	2000
153.0355	153.9175	10000
simulated annealing		
153.2583	155.2006	1142
153.2411	155.5416	2309

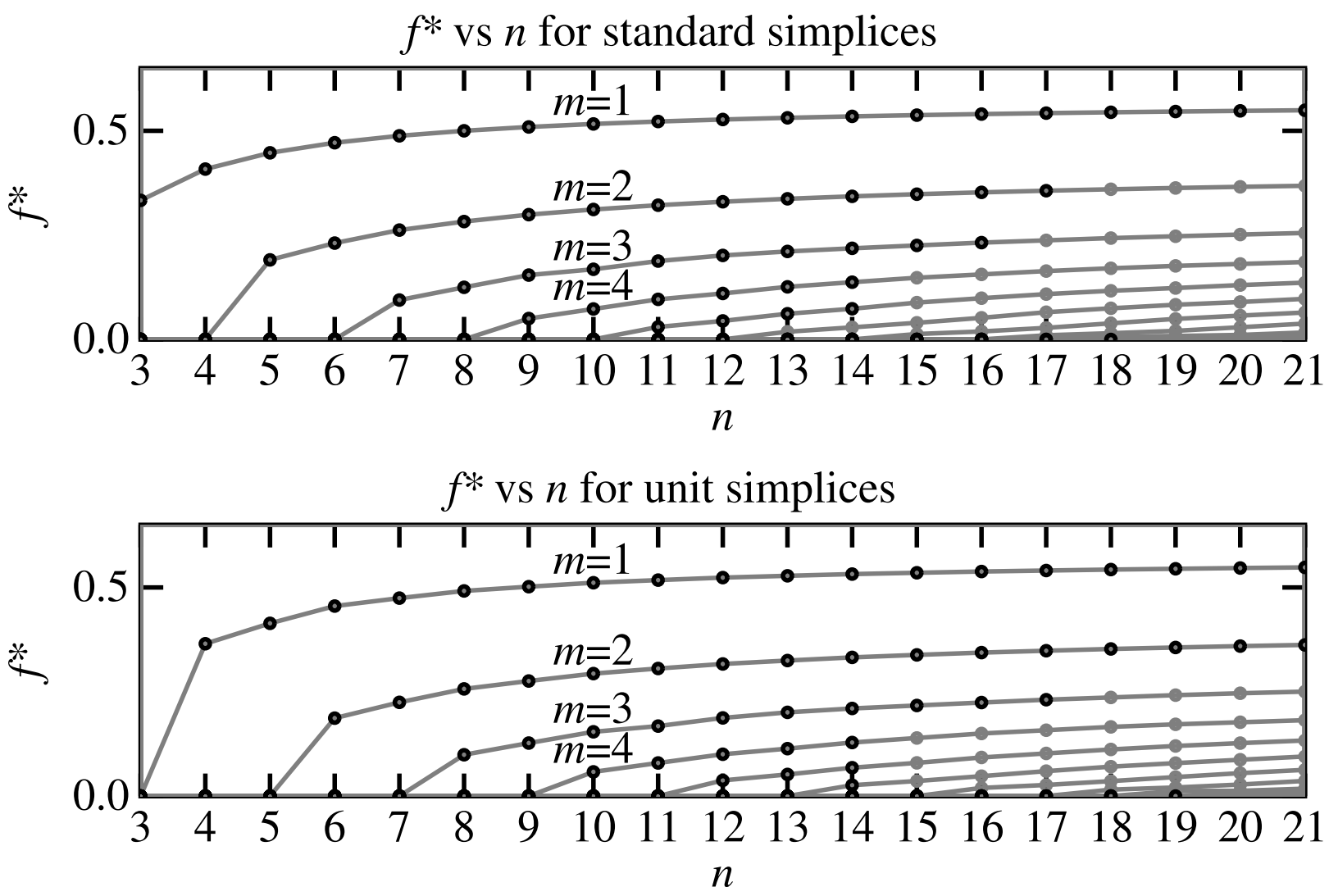


Evolutionary algorithm on SUN Fire E15k: unit simplices, $m = 2$, $t_c = 10$s

	1 processor				8 processors			
n	f^*_{min}	f^*_{mean}	f^*_{max}	*perc*	f^*_{min}	f^*_{mean}	f^*_{max}	*perc*
6	**0.1869**	0.1869	0.1869	100	**0.1869**	0.1869	0.1869	100
7	**0.2247**	0.2247	0.2247	100	**0.2247**	0.2247	0.2247	100
8	**0.2569**	0.2569	0.2569	100	**0.2569**	0.2569	0.2569	100
9	**0.2759**	0.2759	0.2759	100	**0.2759**	0.2759	0.2759	100
10	**0.2936**	0.2936	0.2936	100	**0.2936**	0.2936	0.2936	100
11	**0.3058**	0.3058	0.3058	100	**0.3058**	0.3058	0.3058	100
12	**0.3167**	0.3167	0.3167	100	**0.3167**	0.3167	0.3167	100
13	0.3249	0.3249	0.3249	100	0.3249	0.3249	0.3249	100
14	0.3325	0.3325	0.3330	93	0.3325	0.3325	0.3325	100
15	0.3384	0.3386	0.3391	70	0.3384	0.3384	0.3384	100
16	0.3439	0.3443	0.3448	25	0.3439	0.3439	0.3443	94
17	0.3484	0.3490	0.3497	8	0.3484	0.3486	0.3490	56
18	0.3526	0.3532	0.3538	3	0.3526	0.3529	0.3531	17
19	0.3562	0.3568	0.3575	2	0.3562	0.3565	0.3568	5
20	0.3597	0.3602	0.3607	4	0.3595	0.3599	0.3602	2
21	0.3625	0.3630	0.3636	4	0.3623	0.3627	0.3631	2

Evolutionary algorithm on SUN Fire E15k: $t_c = 30\text{s}/p$

	$p=1$		$p=4$		$p=8$		$p=12$		$p=16$	
n	*perc*	f^*	*perc*	f^*	*perc*	f^*	*perc*	f^*	*perc*	f^*
standard simplices										
5	100	**0.1907**	100	**0.1907**	100	**0.1907**	100	**0.1907**	100	**0.1907**
6	100	**0.2309**	100	**0.2309**	100	**0.2309**	100	**0.2309**	100	**0.2309**
7	100	**0.2621**	100	**0.2621**	100	**0.2621**	100	**0.2621**	100	**0.2621**
8	100	**0.2825**	100	**0.2825**	100	**0.2825**	100	**0.2825**	100	**0.2825**
9	100	**0.2991**	100	**0.2991**	100	**0.2991**	100	**0.2991**	100	**0.2991**
10	99	**0.3115**	100	**0.3115**	100	**0.3115**	100	**0.3115**	100	**0.3115**
11	95	**0.3217**	100	**0.3217**	100	**0.3217**	100	**0.3217**	100	**0.3217**
12	79	**0.3300**	100	**0.3300**	100	**0.3300**	100	**0.3300**	98	**0.3300**
13	60	0.3371	95	0.3371	86	0.3371	52	0.3371	29	0.3371
14	45	0.3429	87	0.3429	34	0.3429	6	0.3429	1	0.3429
15	35	0.3481	20	0.3481	2	0.3481	1	0.3481	1	0.3484
16	26	0.3525	7	0.3525	1	0.3527	1	0.3527	1	0.3527
unit simplices										
6	100	**0.1869**	100	**0.1869**	100	**0.1869**	100	**0.1869**	100	**0.1869**
7	100	**0.2247**	100	**0.2247**	100	**0.2247**	100	**0.2247**	100	**0.2247**
8	100	**0.2569**	100	**0.2569**	100	**0.2569**	100	**0.2569**	100	**0.2569**
9	100	**0.2759**	100	**0.2759**	100	**0.2759**	100	**0.2759**	100	**0.2759**
10	100	**0.2936**	100	**0.2936**	100	**0.2936**	100	**0.2936**	100	**0.2936**
11	100	**0.3058**	100	**0.3058**	100	**0.3058**	100	**0.3058**	100	**0.3058**
12	100	**0.3167**	100	**0.3167**	100	**0.3167**	100	**0.3167**	100	**0.3167**
13	91	0.3249	100	0.3249	100	0.3249	77	0.3249	62	0.3249
14	92	0.3325	100	0.3325	49	0.3325	20	0.3325	13	0.3325
15	69	0.3384	61	0.3384	4	0.3384	3	0.3384	2	0.3384
16	64	0.3439	8	0.3439	1	0.3439	1	0.3443	3	0.3443
cubes										
8	100	**0.2245**	100	**0.2245**	100	**0.2245**	100	**0.2245**	100	**0.2245**
16	35	0.2965	1	0.2965	1	0.2965	1	0.2974	1	0.3009


f* vs n for standard simplices
f*
0.5
0.0
m=1
m=2
m=3
m=4
3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
n
f* vs n for unit simplices
f*
0.5
0.0
m=1
m=2
m=3
m=4
3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
n

Multidimensional Data Visualization

Dimensionality of Embedding Space

Multidimensional scaling (MDS) is a difficult global optimization problem

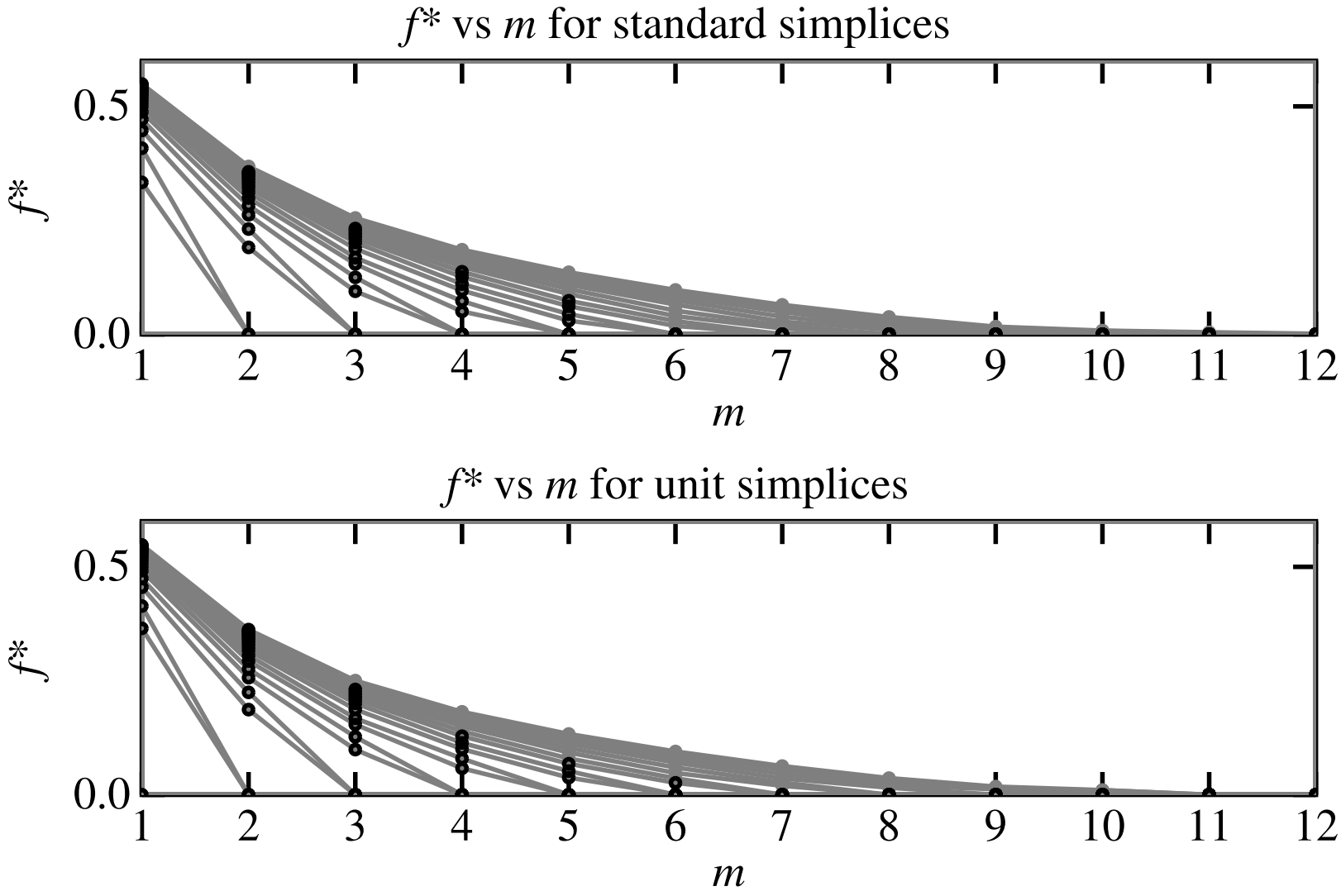
- The points representing objects should be found whose inter-point distances fit the given dissimilarities.
- The problem is reduced to minimization of a fitness criterion, e.g. so called *Stress* function

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(d\left(\mathbf{x}_i, \mathbf{x}_j\right) - \delta_{ij} \right)^2 .$$

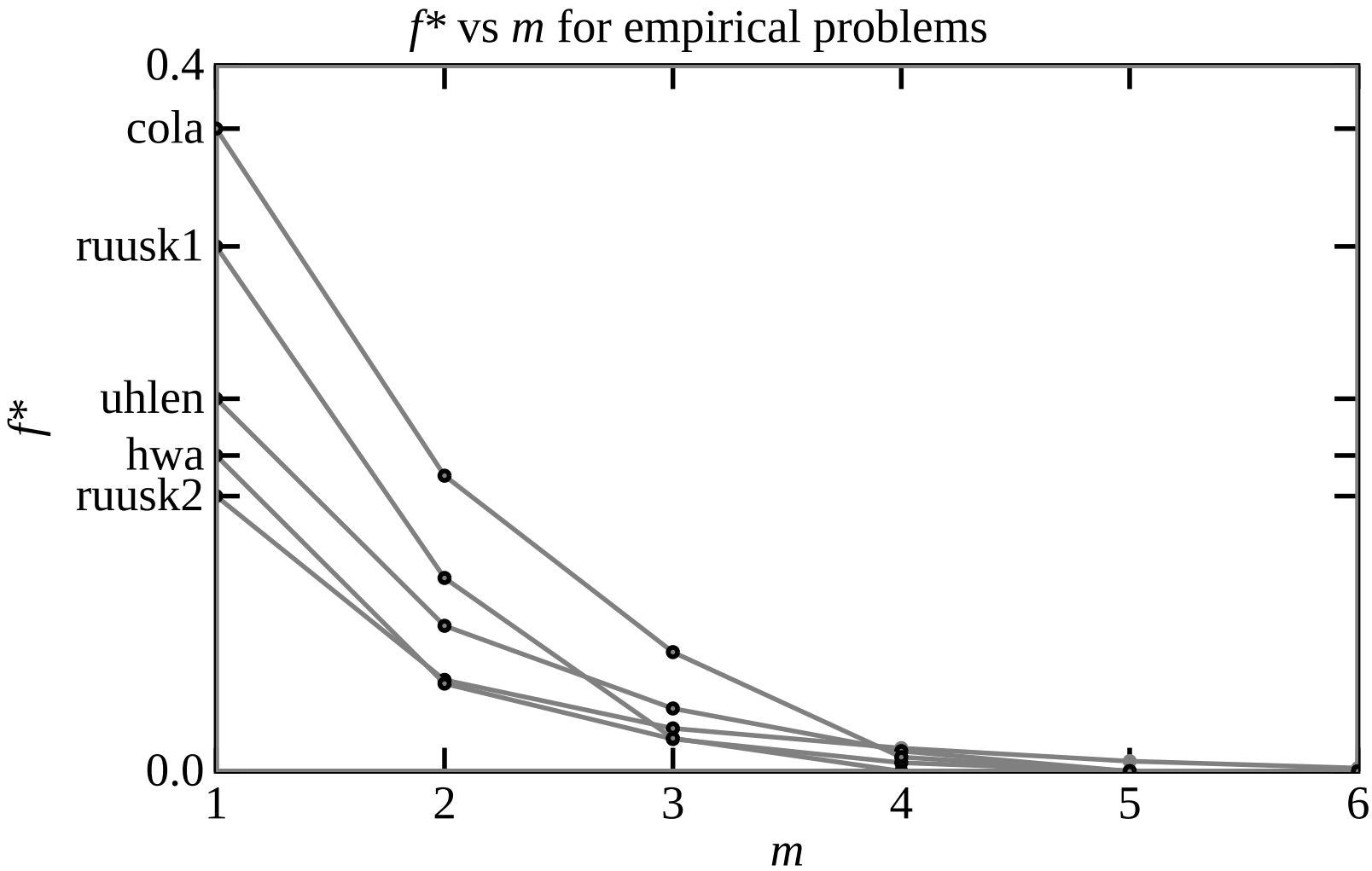
- Although *Stress* function seems rather simple, it normally has many local minima.
- The problem is high dimensional: $\mathbf{x} \in \mathbb{R}^N$ and the number of variables is equal to $N = n \times m$.
- Non-differentiability normally cannot be ignored. Minkowski distances:

$$d_r(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^r \right)^{1/r} .$$

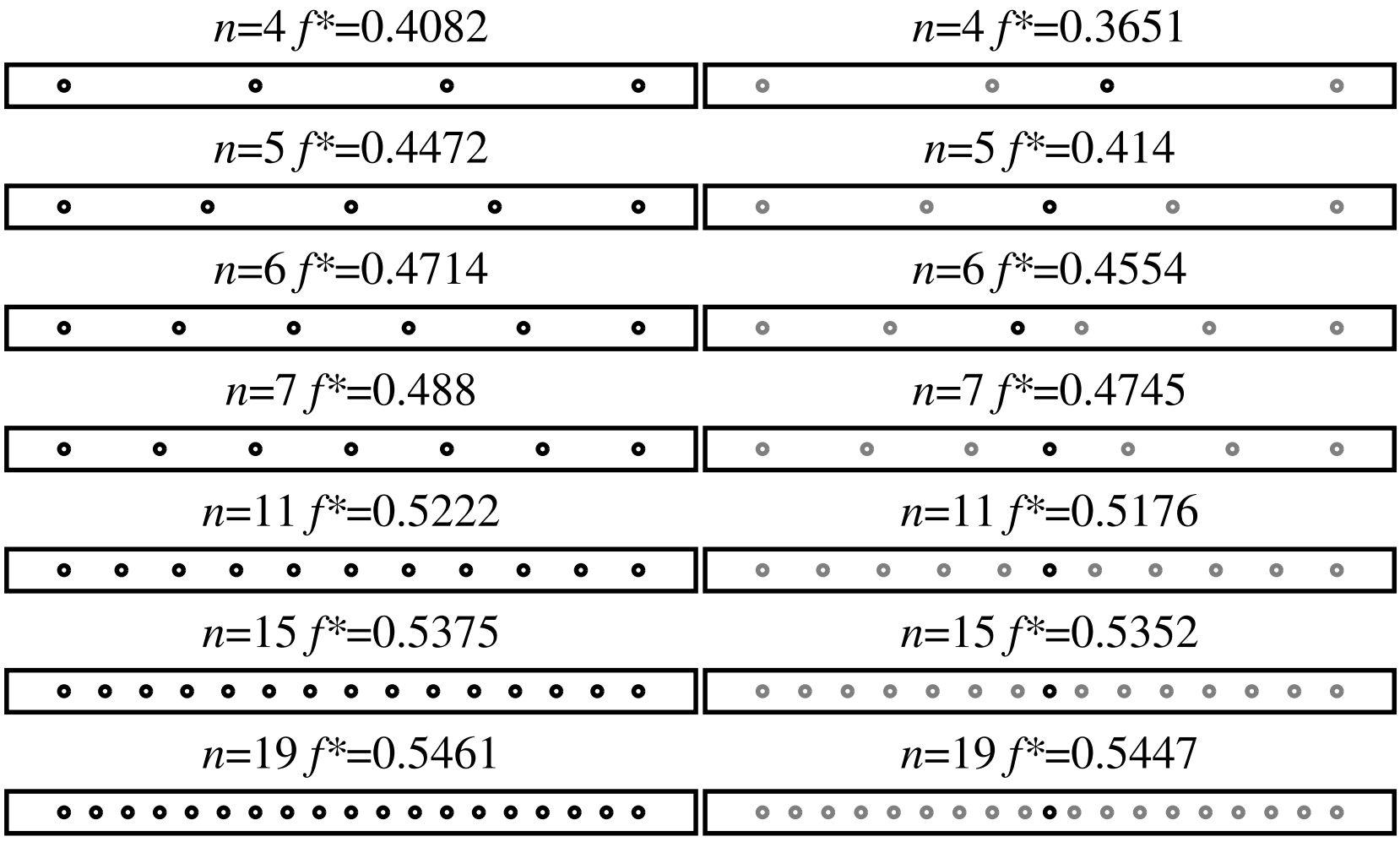
On dimensionality of embedding space



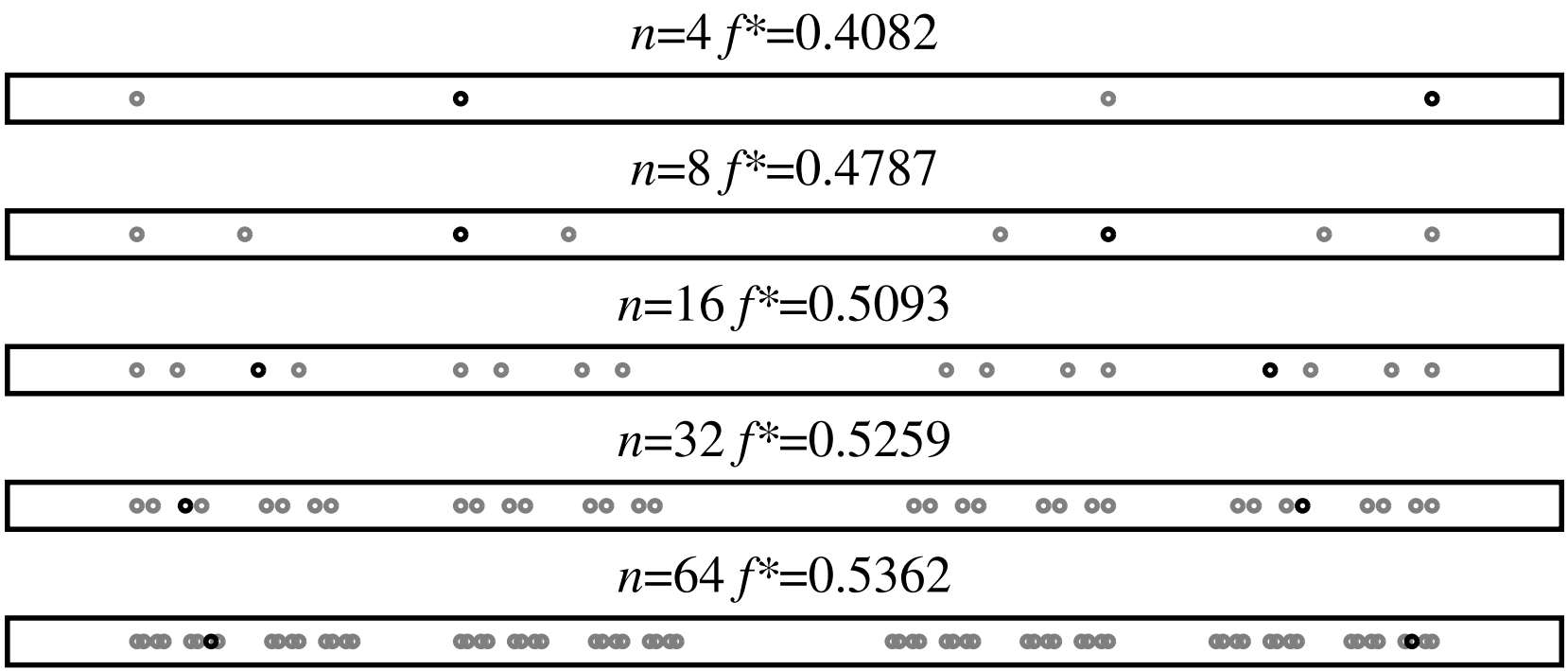
On dimensionality of embedding space



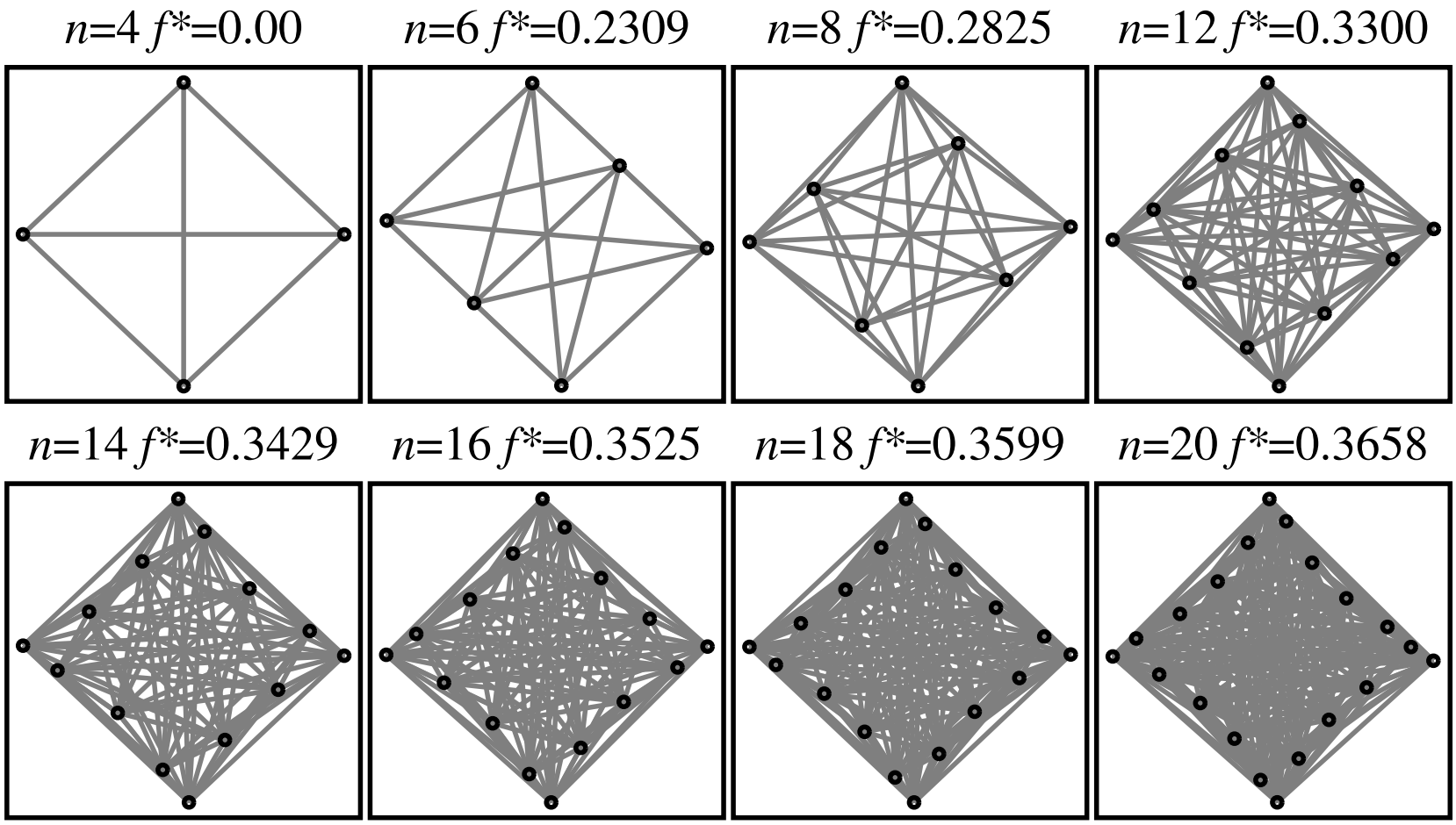
Images of simplices, $m = 1$



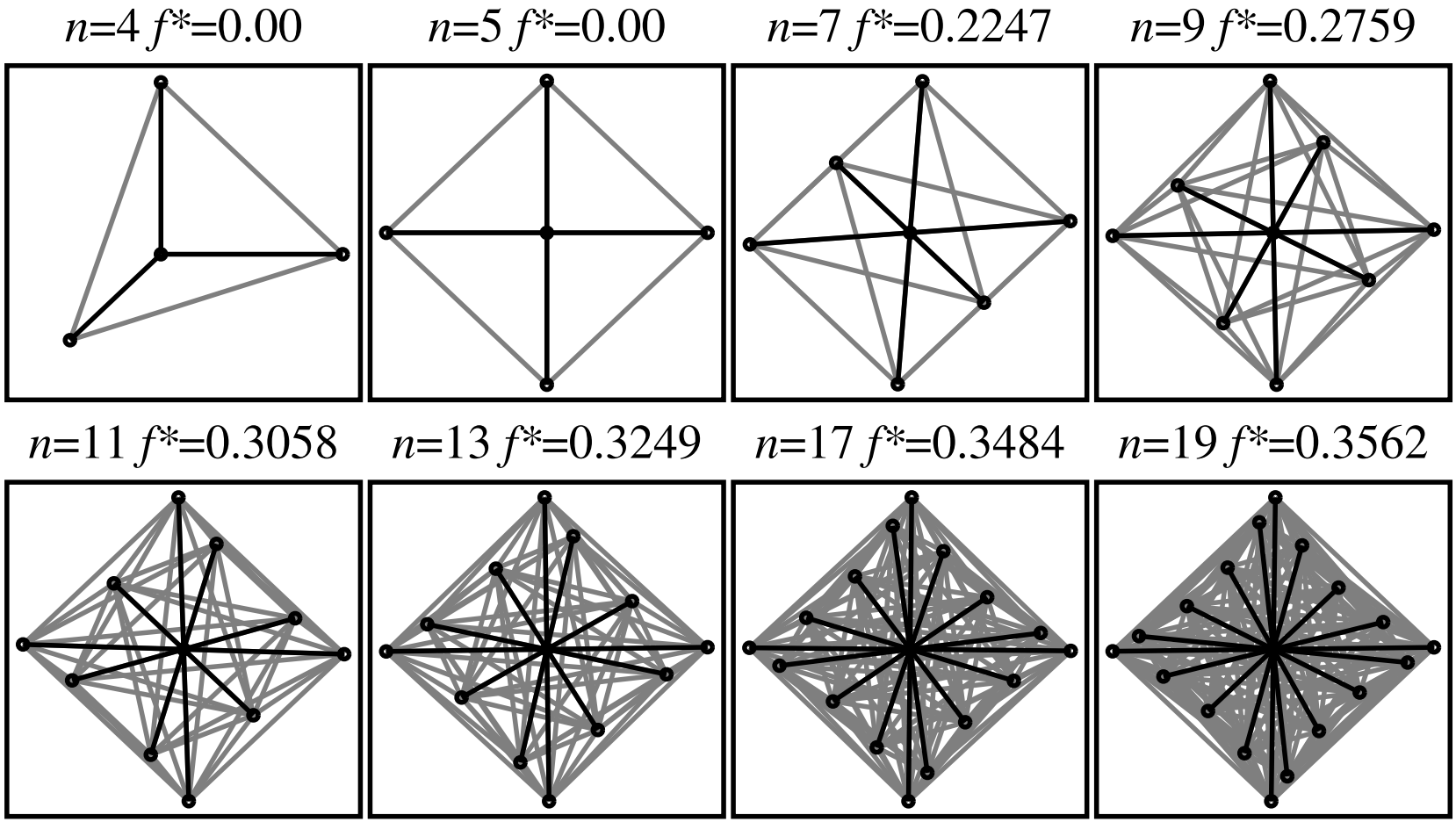
Images of cubes, $m = 1$



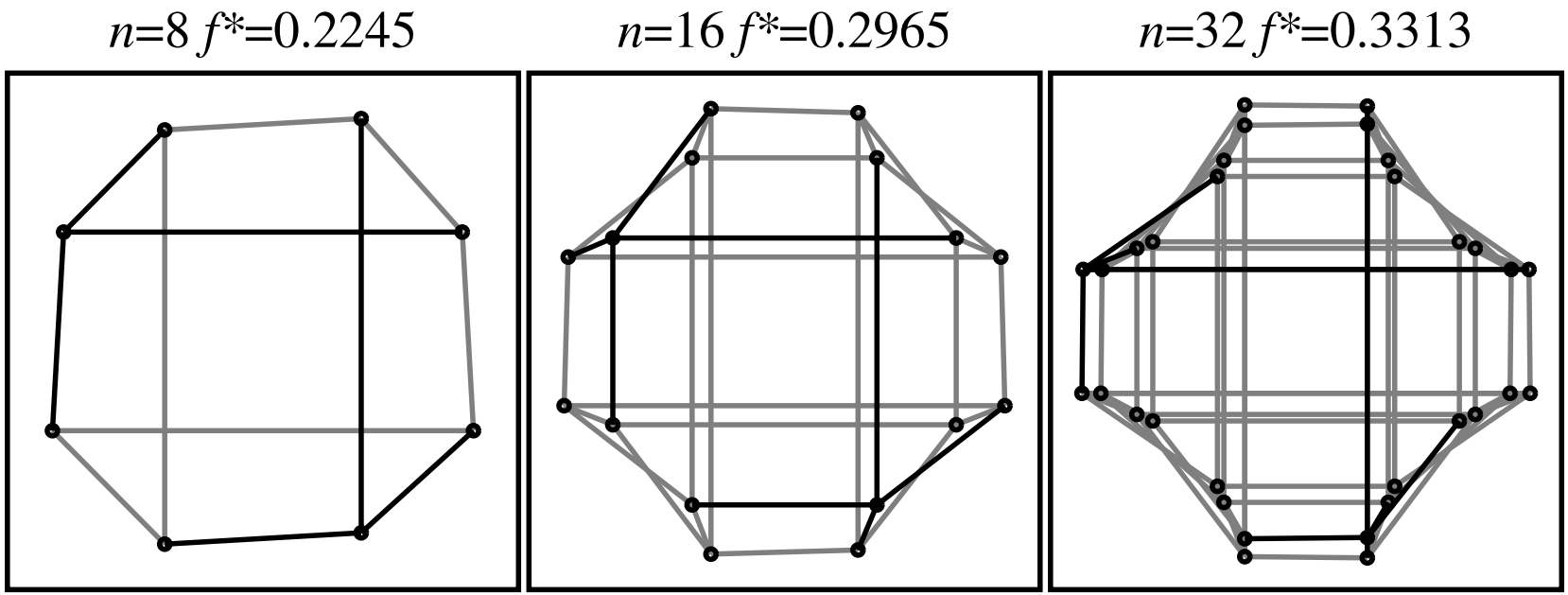
Images of standard simplices, $m = 2$



Images of unit simplices, $m = 2$

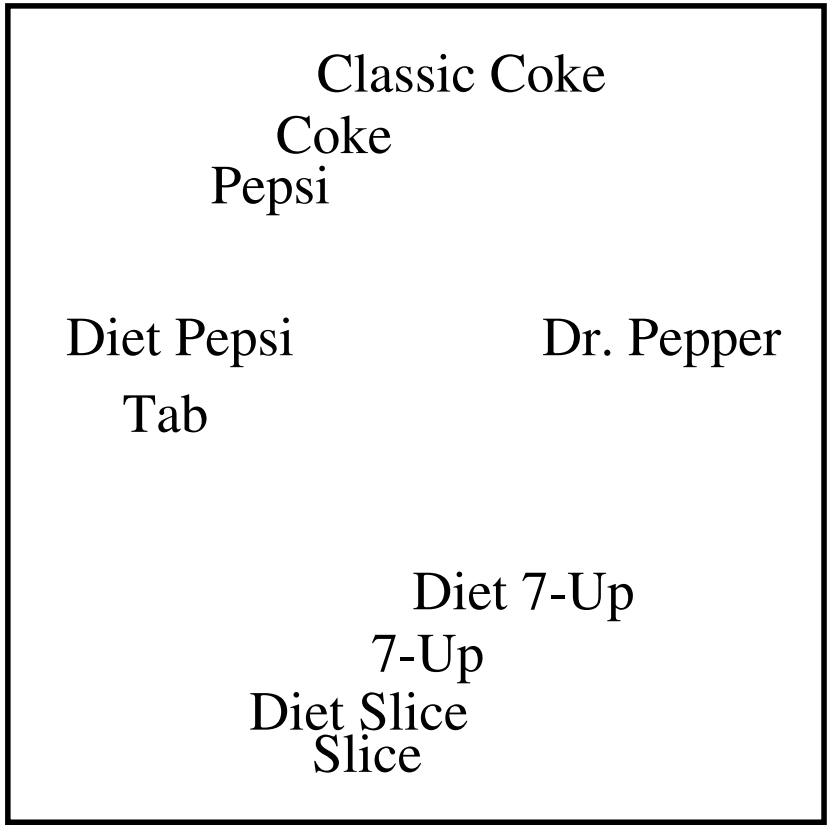


Images of cubes, $m = 2$

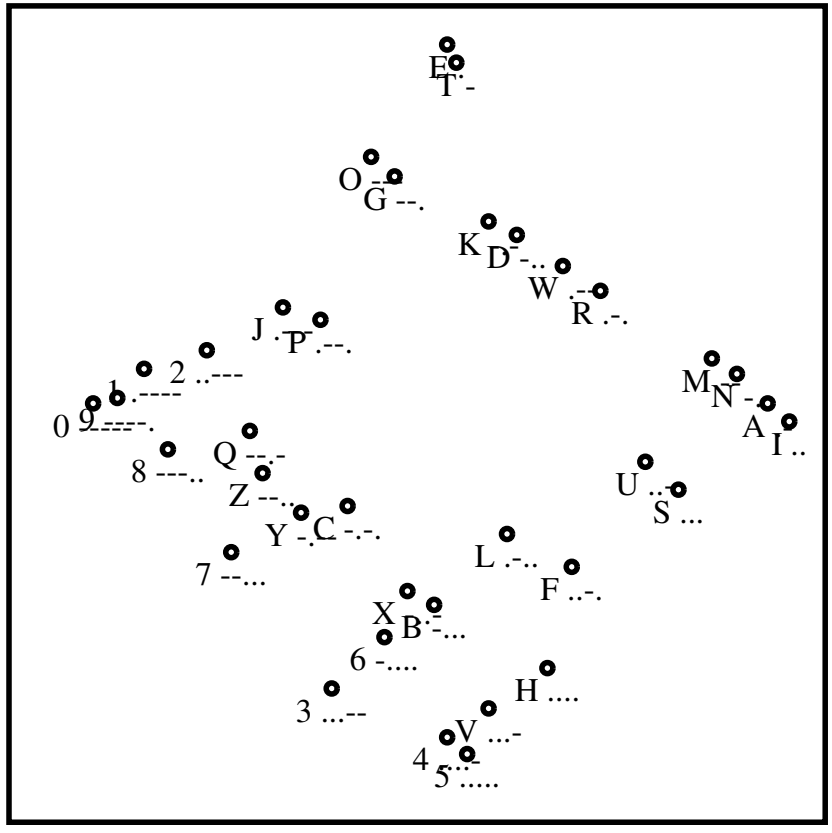


Images of empirical data, $m = 2$

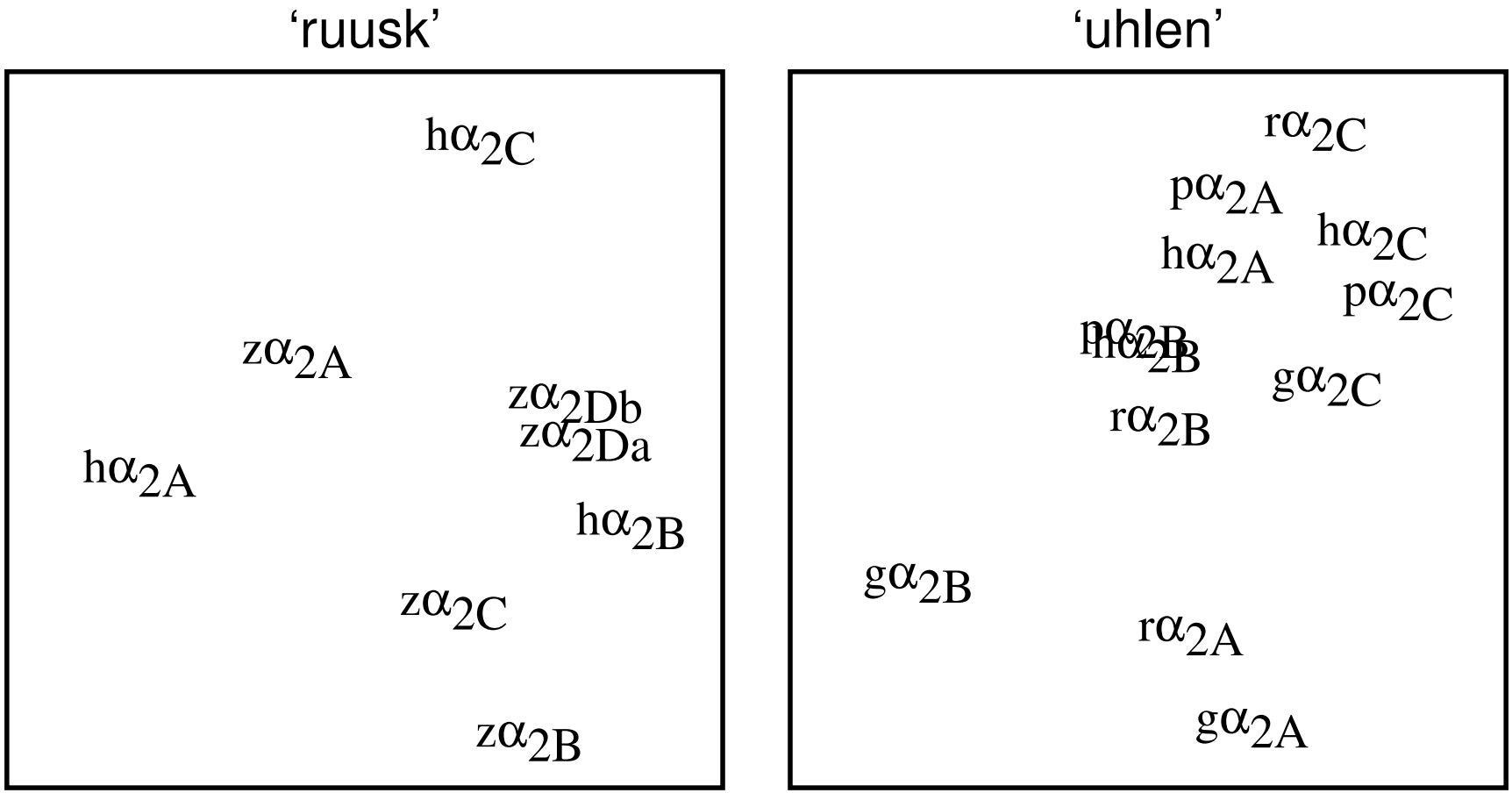
‘cola’



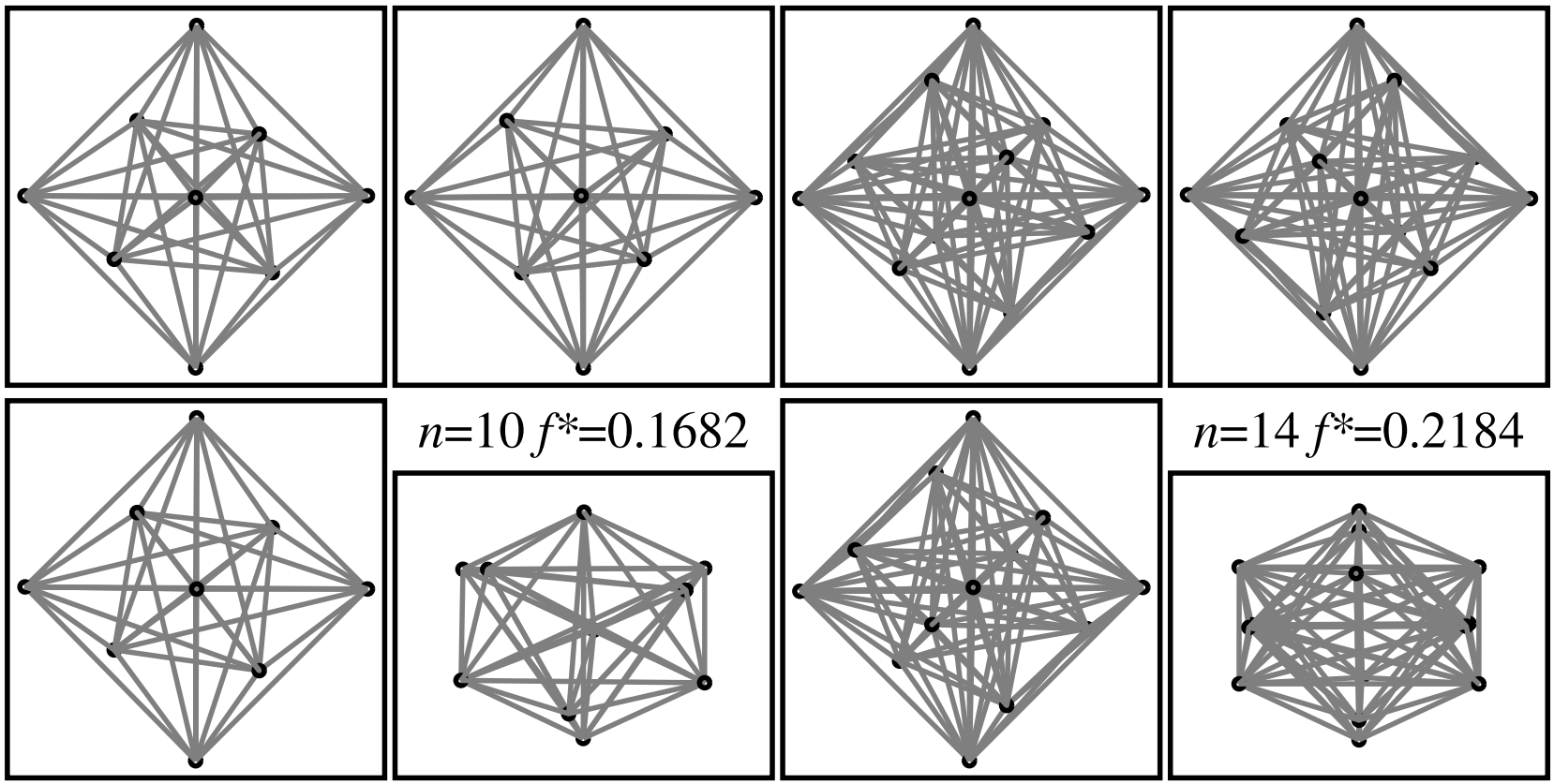
‘morseodes’



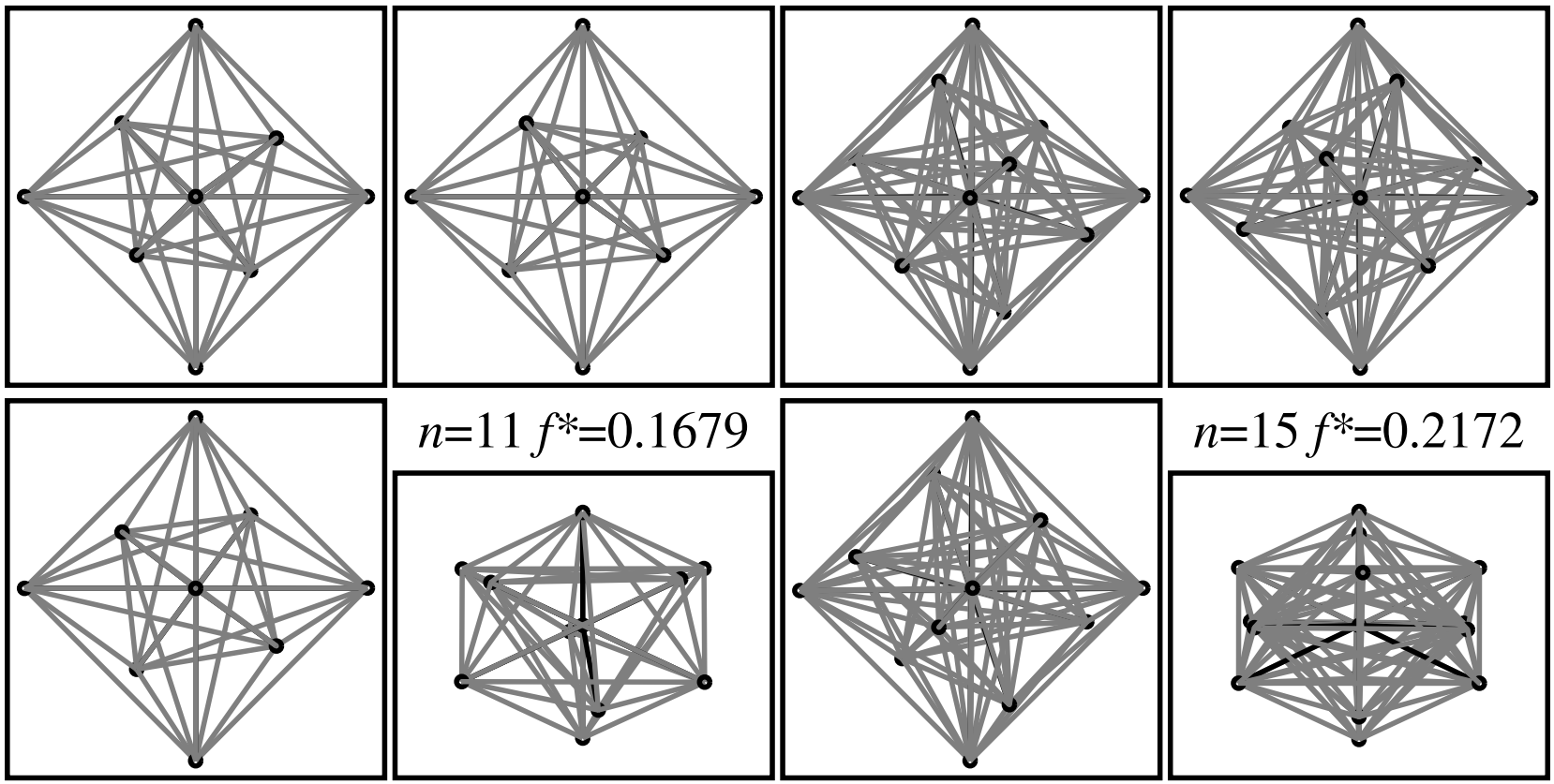
Images of pharmacological data, $m = 2$



Images of standard simplices, $m = 3$



Images of unit simplices, $m = 3$



Images of cubes, $m = 3$

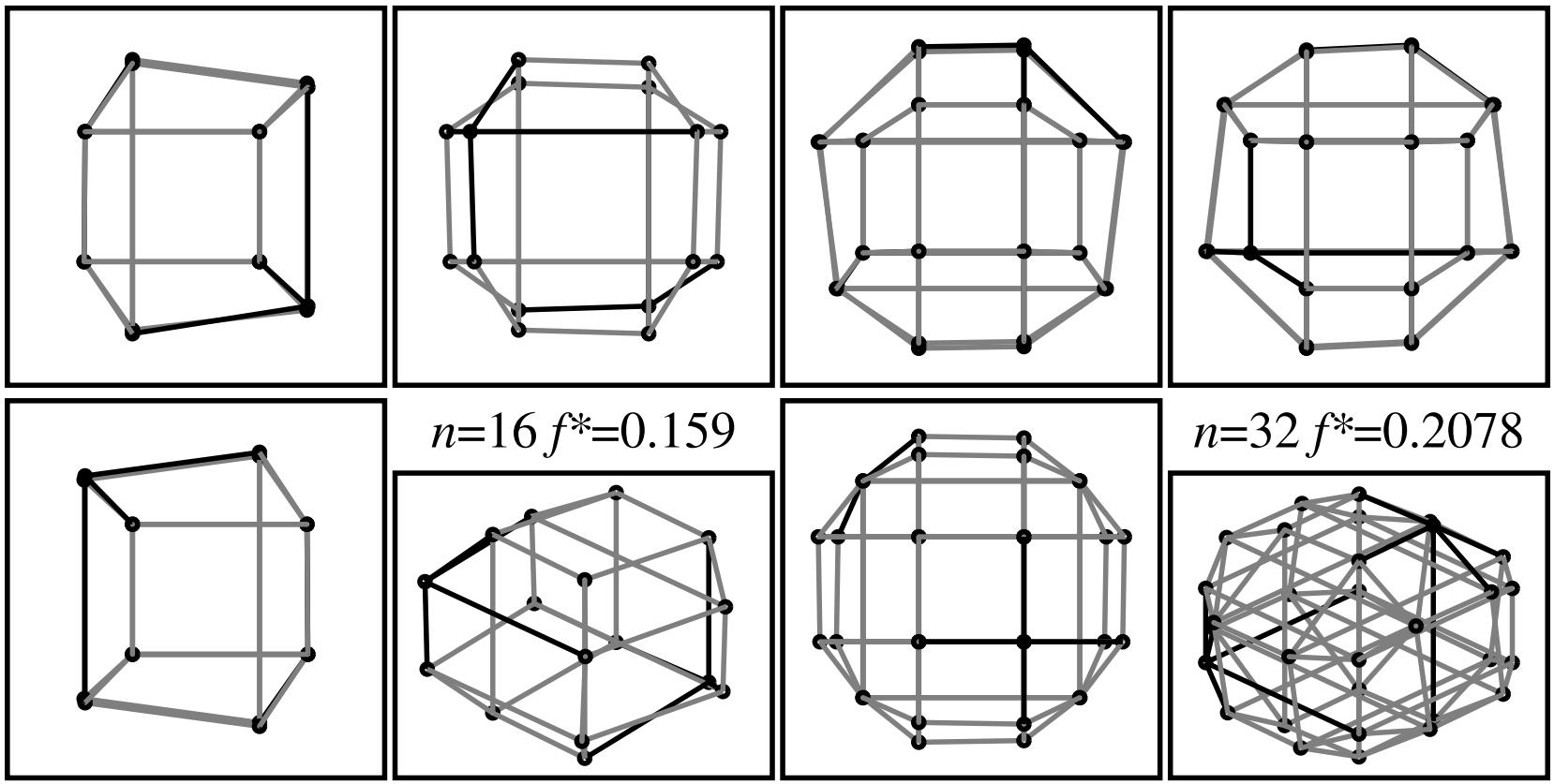


Image of the properties of human and zebrafish α_2-adrenoceptors, $m = 3$

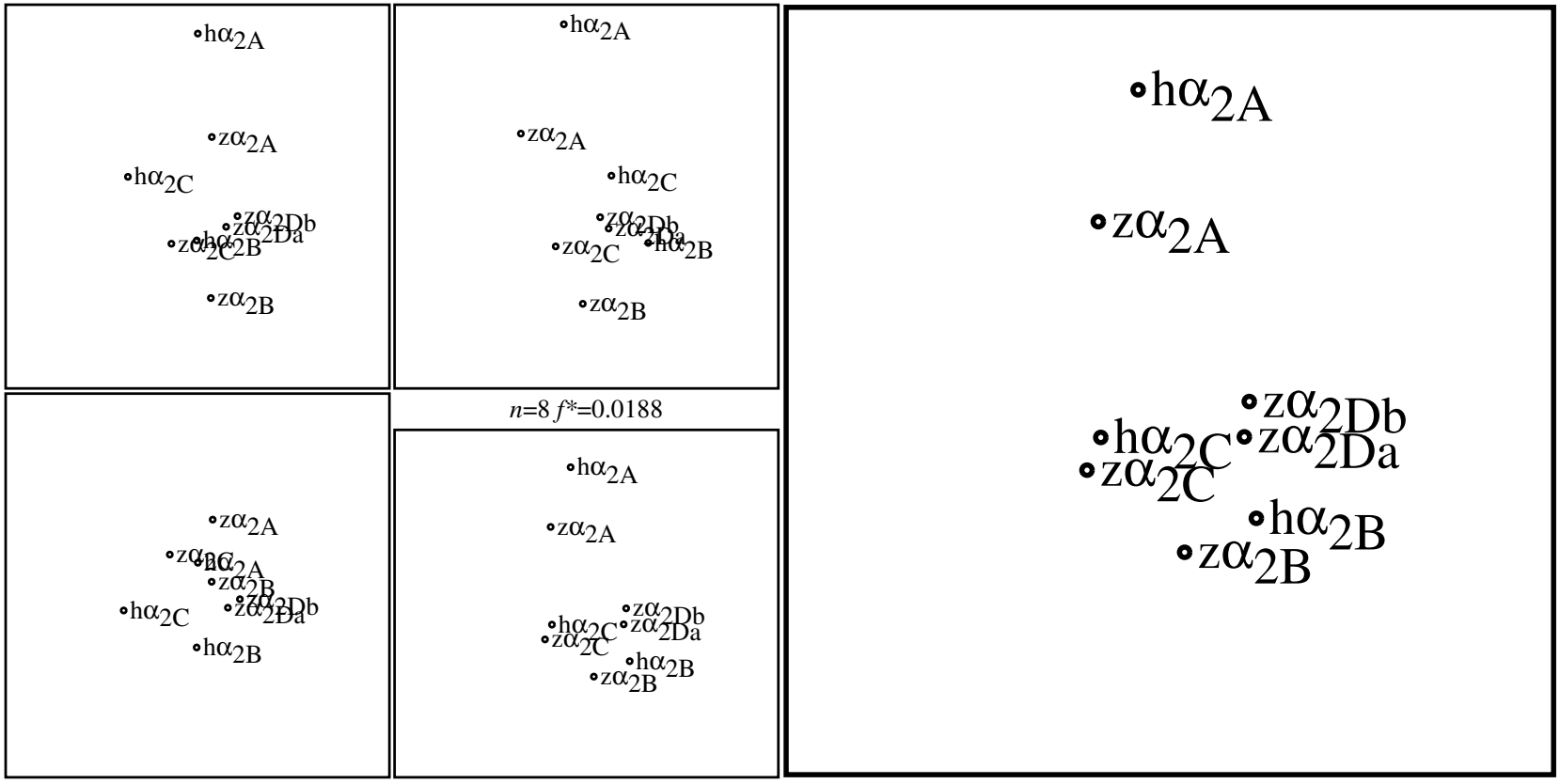


Image of the properties of 20 ligands binding human and zebrafish α_2-adrenoceptors, $m = 3$

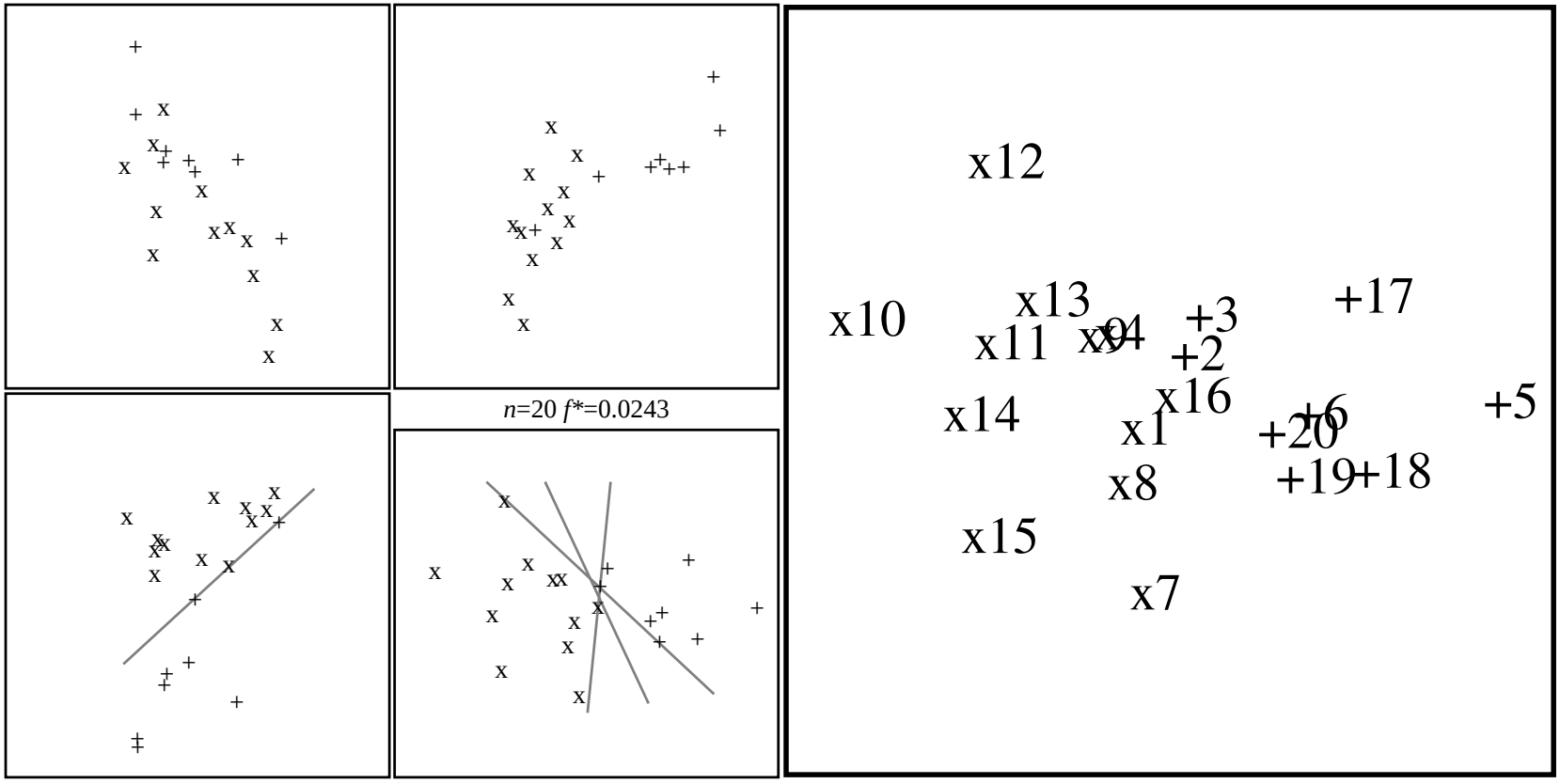


Image of the properties of human, rat, guinea pig and pig α_2-adrenoceptors, $m = 3$

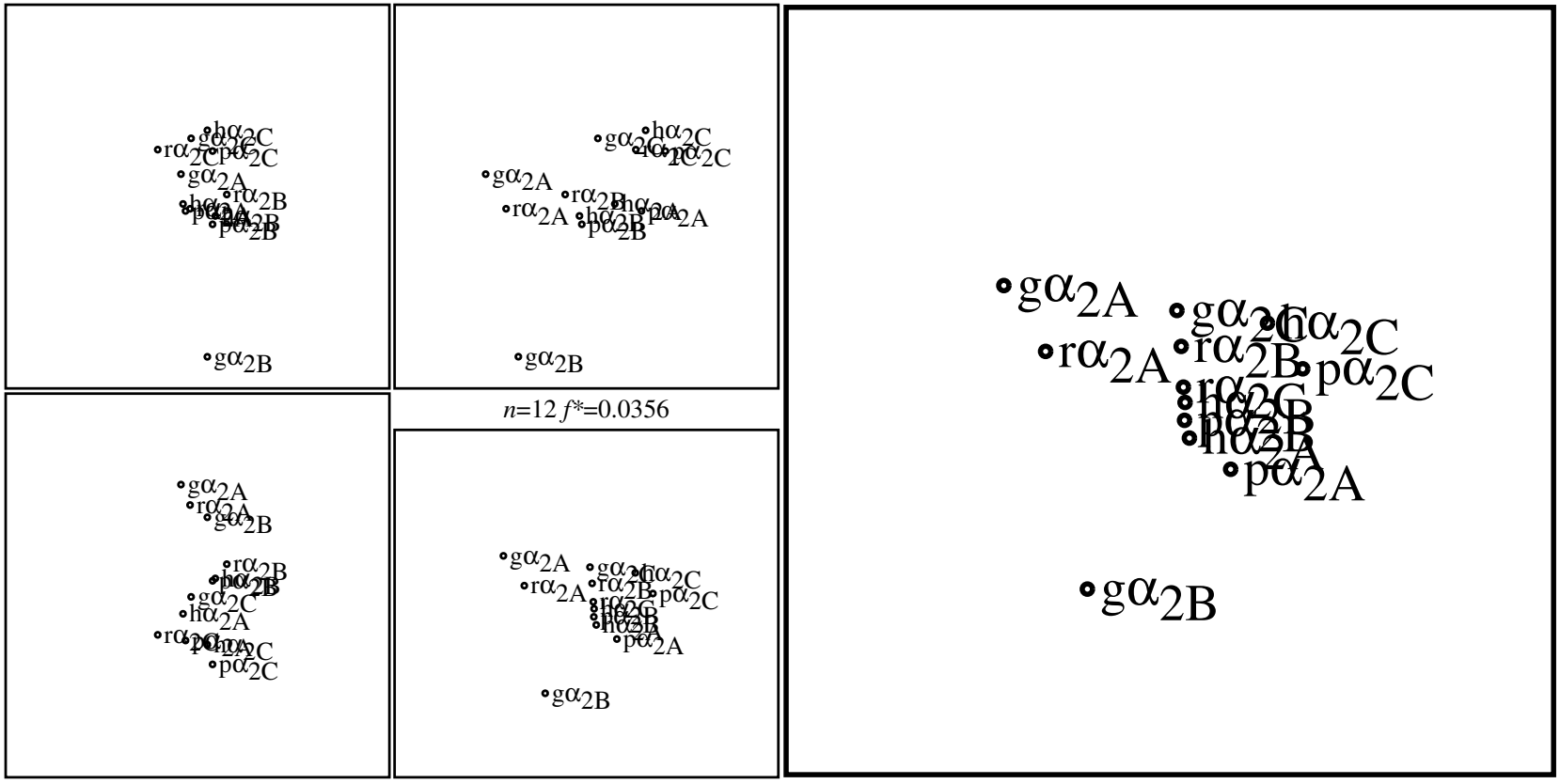
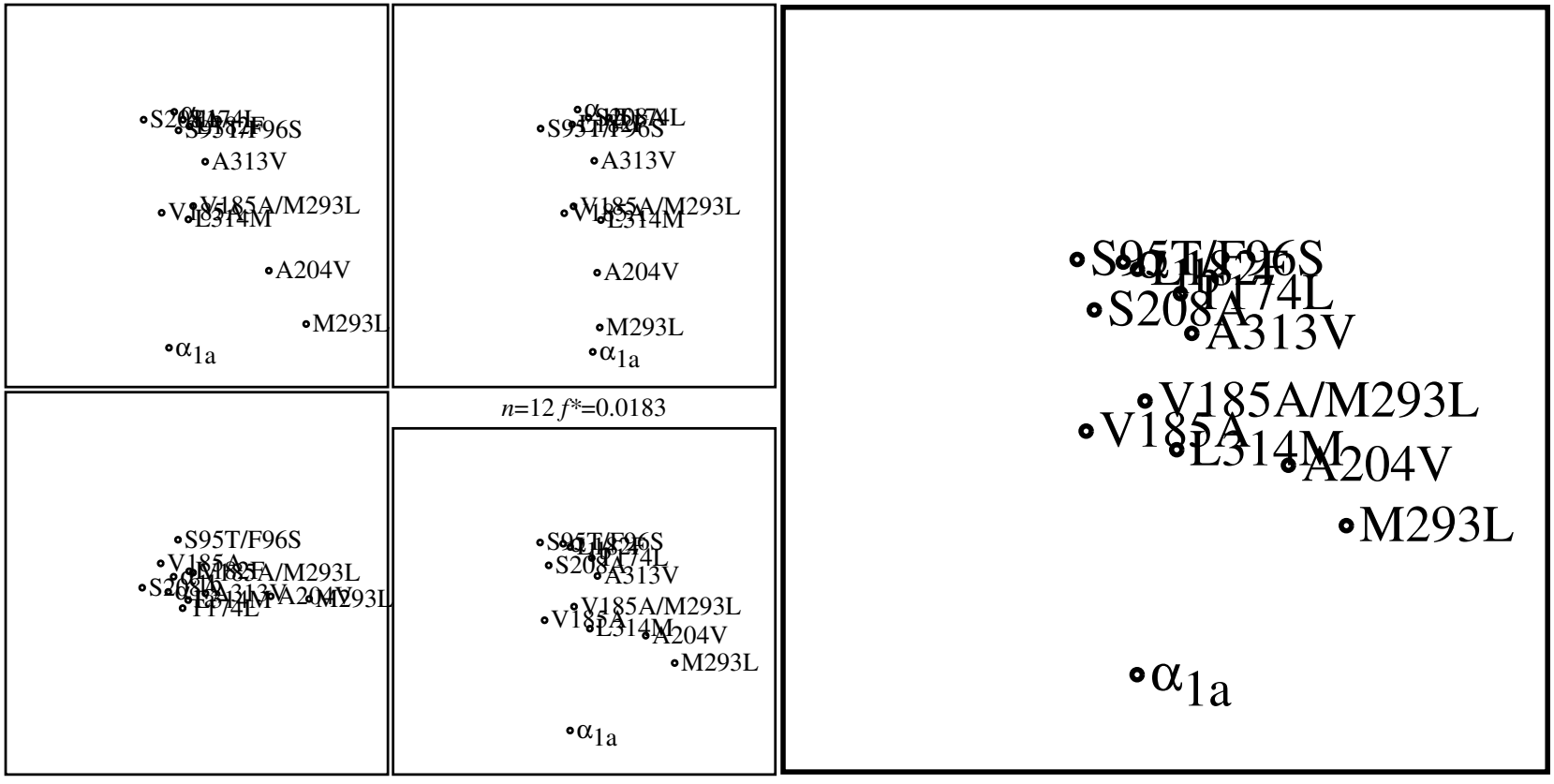


Image of the properties of wild type and mutant α_1-adrenoceptors, $m = 3$



Multidimensional Data Visualization

Overview

Strategies for Multidimensional Data Visualization

- Direct visualization methods, where each feature, characterizing a multidimensional object, is represented in a visual form;
- Projection, so-called dimensionality reduction, methods, allowing us to represent the multidimensional data on a low-dimensional space. Artificial neural networks may also be used for visualizing multidimensional data. They realize various nonlinear projections.

Direct Visualization

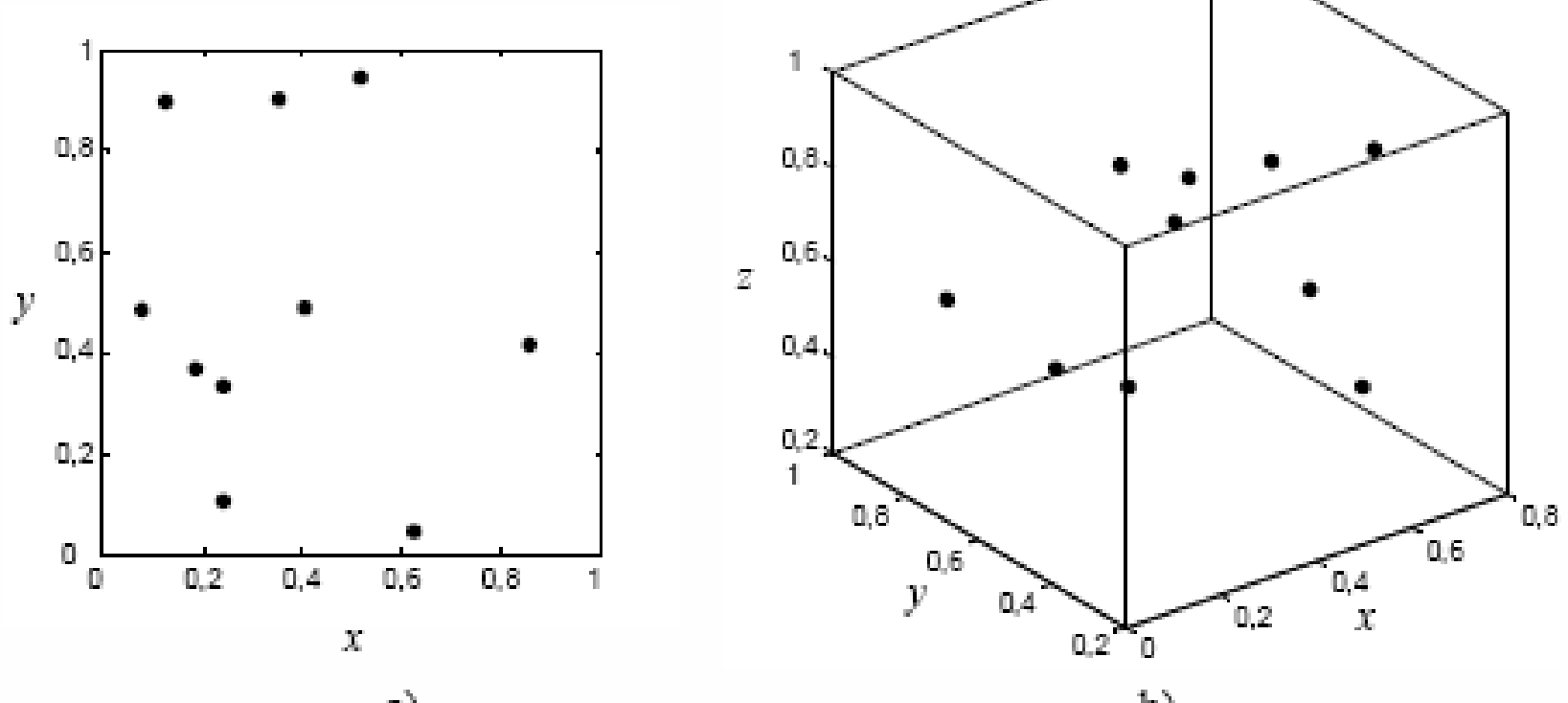
- The direct data visualization is a graphical presentation of the data set that provides a qualitative understanding of the information contents in a natural and direct way.
- The commonly used methods are scatter plot matrices, parallel coordinates, Andrews curves, Chernoff faces, stars, dimensional stacking, etc.
- The direct visualization methods do not have any defined formal mathematical criterion for estimating the visualization quality.
- Each of the features $x_1, x_2, \ldots, x_n$ characterizing the object $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i \in \{1, \ldots, m\}$, is represented in a visual form acceptable for a human being.

Direct Visualization Methods

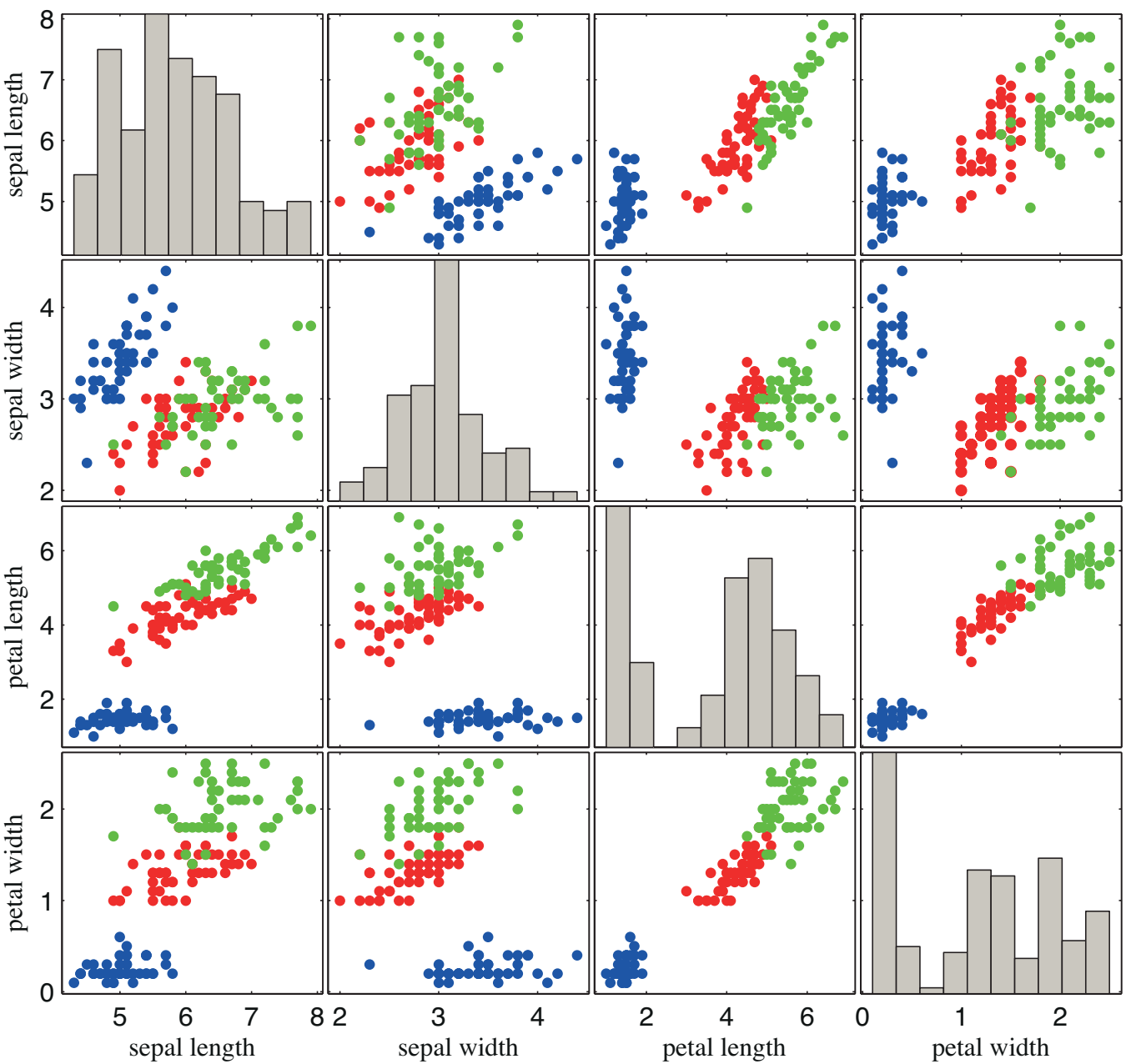
1. Geometric methods:
   a) scatter plots,
   b) matrix of scatter plots,
   c) multiline graphs,
   d) permutation matrix,
   e) survey plots,
   f) Andrews curves,
   g) parallel coordinates,
   h) radial visualization (RadViz) and its modifications GridViz and PolyViz.
2. Iconographic displays:
   a) Chernoff faces,
   b) star glyphs,
   c) stick figure,
   d) color icon.
3. Hierarchical displays:
   a) dimensional stacking,
   b) trellis display,
   c) hierarchical parallel coordinates.

Geometric Methods

- Geometric visualization methods are the methods where multidimensional points are displayed using the axes of the selected geometric shape.
- *Scatter plots* are one of the most commonly used techniques for data representation on a plane $\mathbb{R}^2$ or space $\mathbb{R}^3$. Points are displayed in the classic (x, y) or (x, y, z) format.



Scatter Plot Matrix of the Iris Data



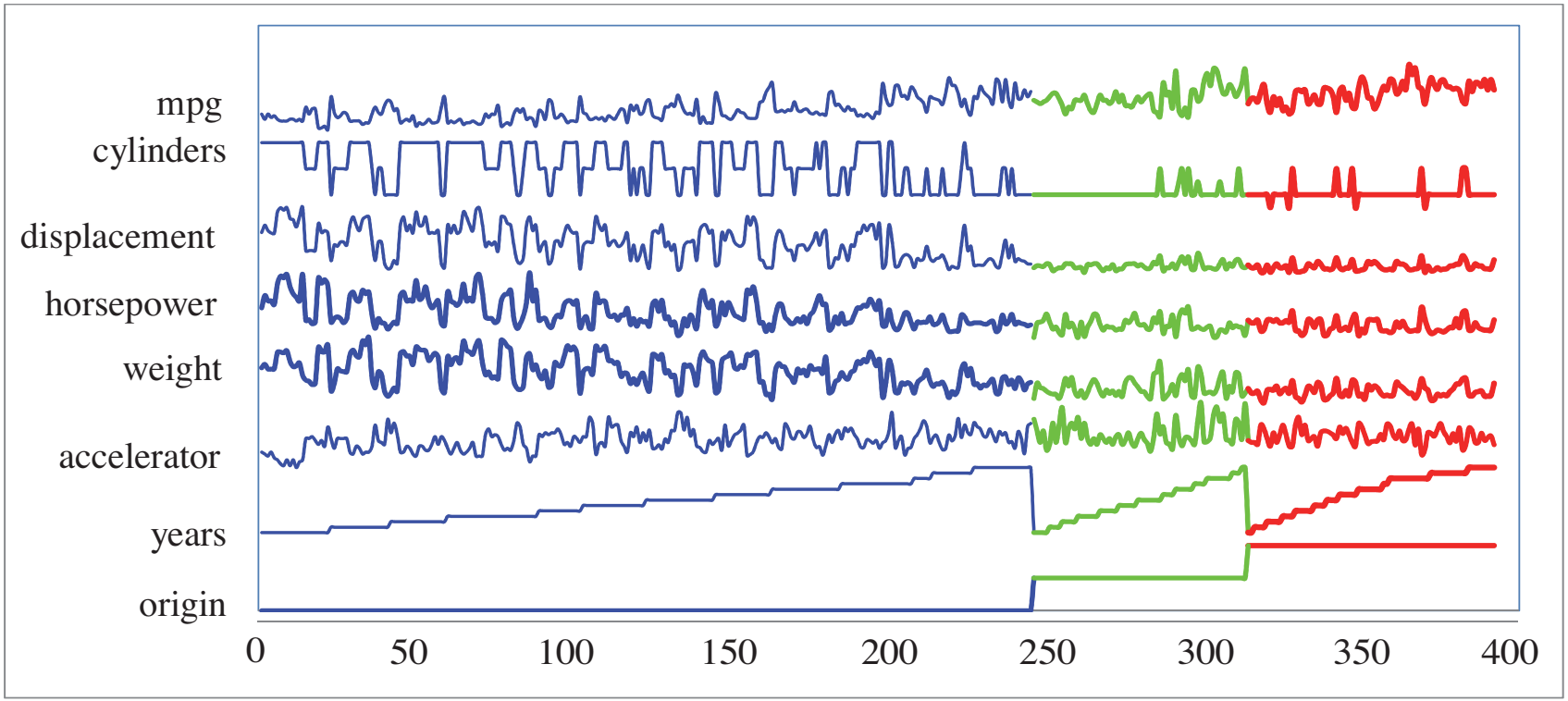
Setosa

Versicolor

Virginica

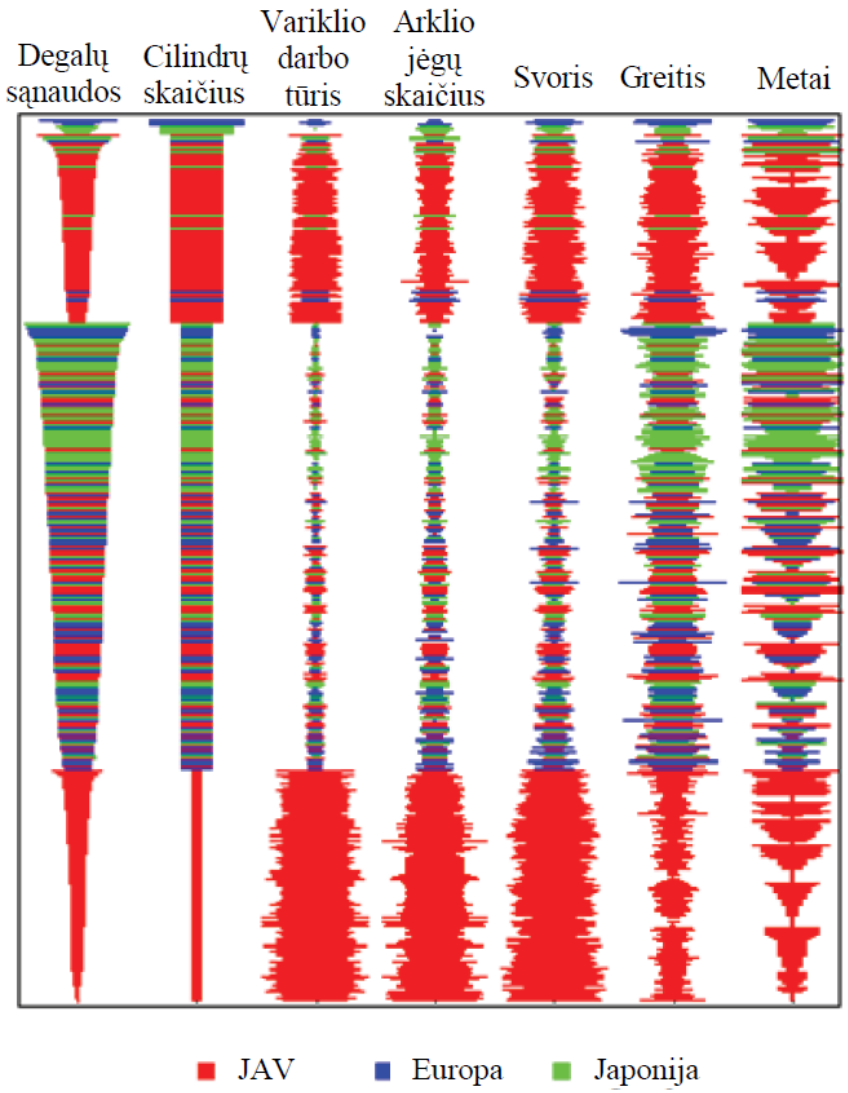
Multiline Graphs of the Auto MPG Data

- The data are aligned according to origin (USA, Japan, Europe).

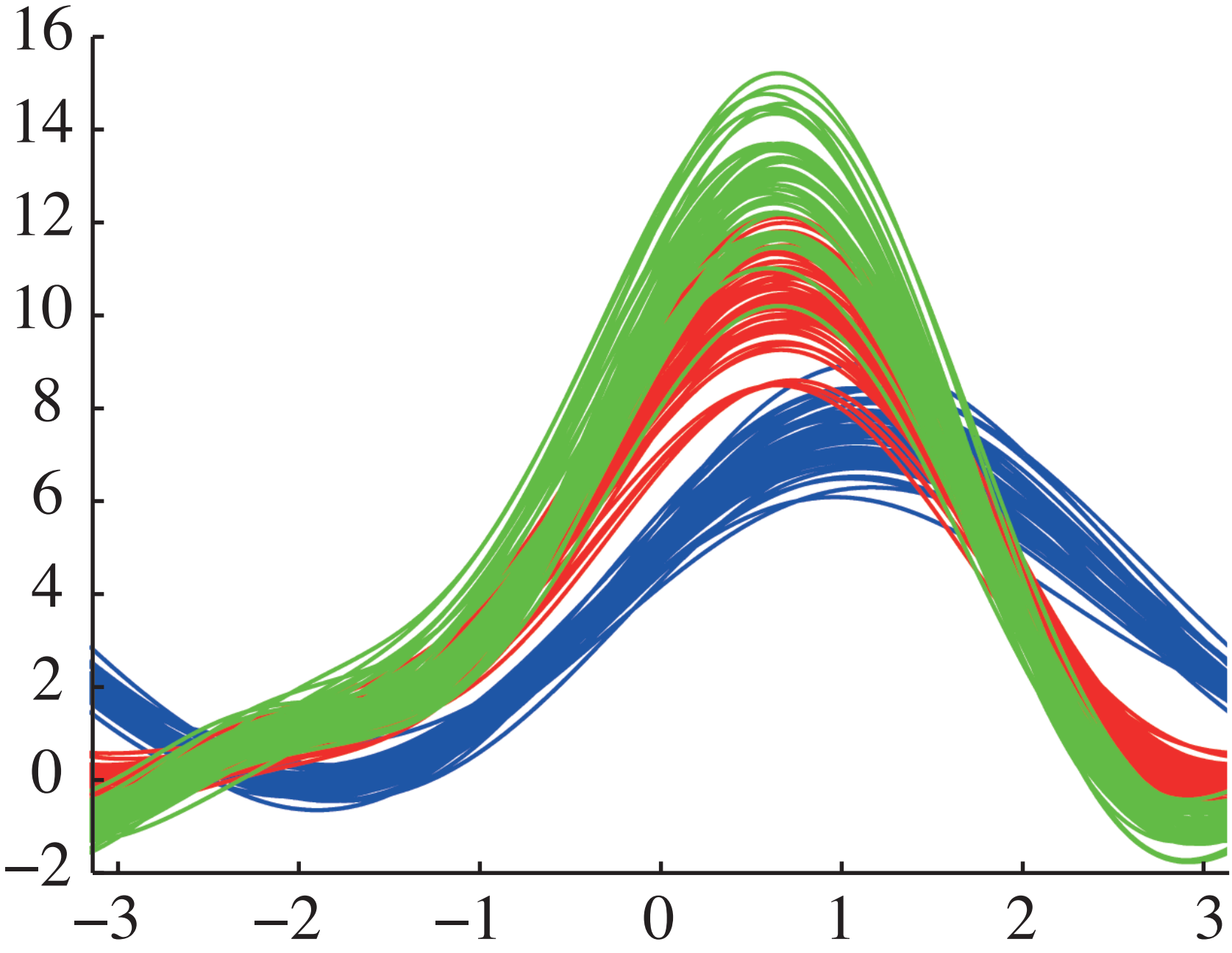


Survey Plot of Auto MPG Data

- Sorted according to the number of cylinders and MPG.

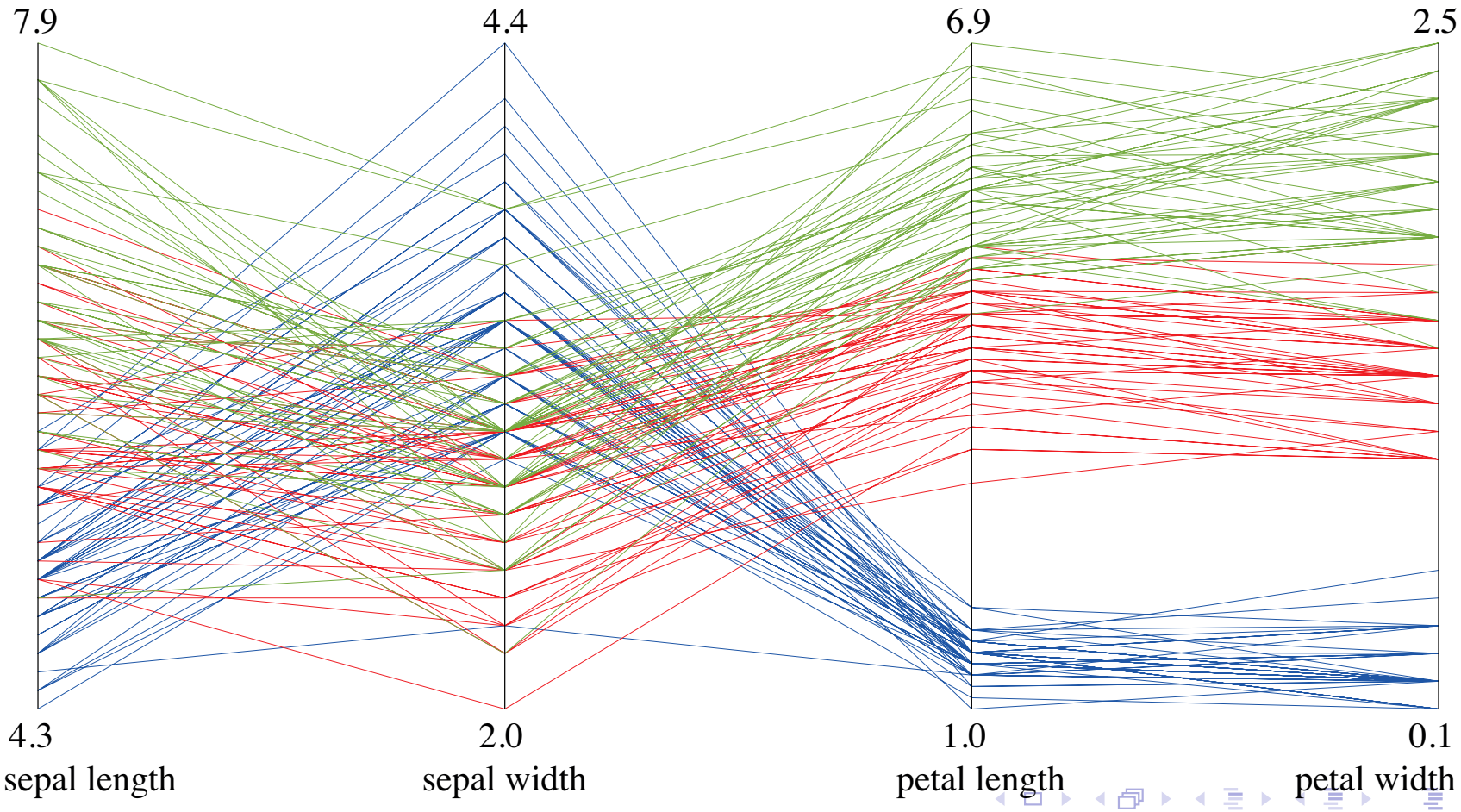


Andrews Curves of Iris Data



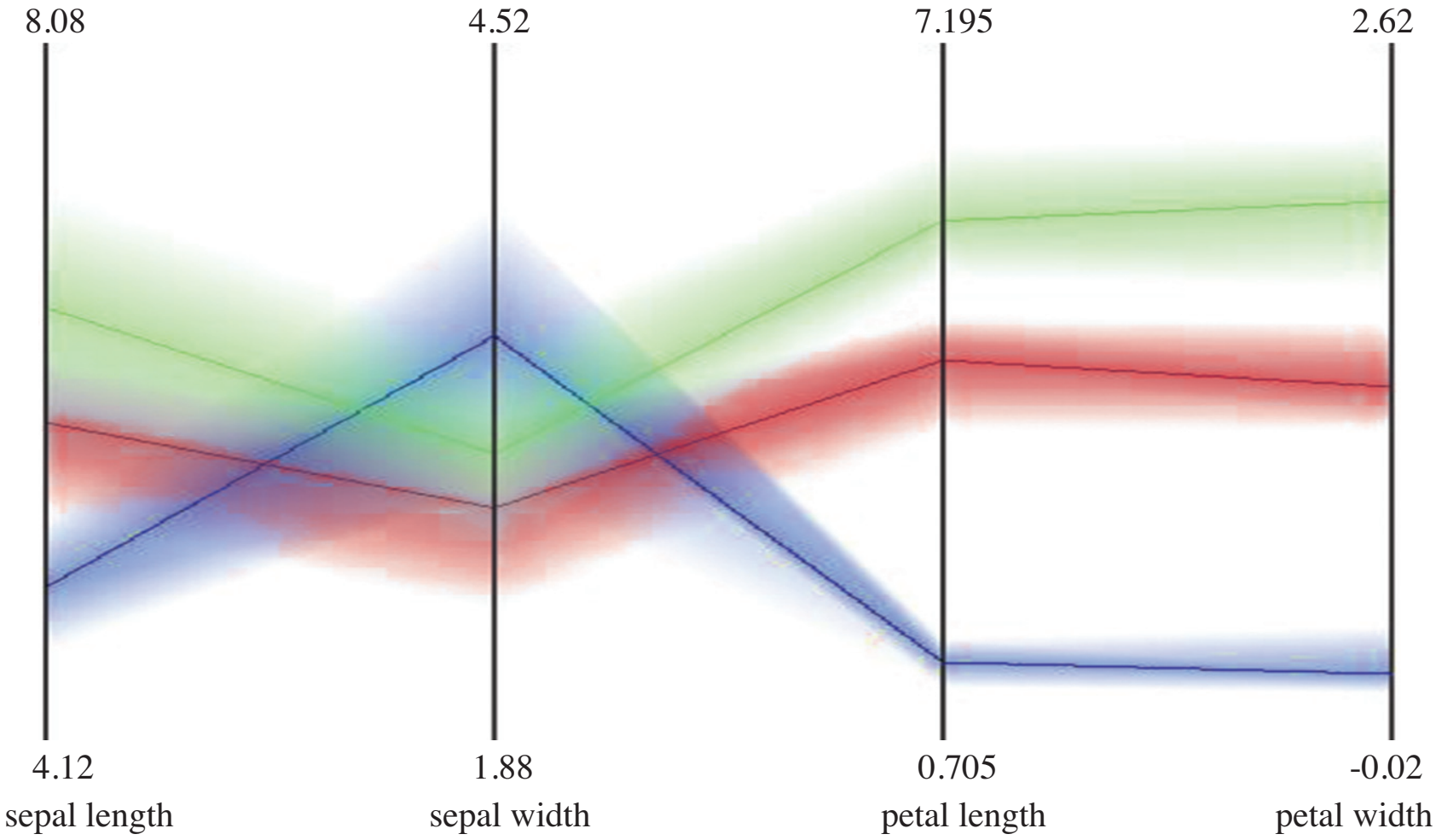
Iris Data Represented on the Parallel Coordinates

- The image is obtained using the system *Orange*.
- Different colors correspond to the different species.
- We see that the species are distinguished best by the petal length and width. It is difficult to separate the species by the sepal length and width.



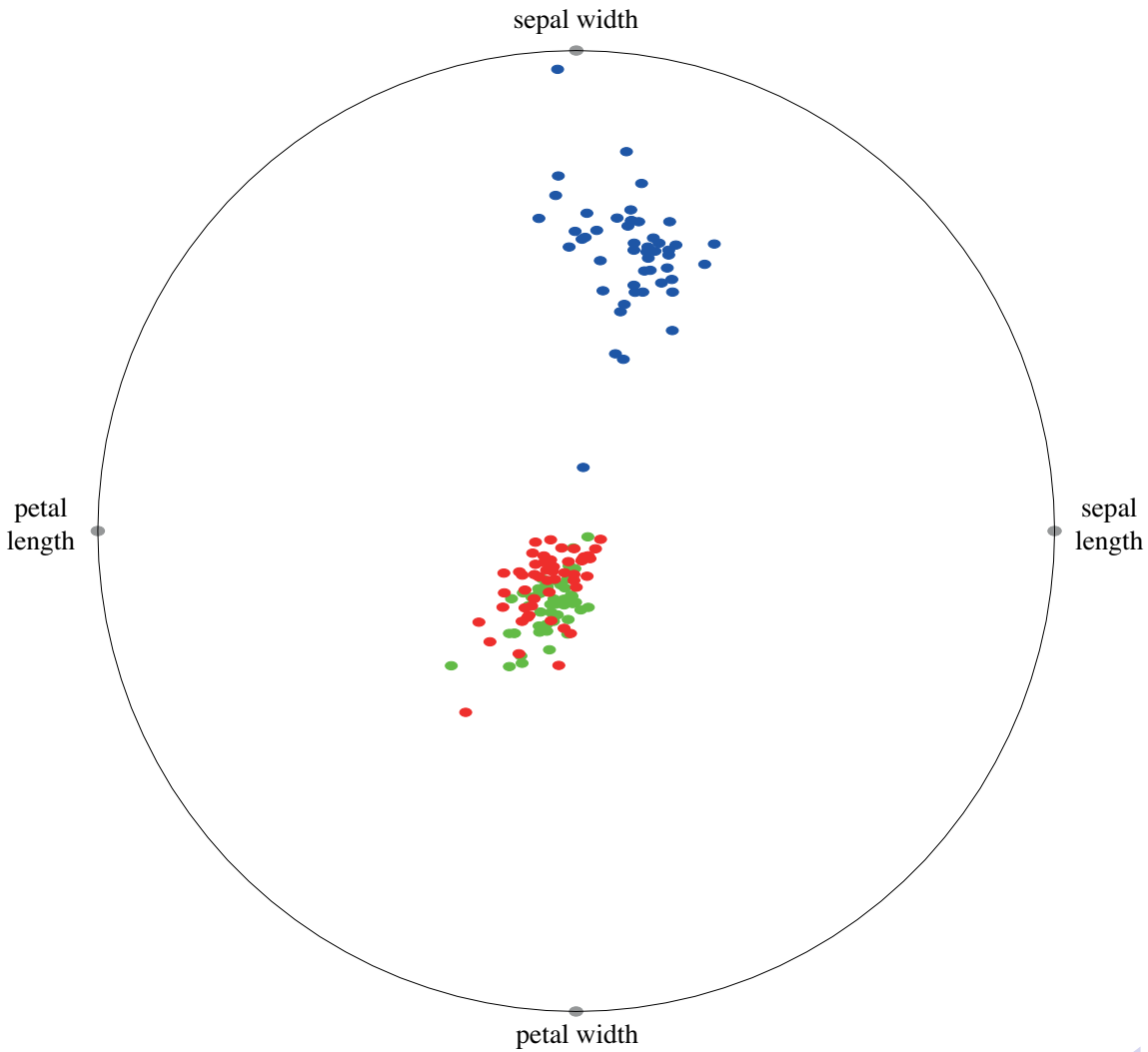
Iris Data Represented on the Hierarchical Parallel Coordinates

- The image is obtained using the system *Xmdv*.

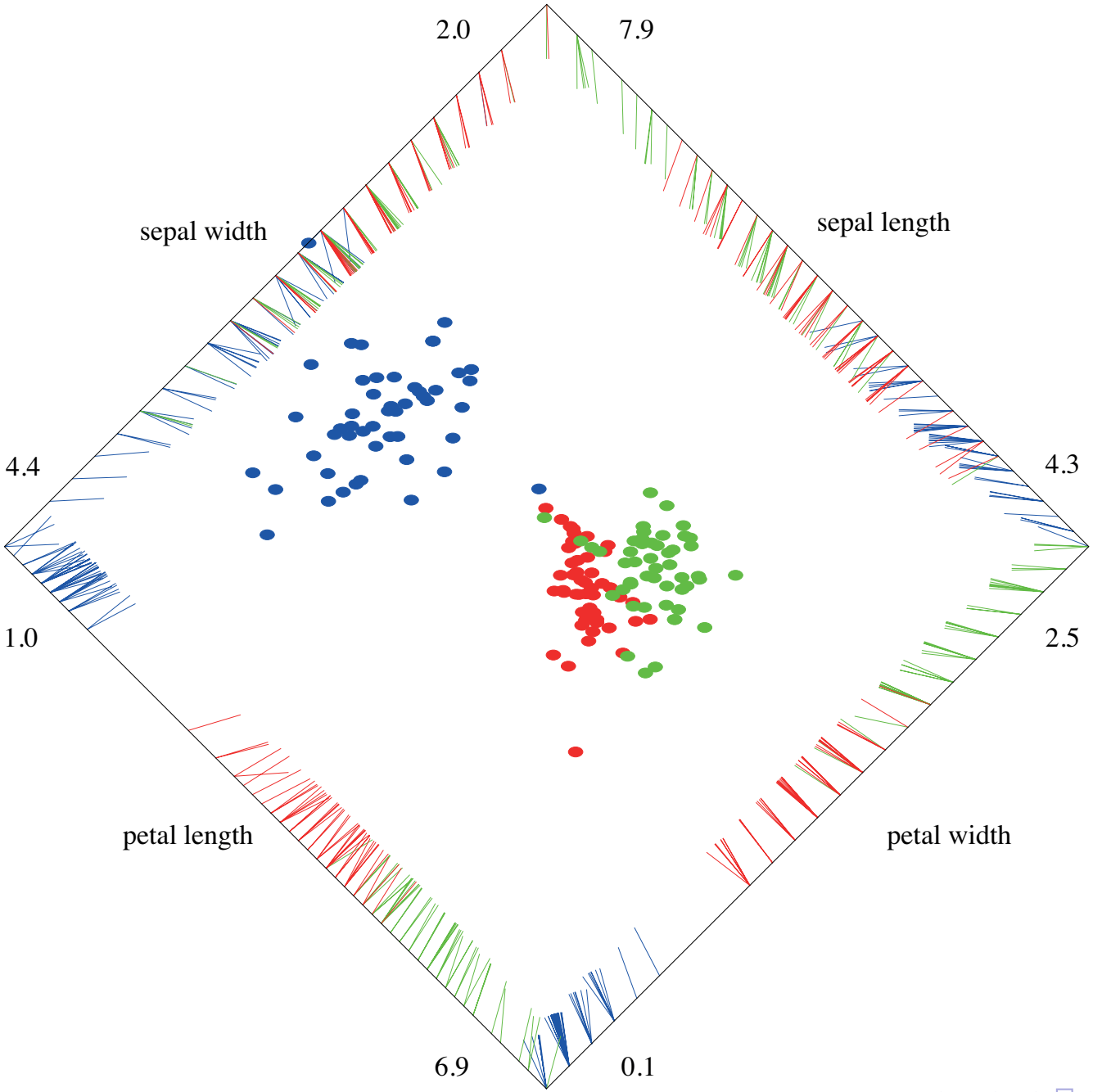


Iris data visualized by the RadViz method

- The petal length, petal width, sepal length, and sepal width are dimensional anchors. The image is obtained using the system *Orange*.



Iris Data Visualized by the PolyViz Method

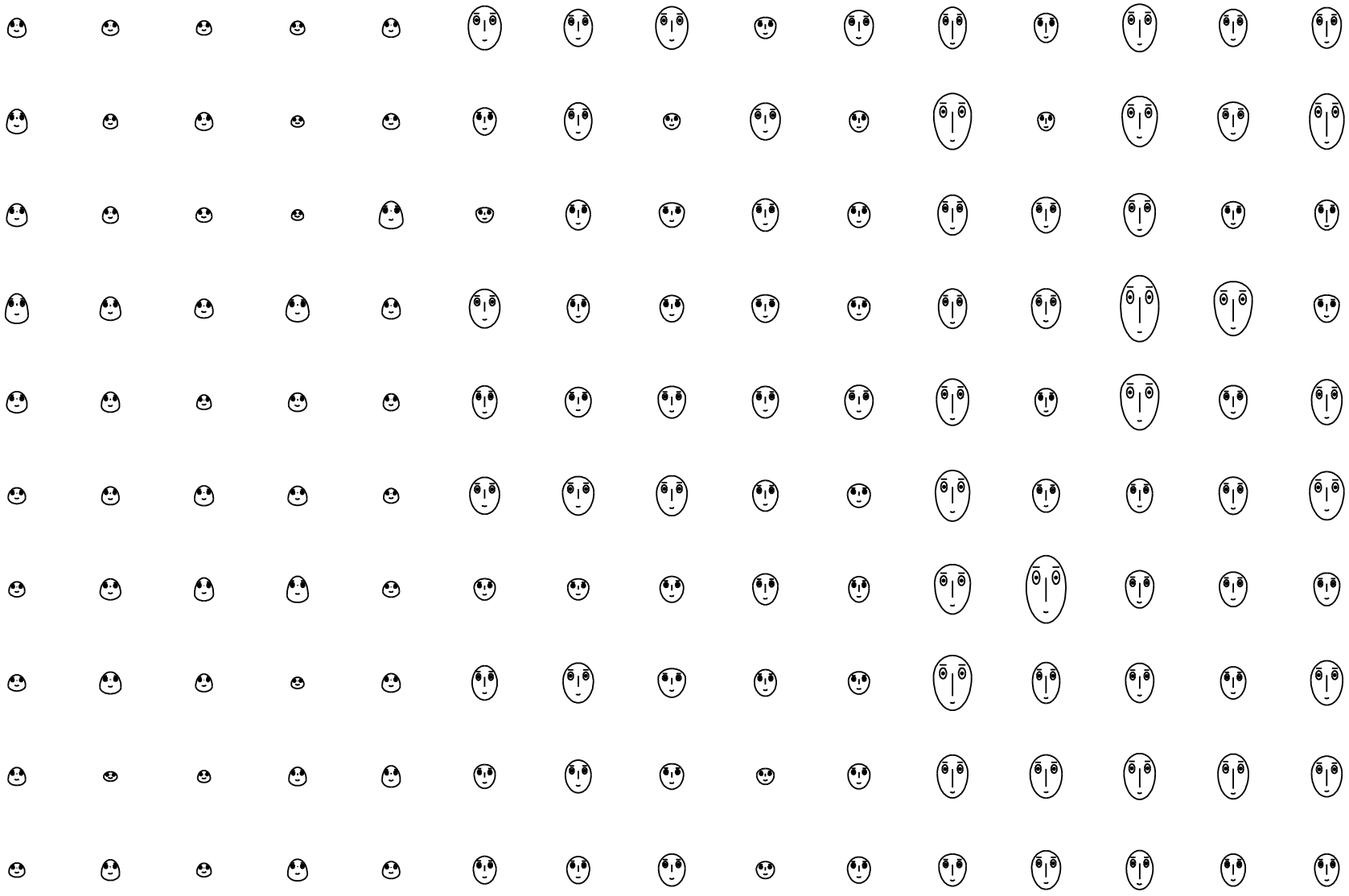


Iconographic Displays

- The aim of visualization of multidimensional data is not only to map the data onto a two- or three-dimensional space, but also to help perceiving them.
- The second aim may be achieved visualizing multidimensional data by *iconographic display* methods. They are also called *glyph* methods.
- Each object that is defined by the n features is displayed by a glyph. Color, shape, and location of the glyph depend on the values of features.
- The most famous methods are *Chernoff faces* and the *star* method, however some methods of more complicated other glyphs may be used as well.

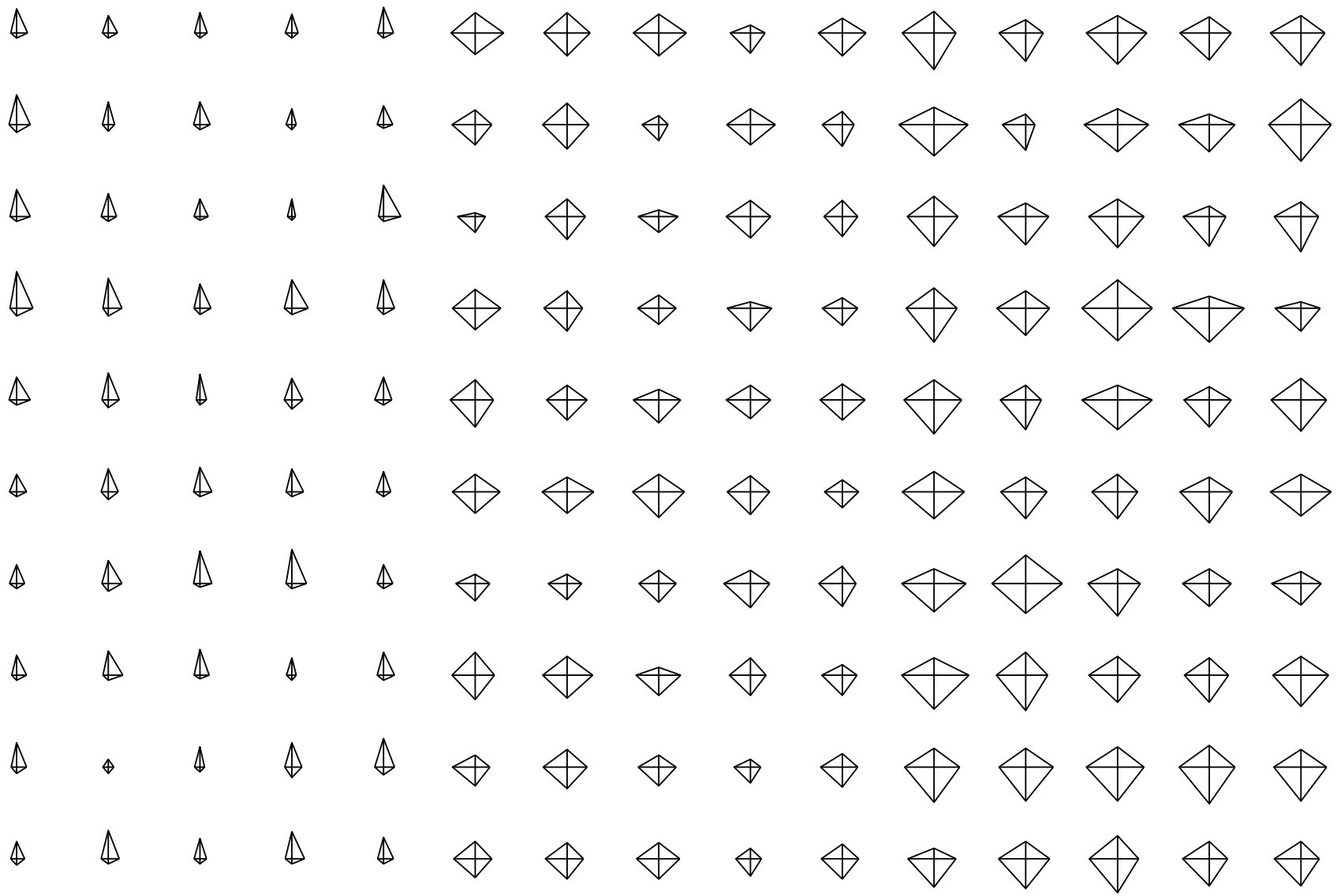
Iris Data Visualized by Chernoff Faces

- Sepal length corresponds to the size of face, sepal width – shape of forehead, petal length – shape of jaw, and petal width – length of nose. *Matlab* is used to obtain the image.



Iris Data Set Visualized by Star Glyphs

- The stars, corresponding to Setosa irises, are smaller than the other. The larger stars correspond to Virginica irises.

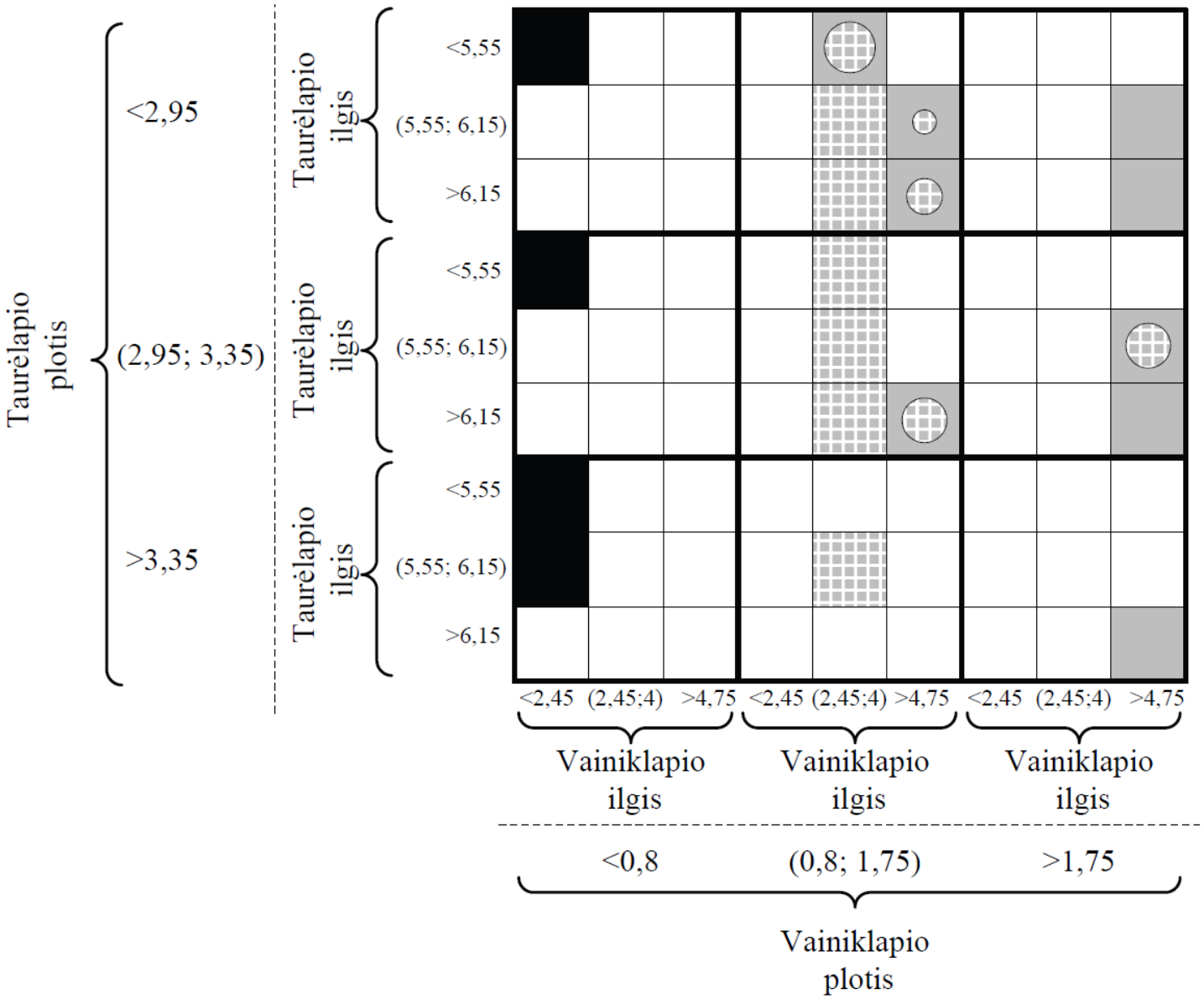


Hierarchical Displays

- *Hierarchical displays* create a structure of an image such that some features are embedded in displays of other features.
- Visualization of some features is displayed in the structure depending on the values of other features.
- Here we discuss two such techniques: *dimensional stacking* and *trellis display*.

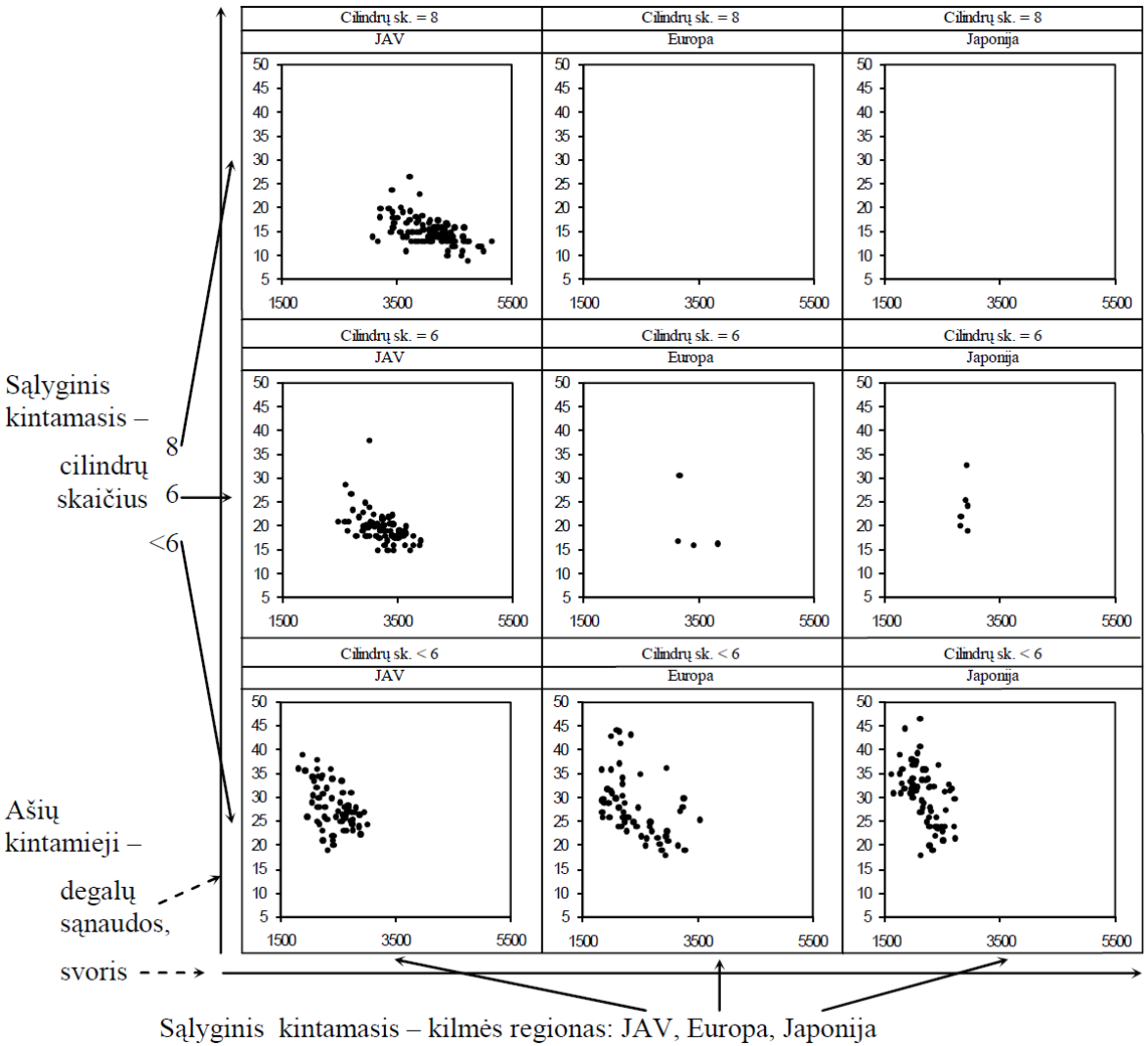
Iris data visualized by dimensional stacking

- Setosa irises (black cells) are displayed separately from the other two species. The other two species overlap.



Auto MPG Data Visualized by Trellis Display

- The axis variables are weight and miles per gallon (MPG).
- The origin and number of cylinders are conditioning variables.



Projection Methods

- Methods that allow us to represent multidimensional data from $\mathbb{R}^n$ in a low-dimensional space $\mathbb{R}^d$, $d < n$, are called projection (dimensionality reduction) methods.
- If the dimensionality of the *projection space* is small enough ($d = 2$ or $d = 3$), these methods may be used to visualize the multidimensional data.
- In such a case, the projection space can be called a *display*, *embedding* or *image space*.
- These methods usually invoke formal mathematical criteria by which the projection distortion is minimized.
  1. Linear projection methods:
     a) principal component analysis,
     b) linear discriminant analysis,
     c) projection pursuit.
  2. Nonlinear projection methods:
     a) multidimensional scaling,
     b) locally linear embedding,
     c) isometric feature mapping,
     d) principal curves.

Criteria of the Projection Quality

- There are some formal mathematical criteria of the projection quality. These criteria are optimized in order to get the optimal projection of multidimensional data onto a low-dimensional space.
- The main goal is to preserve the proportions of distances or estimations of other proximities between the multidimensional points in the image space as well as to preserve, or even to highlight other characteristics of the multidimensional data (for example, clusters).

Linear Transformation

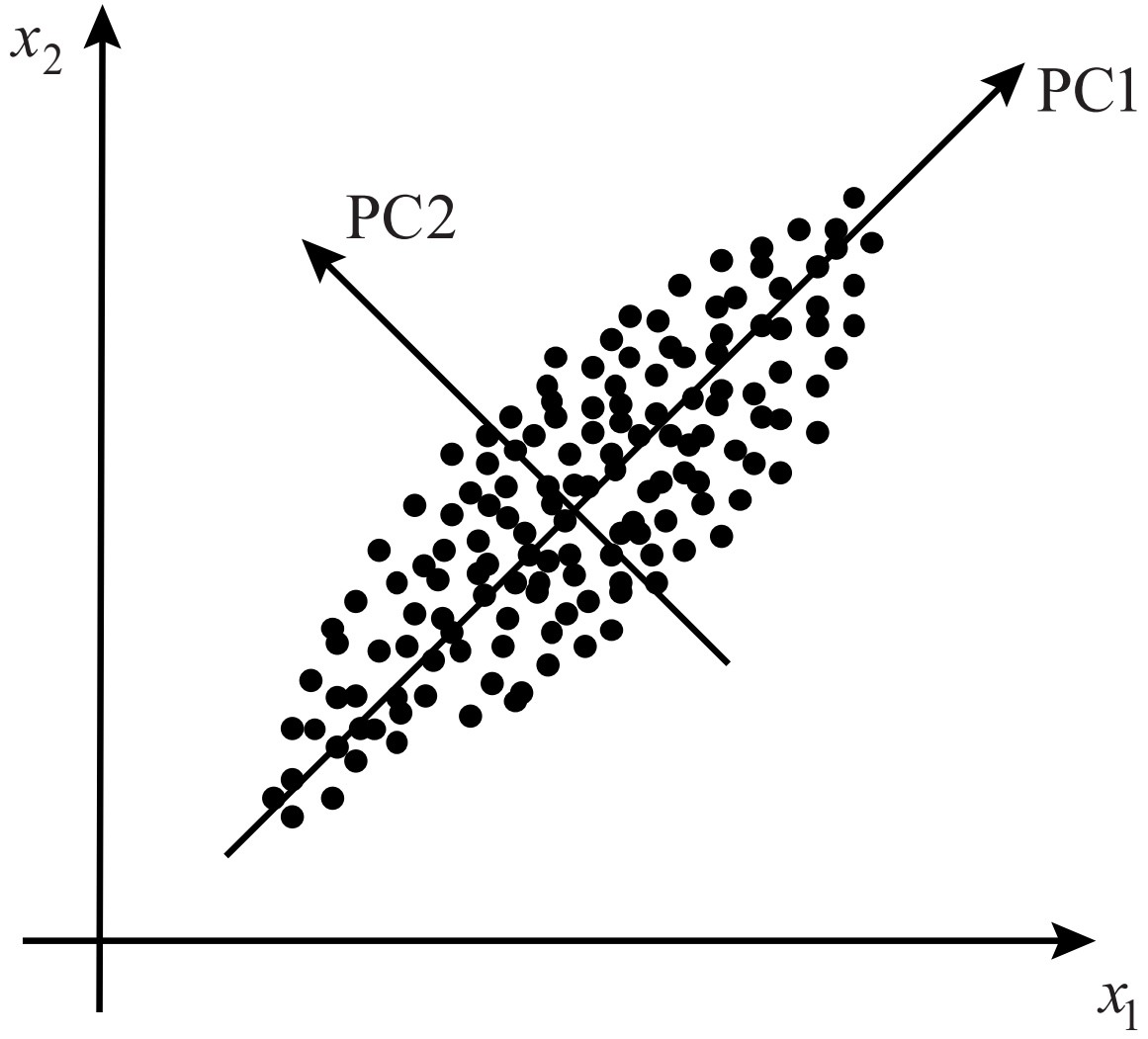
- There are linear and nonlinear projection methods.
- Linear projection methods pursue a linear transformation of data. There are various linear transformations: rotation, shearing, reflection, scaling, etc.
- A *linear transformation* may be described by linear equations
$$Y_i = X_i A.$$
- If $d = n$, i.e. $Y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$ and $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, then A is a square matrix, consisting of n rows and n columns. The matrix A is called a *transformation matrix*.
- If a linear transformation is used for dimensionality reduction, then $d < n$, $Y_i = (y_{i1}, y_{i2}, \ldots, y_{id})$, $i = 1, \ldots, m$, and A is a matrix, consisting of n rows and d columns.

Principal Component Analysis

- The principal component analysis (PCA) is a well-known data analysis technique invented in 1901 by Pearson.
- It is a way of linear transforming a set X of n-dimensional points $X_1, X_2, \ldots, X_m$ into another set Y of n-dimensional points $Y_1, Y_2, \ldots, Y_m$.
- The property of the set is that the largest part of its information content is stored in the first few coordinates (components) of points Y_i, $i = 1, \ldots, m$.
- The principal component analysis is often used to reduce the dimensionality of multidimensional points X_i, $i = 1, \ldots, m$, by discarding some of the components of the points Y_i and by leaving only the first (principal) d ones.
- The principal component analysis projects the data linearly into a low-dimensional space preserving the variance of the data best.

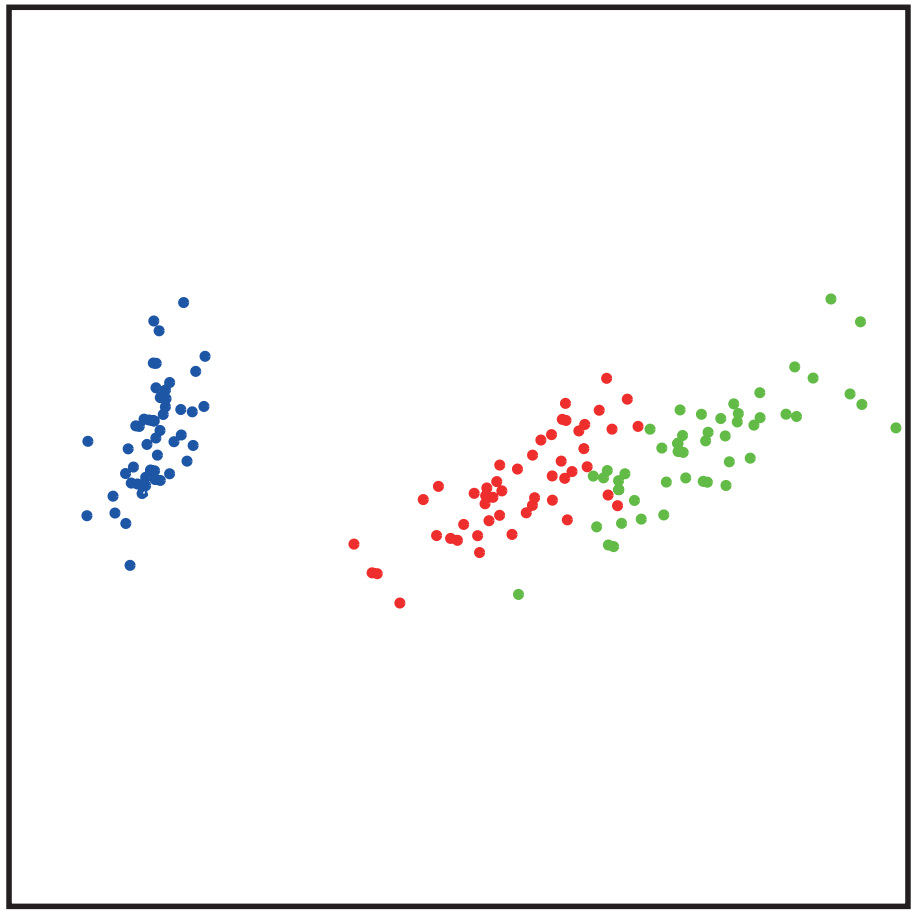
Illustration of Principal Component Analysis

- Figure illustrates a two-dimensional case with two principal components PC1 and PC2.



Iris Data Set Visualized Using PCA

- Setosa irises (marked in blue) are faraway from Versicolor (red) and Virginica (green) irises. There is no exactly expressed boundary between these two species.

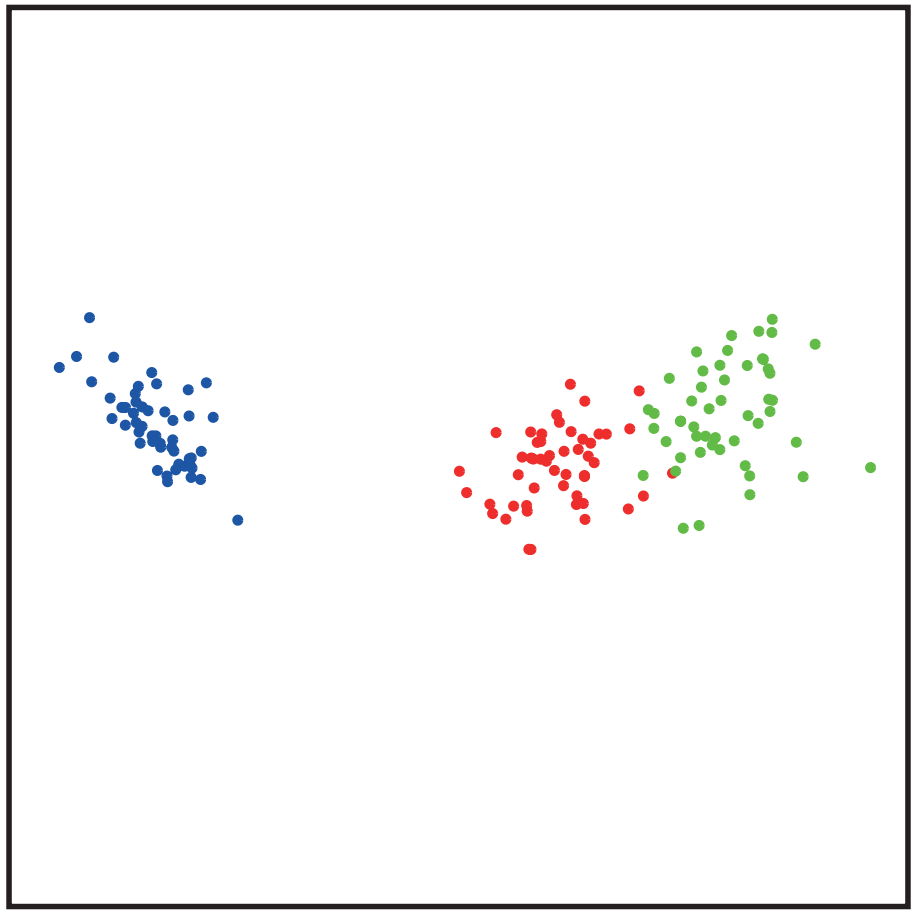


Linear Discriminant Analysis

- In contrast to most other dimensionality reduction methods, a *linear discriminant analysis* (LDA) is a supervised method.
- The method is often called Fisher's discriminant analysis.
- In a supervised strategy, some known properties of data (for example, belonging of the objects to one of classes) are applied.
- LDA transforms multidimensional data to a low-dimensional space, maximizing the linear separability between objects belonging to different classes.

Iris Data Set Visualized Using LDA

- The difference between LDA and PCA is that, in addition, the known classes of objects are applied.



Nonlinear Transformation

- A nonlinear transformation may be described as follows:

$$Y = f(X),$$

where f is a nonlinear function and

$$Y = \{Y_1, Y_2, \ldots, Y_m\} = \{y_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\}.$$

- The nonlinear transformation is more complicated than the linear one and requires more time-consuming computations. However, such a transformation allows us to preserve the characteristics of multidimensional data better as compared with the linear transformation if $d < n$, i.e. the data are projected to a lower-dimensional space.

Multidimensional scaling (MDS) – a technique for exploratory analysis of multidimensional data

- Pairwise dissimilarities between n objects are given by a matrix (δ_{ij}), $i, j = 1, \ldots, n$, it is supposed that $\delta_{ij} = \delta_{ji}$.
- The points representing objects in an m-dimensional embedding space $\mathbf{x}_i \in \mathbb{R}^m$, $i = 1, \ldots, n$ should be found whose inter-point distances fit the given dissimilarities.
- The problem is reduced to minimization of a fitness criterion, e.g. so called *Stress* function

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(d\left(\mathbf{x}_i, \mathbf{x}_j\right) - \delta_{ij} \right)^2,$$

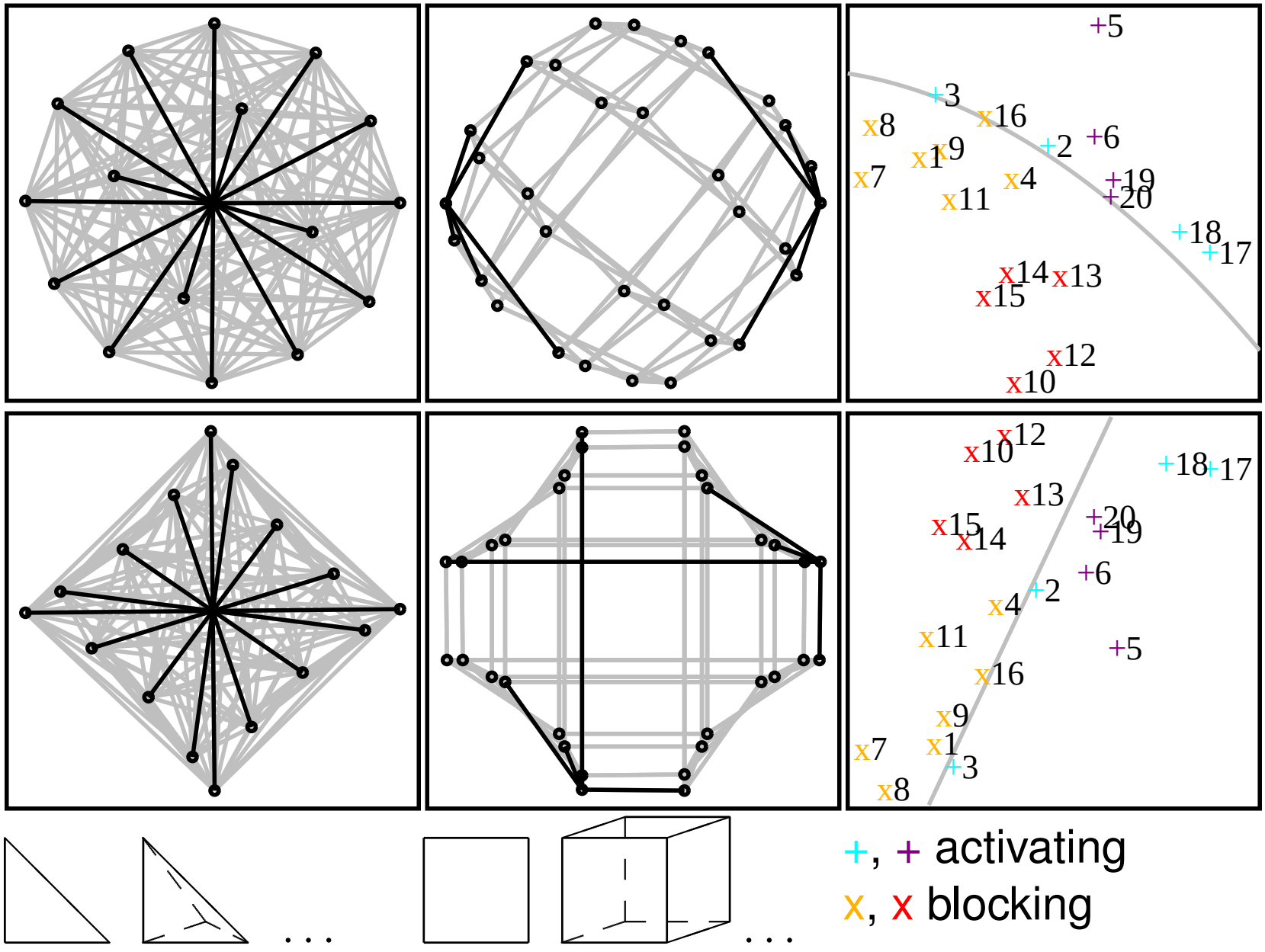
where $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$; $d(\mathbf{x}_i, \mathbf{x}_j)$ denotes the distance between the points $\mathbf{x}_i$ and $\mathbf{x}_j$; weights $w_{ij} > 0, i, j = 1, \ldots, n$.

MDS is a difficult global optimization problem

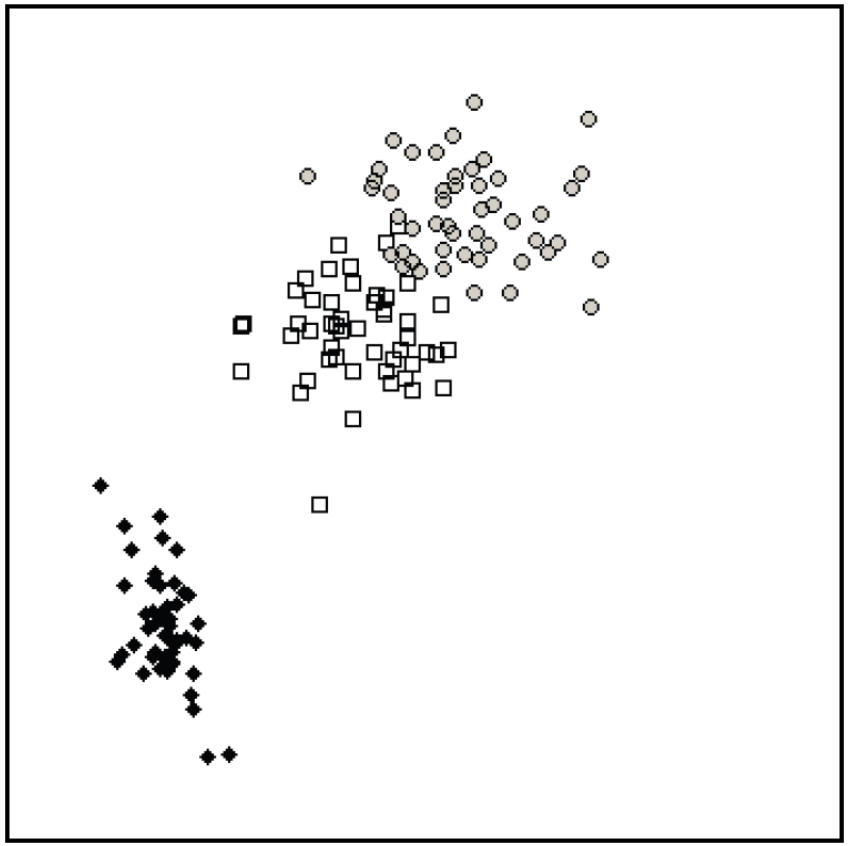
- Although *Stress* function is defined by an analytical formula which seems rather simple, it normally has many local minima.
- The problem is high dimensional: $\mathbf{x} \in \mathbb{R}^N$ and the number of variables is equal to $N = n \times m$.
- *Stress* function is invariant with respect to translation, rotation and mirroring.
- Smoothness of *Stress* function depends on distances $d(\mathbf{x}_i, \mathbf{x}_j)$, however, non-differentiability normally cannot be ignored. Minkowski distances

$$d_r(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^r \right)^{1/r}.$$

MDS with Euclidean and city-block distances



Iris Data Set Visualized Using Multidimensional Scaling

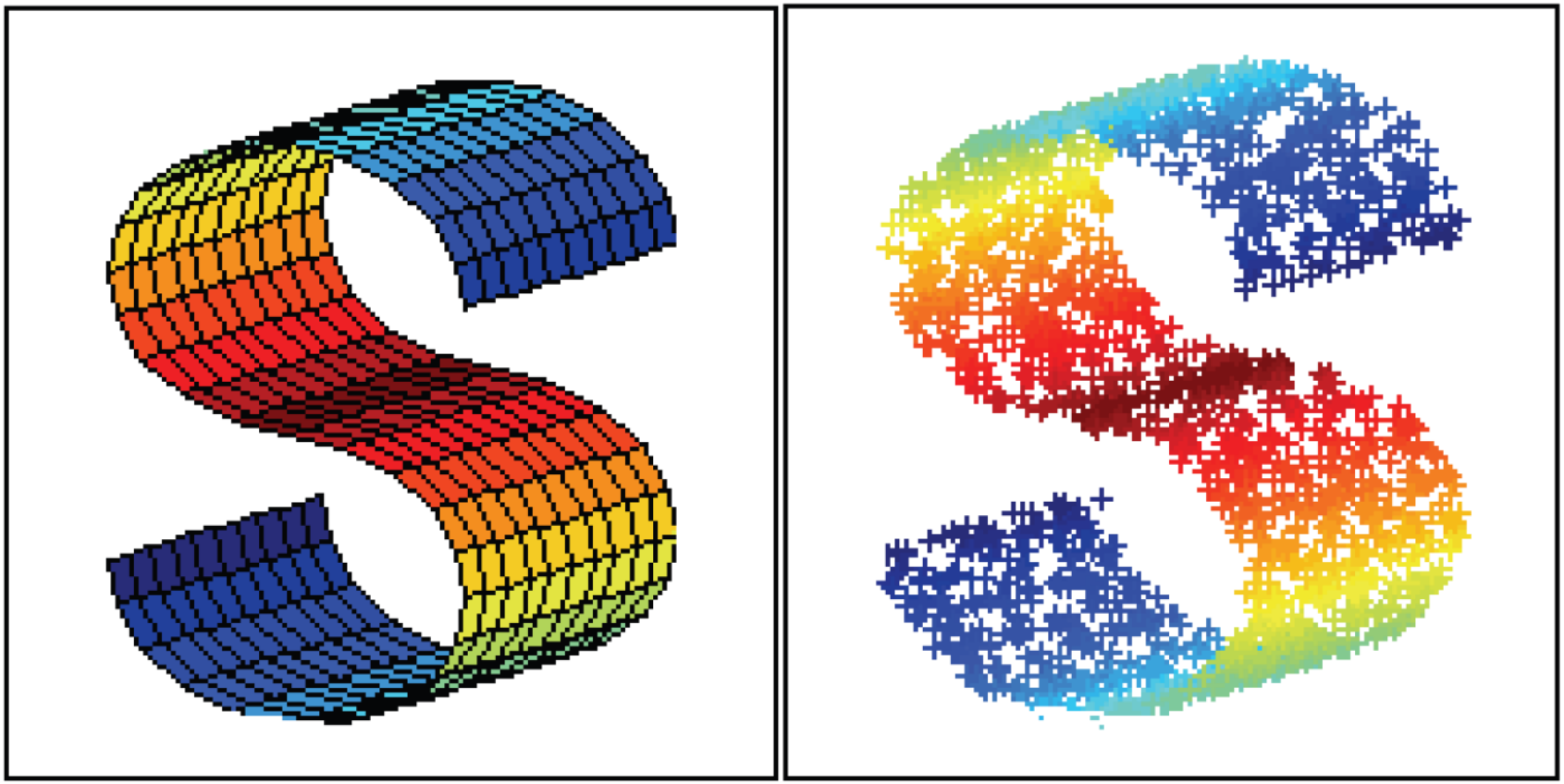


Isometric feature mapping (ISOMAP)

- *Isometric feature mapping* (ISOMAP) can be assigned to the group of multidimensional scaling. ISOMAP is designed for dimensionality reduction as well as for visualization of multidimensional data. An assumption is made that the multidimensional points are located on a lower-dimensional manifold. Therefore geodesic distances are used as a measure of proximity between the multidimensional points.
- Usually Euclidean distances between the points (as a proximity measure) are used in multidimensional scaling. In this case, the existence of a manifold is not taken into consideration.
- In ISOMAP, the geodesic distance is a proximity measure between the multidimensional points. A *geodesic distance* is the length of the shortest path between two points along the surface of a manifold.

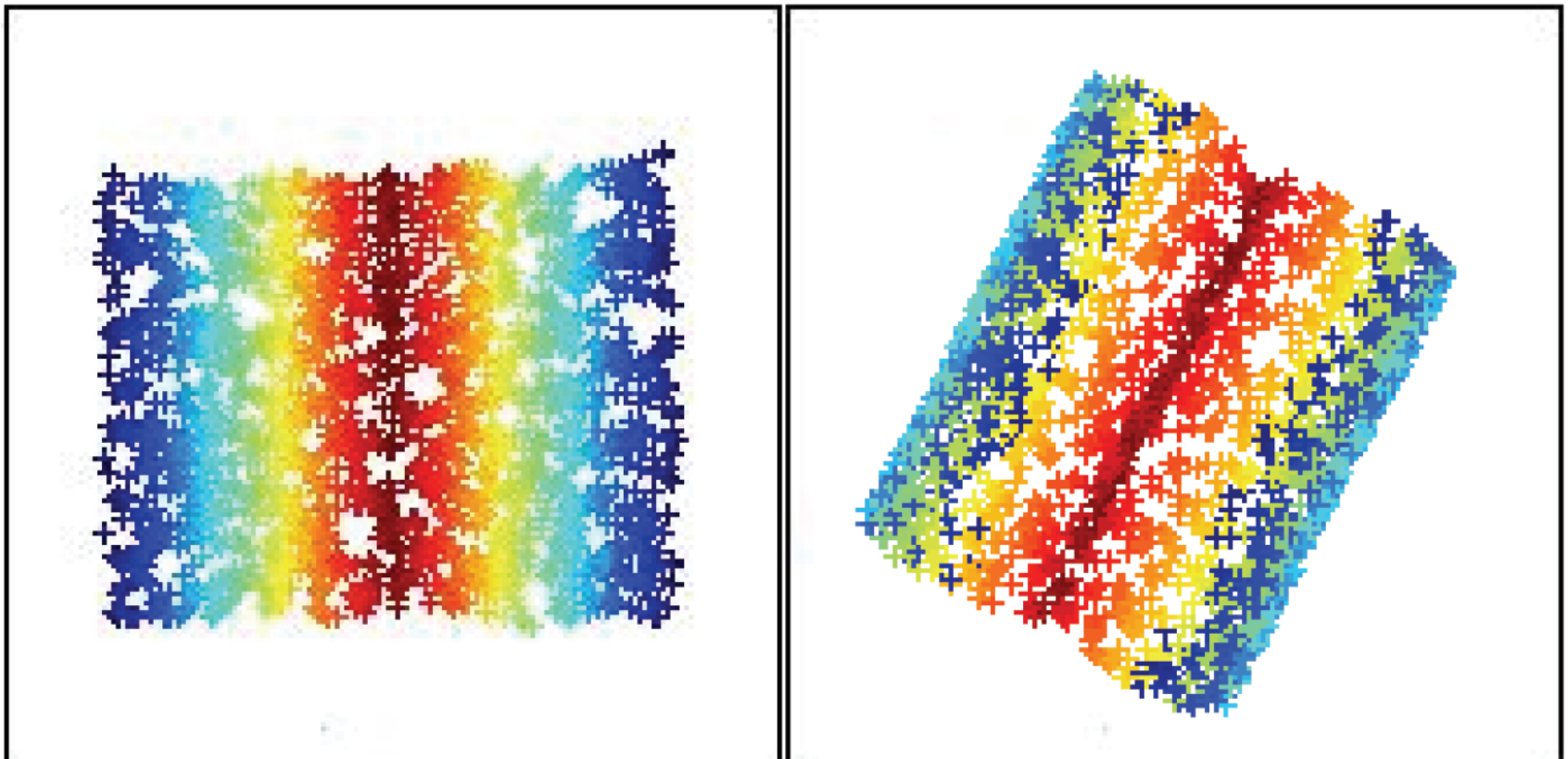
S-manifold

- The S-manifold is presented, $n = 3$. The points on the manifold are also shown, $m = 1000$.



ISOMAP and MDS Projections of S-manifold

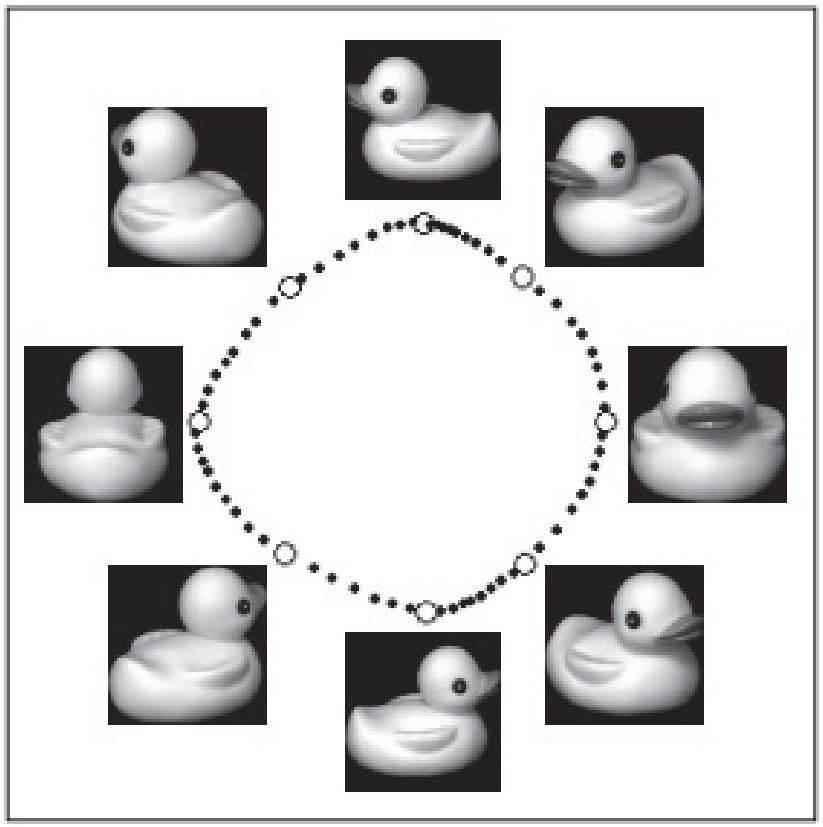
- The structure of the manifold is well preserved by ISOMAP, because the S-manifold is unfolded: the farthest points on the manifold remain the farthest ones on the projection.
- The farthest points obtained by MDS are pale blue. These points are the farthest in multidimensional space in the sense of Euclidean distances, but they are not farthest in the sense of geodesic distances on the S-manifold.



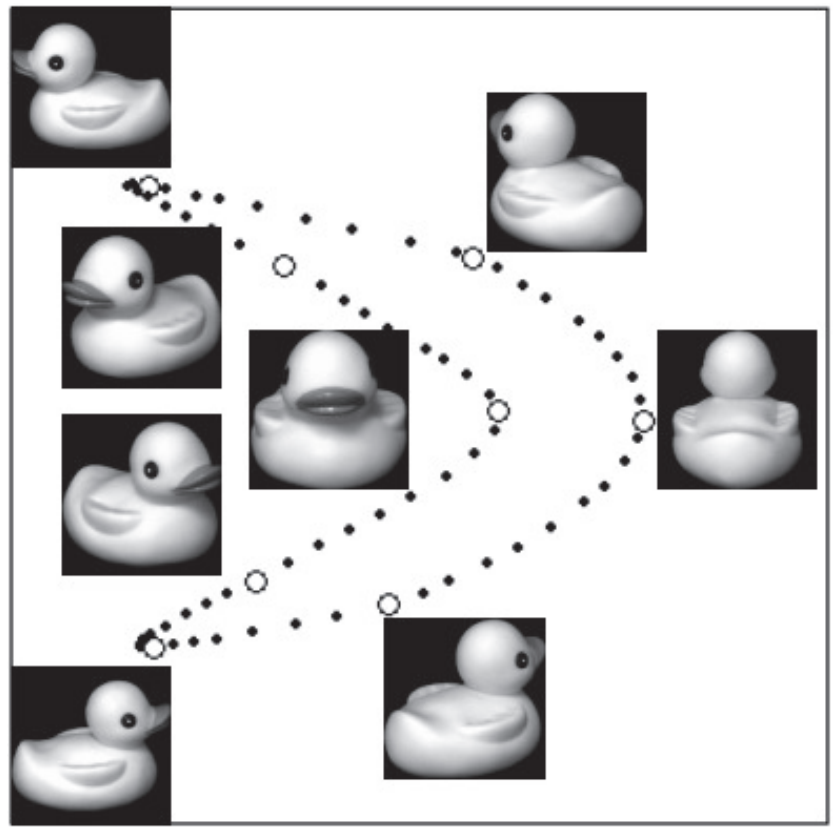
Locally Linear Embedding

- *Locally linear embedding* (LLE) is a nonlinear method for dimensionality reduction and manifold learning.
- Given a set of data points distributed on a manifold in a multidimensional space, LLE is able to project the data to a low-dimensional space by unfolding the manifold.
- LLE works by assuming that the manifold is well sampled, i.e. there are enough data, each data point and its neighbors lie on or close to a locally linear patch.
- Therefore, a data point can be approximated as a weighted linear combination of its neighbors. The basic idea of LLE is that such a linear combination is invariant under linear transformations (translation, rotation, and scaling) and, therefore, it should remain unchanged after the manifold has been unfolded to a low-dimensional space.
- The low-dimensional configuration of data points is obtained by solving two least squares optimization problems.

Visualization of Pictures of a Rotating Duckling by LLE



$k = 4$



$k = 9$