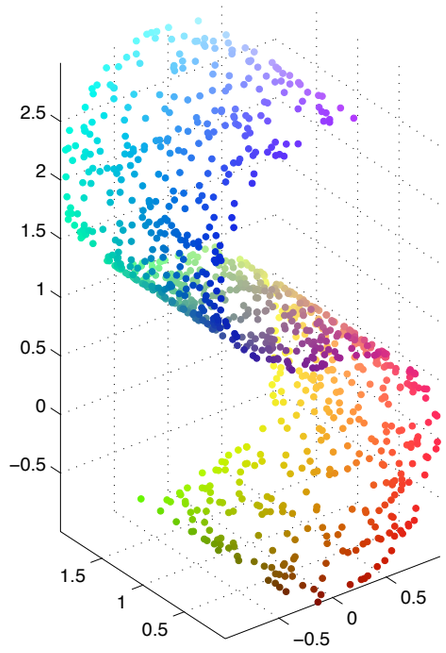
MTTTS17
Dimensionality Reduction and Visualization

Spring 2020
Jaakko Peltonen

Lecture 6: Nonlinear dimensionality reduction, part 1

Reminder: Dimension reduction methods

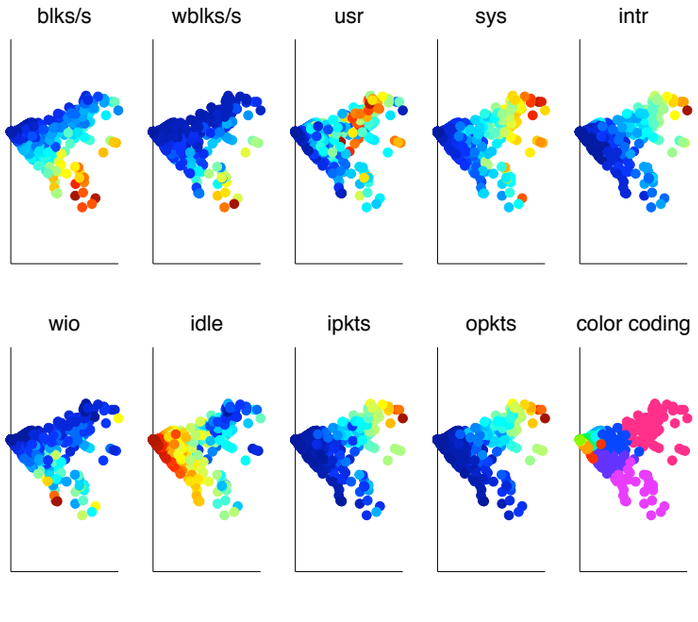
- What to do if... I have to analyze multidimensional data???
- Solution: Dimensionality Reduction from 17 dimensions to 1D, 2D or 3D
- Problem: How to project the nodes *faithfully* into low-dimensional space (1D-3D)?



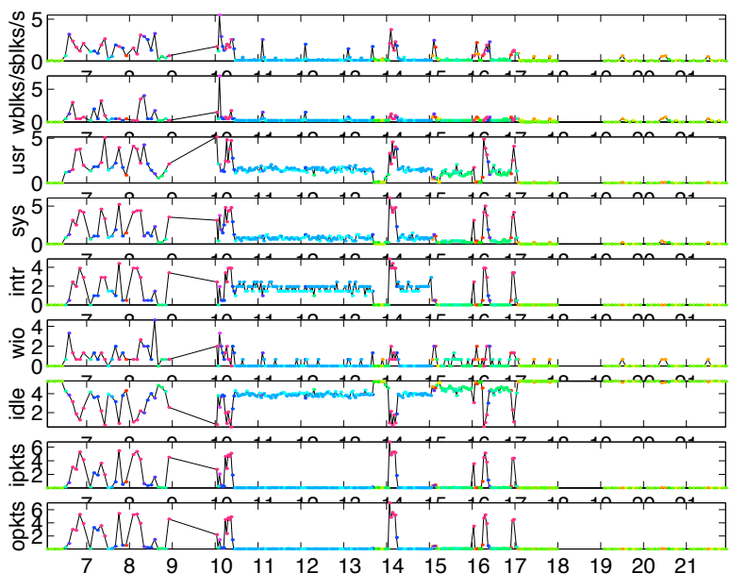
Original 3D data

2D projection by Curvilinear Component Analysis

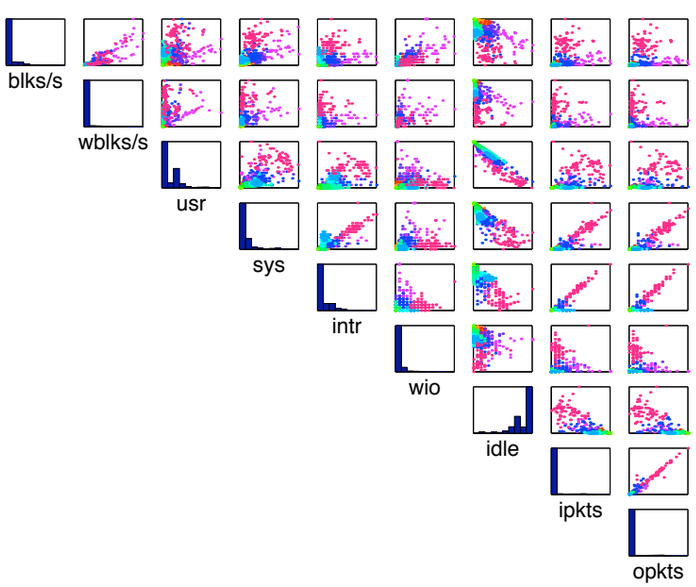
Visualization of a data set



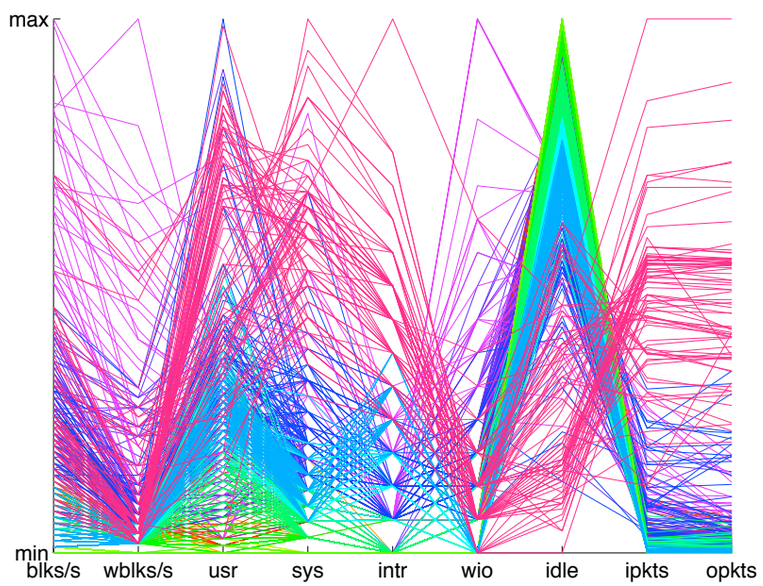
(a) Component planes



(b) Time-series



(c) Scatterplot matrix



(d) Parallel axis

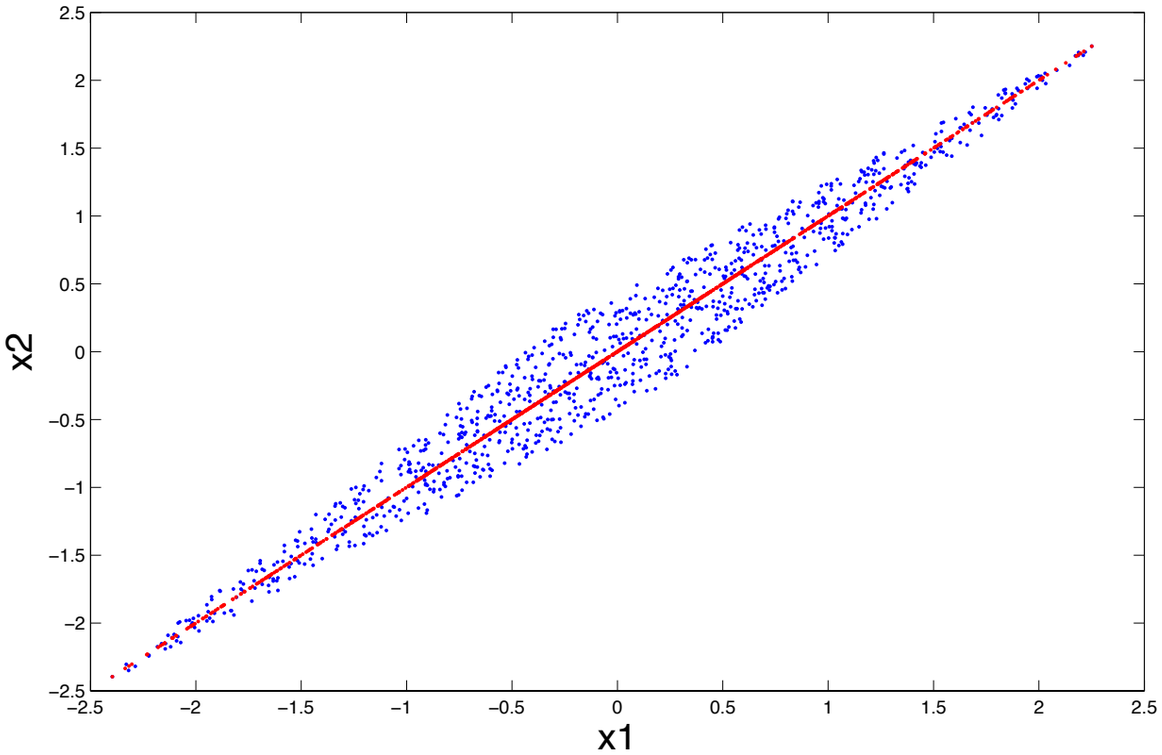
Analysing a lot of dimensions individually takes a lot of time and can miss relevant properties of the data, so we use dimensionality reduction to try to capture the important data properties in a few dimensions.

(Pictures from Juha Vesanto's doctoral thesis, 2002.)

Principal component analysis (PCA)

- PCA is stable, there are no additional parameters, and it is guaranteed always to converge to the same optima.
- Hence, PCA is usually the first dimension reduction method to try (if it doesn't work, then try something more fancy)

Principal component analysis (PCA)



Covariance matrix (after subtracting mean from data)

$$C = X^T X = \begin{bmatrix} 999.00 & 979.84 \\ 979.84 & 999.00 \end{bmatrix}$$

Eigenvector matrix (projection to principal components)

$$V = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

Eigenvalues (variance along principal components)

$$[\lambda_1 \lambda_2] = \begin{bmatrix} 1978.8 & 19.2 \end{bmatrix}$$

Proportion of variance explained

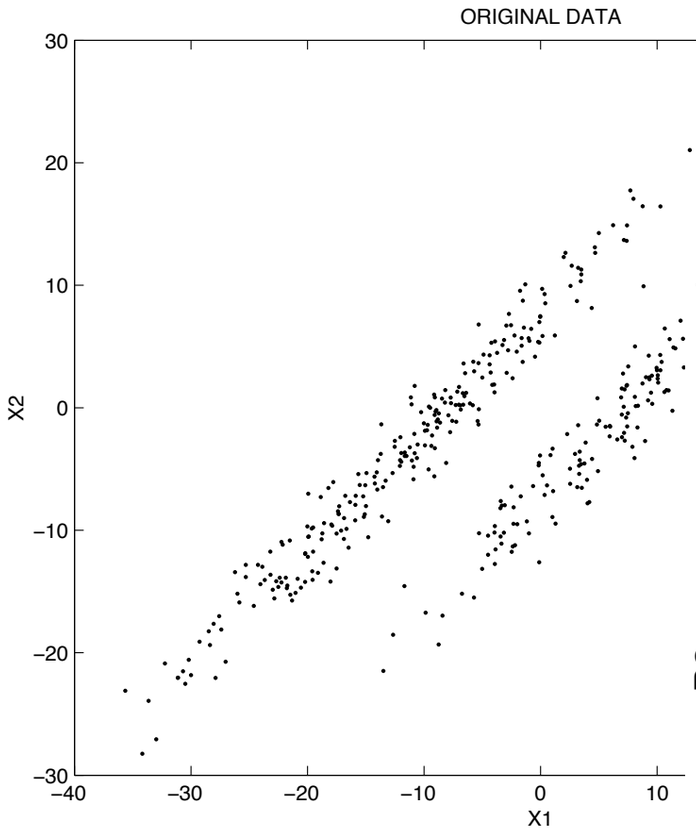
$$TVE = \frac{\lambda_1}{\sum_{i=1}^{d} \lambda_i} = \begin{bmatrix} 0.9904 \end{bmatrix}$$

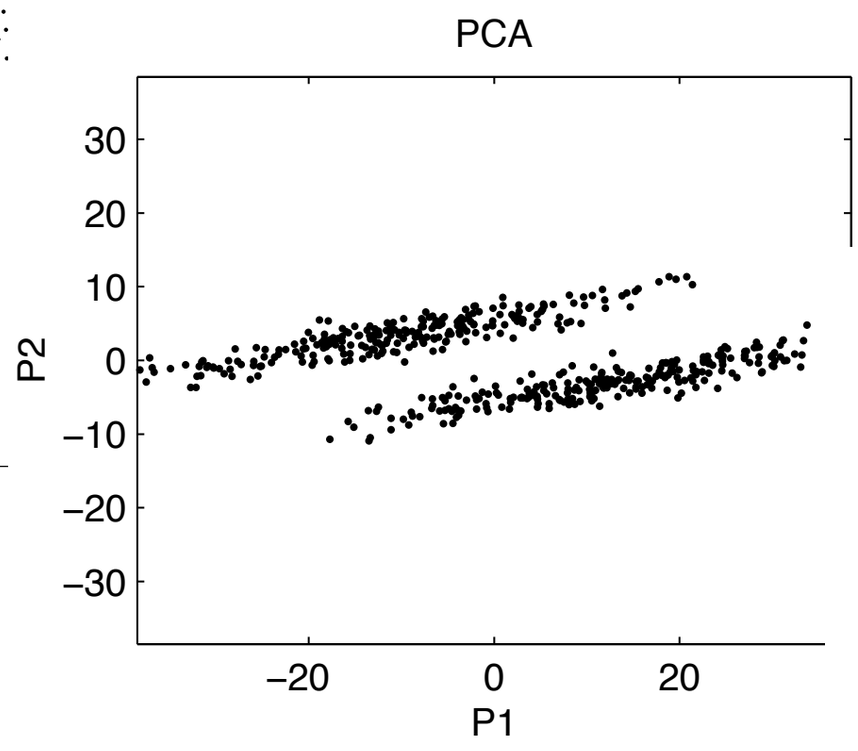
Reconstruction error

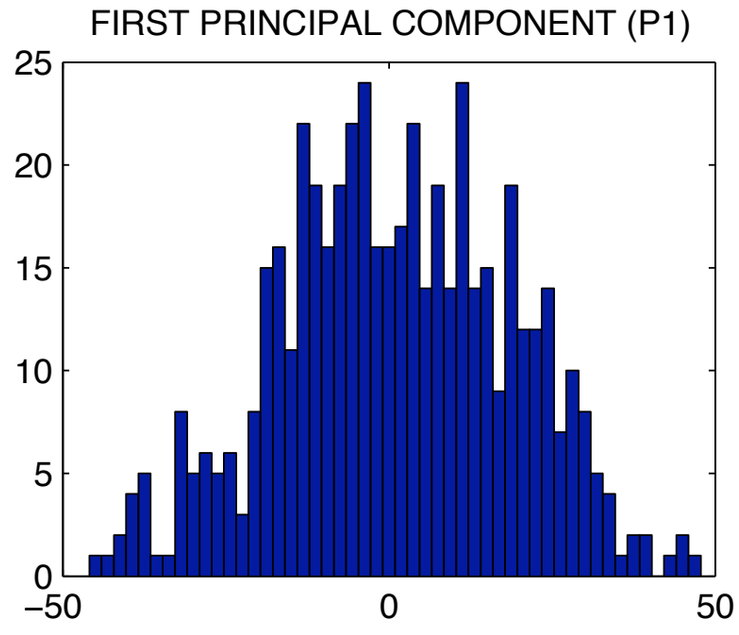
$$E = \frac{\sum_{i=1}^{d} var[(\hat{X}^i - X^i)]}{d} = \begin{bmatrix} 0.0096 \end{bmatrix}$$

Two clusters

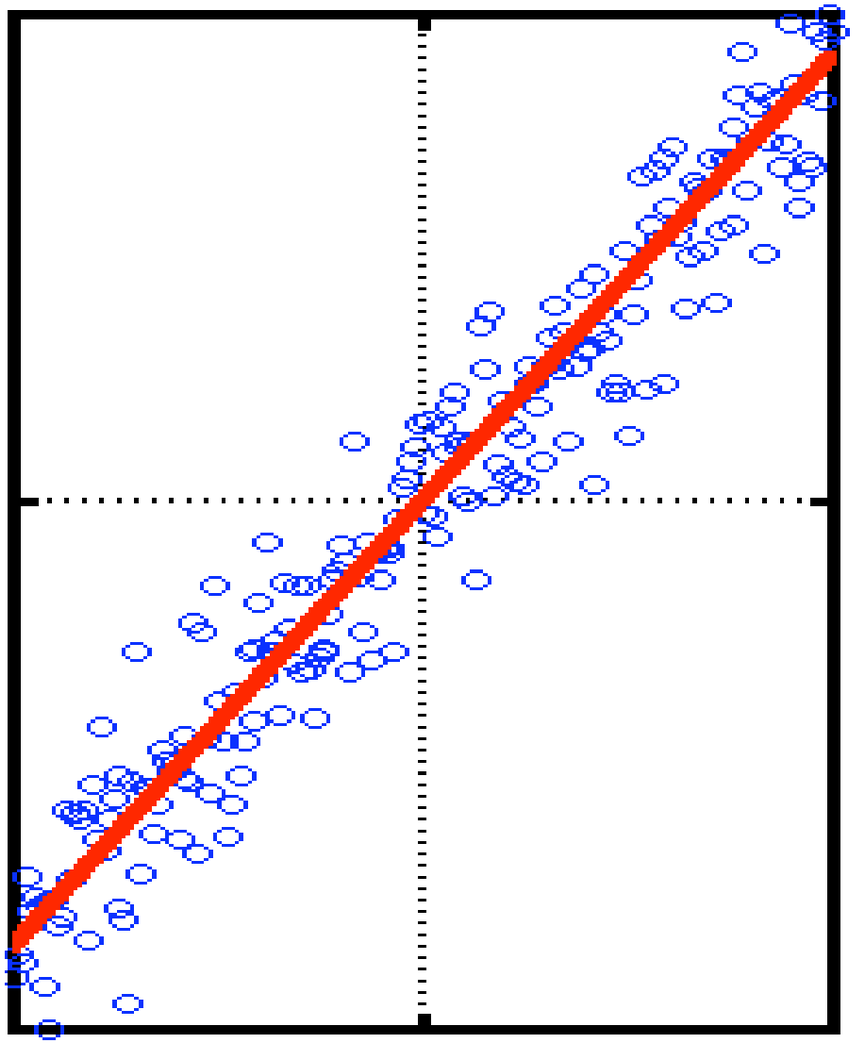
- The PCA fails to separate the clusters (you don't see cluster structure from the 1D visualization, lower right)

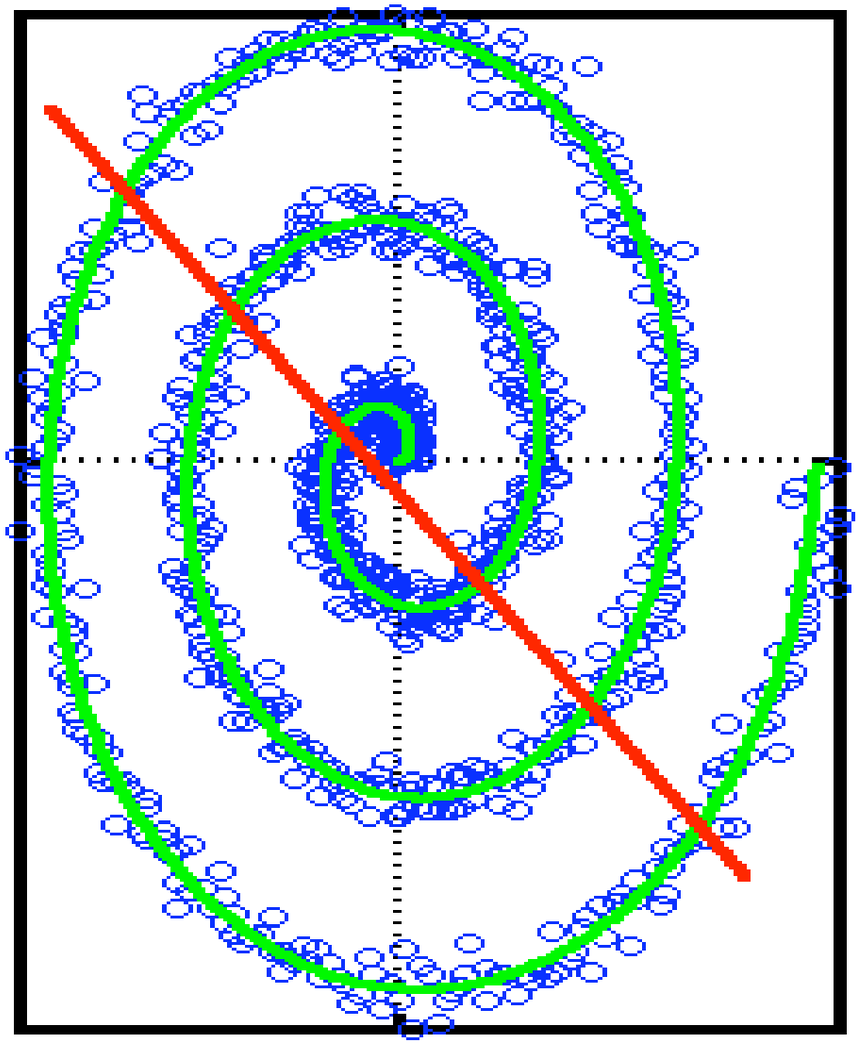






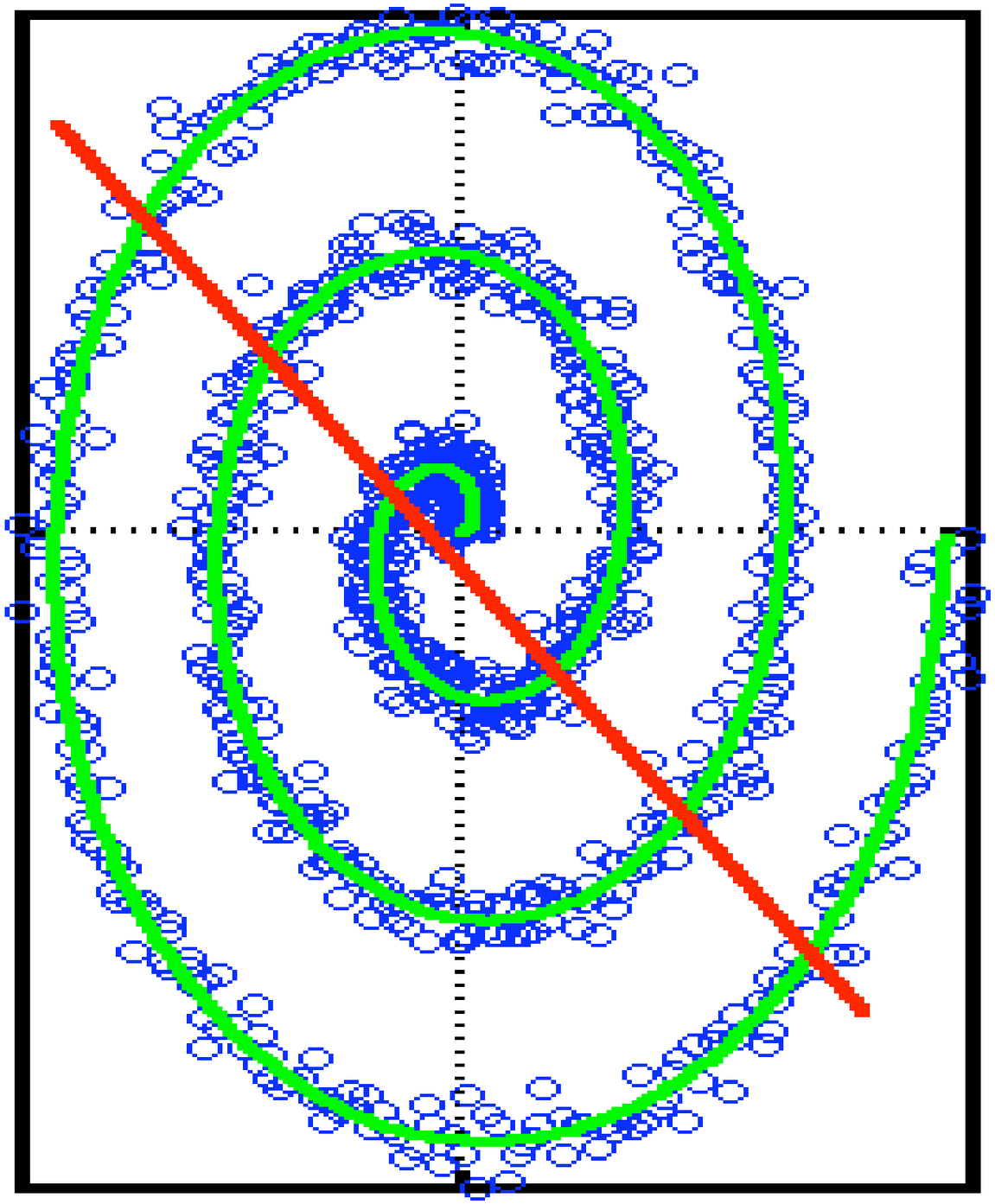
Nonlinear data





The first principal component is given by the red line. The green line on the right gives the "correct" non-linear dimension (which PCA is of course unable to find).

Manifolds



- Left, PCA mapping would not find the "correct" 1D manifold, shown in green, because they try to preserve global features.
- Often, preserving local features, like *trustworthiness*, *continuity* (next lecture) or conformity - or manifold structure - is more important than global properties.

Dimension reduction methods

- Matlab+Excel example - PCA versus nonlinear multidimensional scaling!

Nonlinear dimensionality reduction and Manifold Learning

Topology, Spaces and Manifolds

- If variables depend on each other their joint distribution, the *support* of their joint distribution does not span the whole space → induce some structure in distribution (geometrical locus, object in space)
- Topology (mathematics) studies properties of objects that are preserved through deformations, twisting and stretches
  Tearing forbidden, guaranteeing that the intrinsic structure or connectivity is not altered (circle topol. equivalent to ellipse)
- Tearing may still be interesting operation (unfolding the earth)
- Objects with same topological properties are called *homeomorphic*

Topology, Spaces and Manifolds

A topological space is a set for which a topology is specified

For a set $\mathcal{Y}$ a topology *T* is defined as a collection of subsets of $\mathcal{Y}$ with the following properties:

- trivially $\varnothing \in T$ and $\mathcal{Y} \in T$
- whenever 2 sets are in *T*, so is their intersection
- whenever 2 or more sets are in *T*, so is their union

Definition holds for Cartesian space $\mathbb{R}^d$ as for graphs (example: topology in $\mathbb{R}$ (set of real numbers) is union of all open intervals)

Topology, Spaces and Manifolds

More generally: topological space can be defined using neighborhoods (Ns) and Haussdorf's axioms (1919)

ϵ-neighborhood for $\mathbf{y} \in \mathbb{R}^d$ or infinitesimal open set often defined as open-Ball $B_\epsilon(\mathbf{y}) \rightarrow$ a set of points inside a *d*-dim. hollow sphere of radius $\epsilon > 0$ centered on **y**

Topology, Spaces and Manifolds

More generally: topological space can be defined using neighborhoods (Ns) and Haussdorf's axioms (1919)

- each point **y** corresponds at least one neighborhood $\mathcal{U}(\mathbf{y})$ containing **y**
- if $\mathcal{U}(\mathbf{y})$ and $\mathcal{V}(\mathbf{y})$ same **y**, then $\mathcal{W}(\mathbf{y}) \subset \mathcal{U}(\mathbf{y}) \cap \mathcal{V}(\mathbf{y})$ exists
- if $\mathbf{z} \in \mathcal{U}(\mathbf{y})$ then $\mathcal{V}(\mathbf{z})$ of **z** exists such that $\mathcal{V}(\mathbf{z}) \subset \mathcal{U}(\mathbf{y})$
- for 2 distinct points 2 disjoint Ns of these points exist

ϵ-neighborhood for $\mathbf{y} \in \mathbb{R}^d$ or infinitesimal open set often defined as open-Ball $B_\epsilon(\mathbf{y}) \rightarrow$ a set of points inside a *d*-dim. hollow sphere of radius $\epsilon > 0$ centered on **y**

Topology, Spaces and Manifolds

- Within this framework a topological *manifold M is a* topological space that is locally Euclidean
(any object that is nearly flat on small scales)
- Manifold can be compact, non-compact, connected or disconnected
- *Embedding*: representation of topological object (manifold, graph,etc.) in cartesian space (usually $\mathbb{R}^d$) preserving topological properties
- Smooth manifold is called a differentiable manifold
- For example: hollow *d*-dimensional hypersphere is a (*d* - 1) manifold

Topology, Spaces and Manifolds

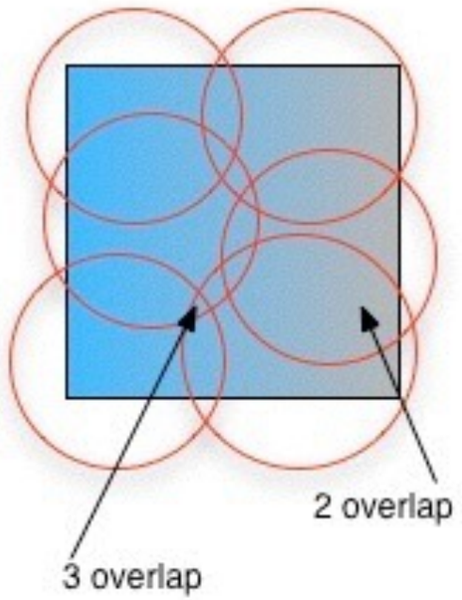
- Whitney (1930) showed that any *P*-manifold can be embedded in $\mathbb{R}^{2P+1}$
  - A line can be embedded in $\mathbb{R}^1$
  - A circle is a compact 1-manifold, can be embedded in $\mathbb{R}^2$
  - Trefoil (knotted circle) reaches the bound $\rightarrow \mathbb{R}^3$
- In practice the manifold is nothing more than the underlying support of a data distribution known only through finite samples
- 2 Problems appear:

  (1) Dimen. reduction works with sparse and limited data

  (2) assuming manifold takes into account the support of data distr. but not other properties such as its density
  → problematic for latent variable finding (the model of data density is of prime importance)
- Manifold does not account for noise (points may lie nearby)
  → DR re-embeds a manifold, noisy data is projected onto it

Intrinsic Dimension

- The *intrinsic dimension(ality)* of a random vector **y** equals the *topological dimension* of the support $\mathcal{Y}$ of the distribution of **y**
- Given a topological space $\mathcal{Y}$, the *covering* of a subset *S* is a collection *C* of open subsets whose union contains *S*

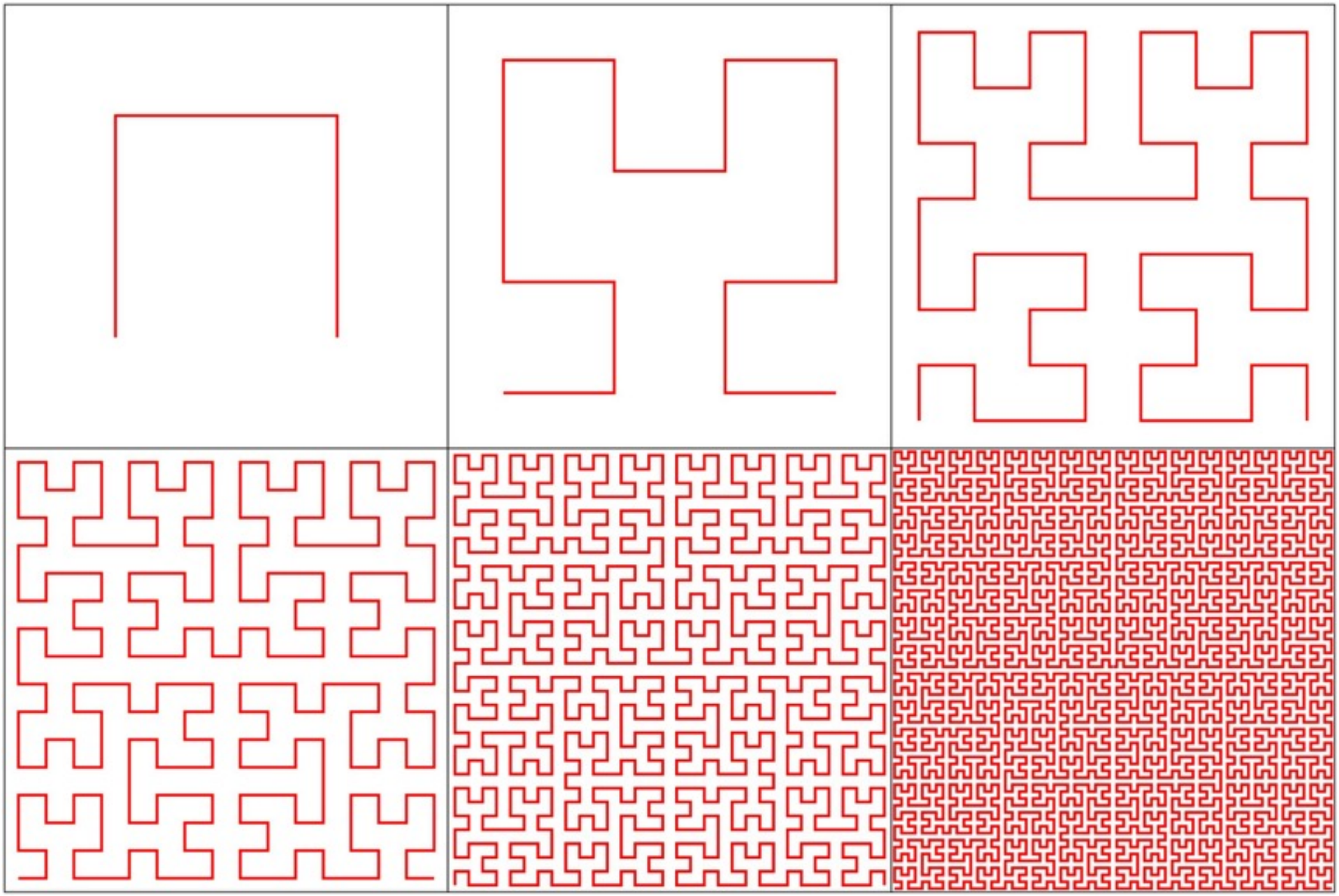


- A subset *S* of topological space $\mathcal{Y}$ has topological dimension D_{top} (*Lebesgue covering dimension*) if every covering *C of S has a refinement C'* in which every point of *S* belongs to a maximum of $D_{top}+1$ open balls

- Topological dimension is difficult to estimate with a finite set of points and for some objects behave weird (Hilbert curve) the topological dimension just seems wrong (Sierpinski triangle)
  → use other definitions of the intrinsic dimension like fractal dimension or definitions based on DR methods

Hilbert Curve

1D object that evolves iteratively and progressively fills a square



Intrinsic Dimension

Fractal Dimensions most popular examples

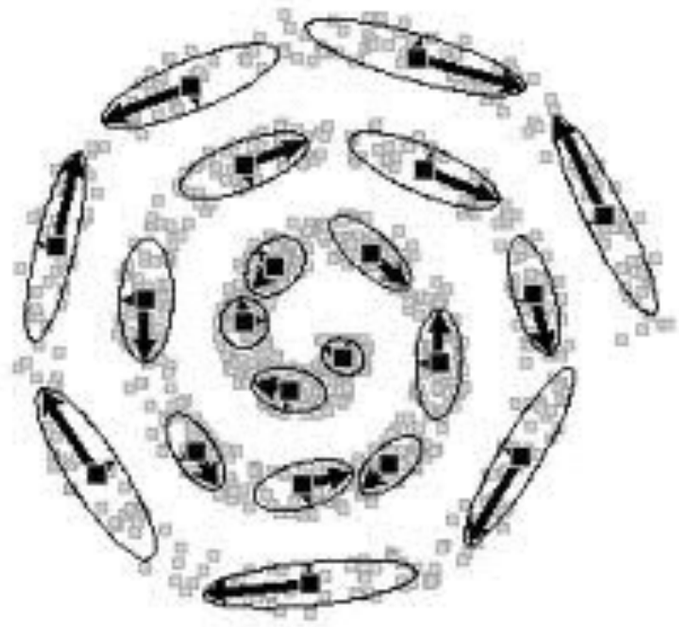
- *Box-Counting* dimension (Capacity dimension)
  Determine the hypercube that circumscribes all data points, decompose it into a grid of smaller hypercubes with edge length ϵ , determine $N(\epsilon)$ (number of hypercubes) that are occupied by one or several data points, compute $P \propto \frac{\log N(\epsilon)}{\log(1/\epsilon)}$
  Limit $\epsilon \rightarrow 0$ gives d_{cap}
- *Correlation* dimension (Grassberger and Procaccia)
  similar interpretation as capacity dimension, but local: look at number of neighboring points closer than certain threshold

Local methods:

- decompose the space into local patches
- carry out PCA on each space window assuming the manifold is approximately linear on that scale
- dimension of the manifold is obtained as average estimate of the local PCAs (weighted by number of points in corresponding window)

Intrinsic Dimension

Sketches for box counting (left) and local PCA (right) intrinsic dimension estimation



Outlook

- (un-)supervised Dimensionality reduction and visualization
- Generalized framework for dimensionality reduction
- Quality assessment

Nonlinear dimensionality reduction methods

Introduction

- Represent high-dimensional data by low-dimensional counterparts preserving as much information as possible
- Ill-posed problem: which data is relevant to the user? (dependent on the specific data domain and situation at hand)
- Huge variety of methods proposed with different properties

Dimension reduction methods

- There are several methods, with different optimization goals and complexities
- We will go through some of them (most not in any detail, last ones not at all):
  - **Principal component analysis (PCA)** - "simple" linear method that tries to preserve global distances
  - **Multidimensional scaling (MDS)** - tries to preserve global distances
  - *Sammon's projection* - a variation of the MDS, pays more attention to short distances
  - *Isometric mapping of data manifolds (ISOMAP)* - a graph-based method (of the MDS spirit)
  - *Curvilinear component analysis (CCA)* - MDS-like method that tries to preserve distances in small neighborhoods
  - *Maximum variance unfolding* - maximizes variance with the constraint that the short distances are preserved (an exercise in semidefinite programming)
  - *Self-organizing map (SOM)* - a flexible and scalable method that tries a surface that passes through all data points (originally developed at HUT)
  - *Independent component analysis (ICA)* - a fast linear method, suitable for some applications
  - Nerv (NEighbor Retrieval Visualizer, originally developed at HUT)

Different methods, different properties

- **Spectral techniques:** the rely on the spectrum of the neighborhood graph of the data, preserving important properties of it.

  - Example methods: Locally Linear Embedding (LLE), Isomap, Laplacian Eigenmaps

  - usually unique algebraic solution of the objective

  - in order to make the cost functions unimodal and to make algebraic solution of objective possible, the methods are based on very simple affinity functions

Different methods, different properties

- **Non-parametric methods:** they usually do not find a general mapping function from a high-dimensional space to a lower-dimensional space, instead they find a mapping a finite data set

  - They can use more complicated affinities between data points, but it comes with higher computational costs

  - Additional modeling/optimization and computational effort must be done for *out-of-sample extension* (for mapping new data points that were not int he training set)

Different methods, different properties

- **Explicit mapping functions:** some methods explicitly learn (infer) a (non-)linear mapping function
  - linear functions: Principal Component Analysis, Linear Discriminant Analysis
  - nonlinear functions: autoencoder networks, locally linear coordination

Different methods, different properties

- **Supervised techniques:** use ”ground truth” information provided by a teacher (oracle) to learn the mapping or mapping function

  - Linear Discriminant Analysis, Partial Least Squares regression, adaptive metrics

    - non-linear extensions by kernels

Dimensionality Reduction Setting

Assume we have

- a high-dimensional data space $\mathcal{X}$
- a data set from the data space: $\mathbf{x}^i \in \mathbb{R}^N,\ i = 1 \ldots n$

We want to find

- an output space (embedding space) $\mathcal{E}$
- low-dimensional representatives $\boldsymbol{\xi}^i \in \mathbb{R}^M$ for the data set in the embedding space

General aim of dimensionality reduction: find a **mapping**

$$f : \mathbb{R}^N \rightarrow \mathbb{R}^M$$

such that the **interesting properties** of the data distribution in $\mathcal{X}$ are **preserved** as well as possible also in $\mathcal{E}$

Multidimensional scaling (MDS)

- *Multidimensional scaling (MDS)* is a dimension reduction method that tries to preserve a measure of similarity (or dissimilarity or distance) between pairs of data points
- MDS has roots in the field of psychology (one consequence: lots of conventional notation)
- MDS can be used as
  - an exploratory visualization technique to find the structure of the data; and
  - a tool to test hypothesis.

Color similarities

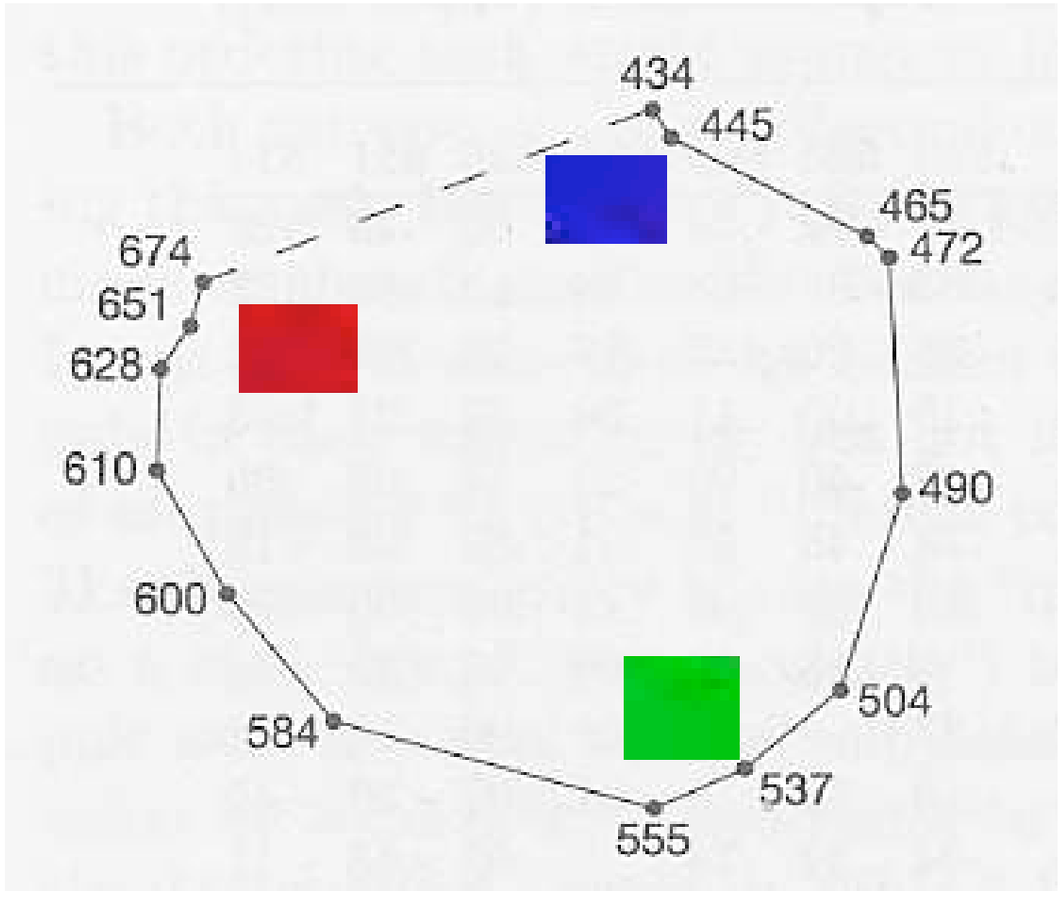
- Psychological test in 1950's: how is the similarity of colors perceived?
- Pairs of 14 colors were rated by 31 people. Ratings were averaged.

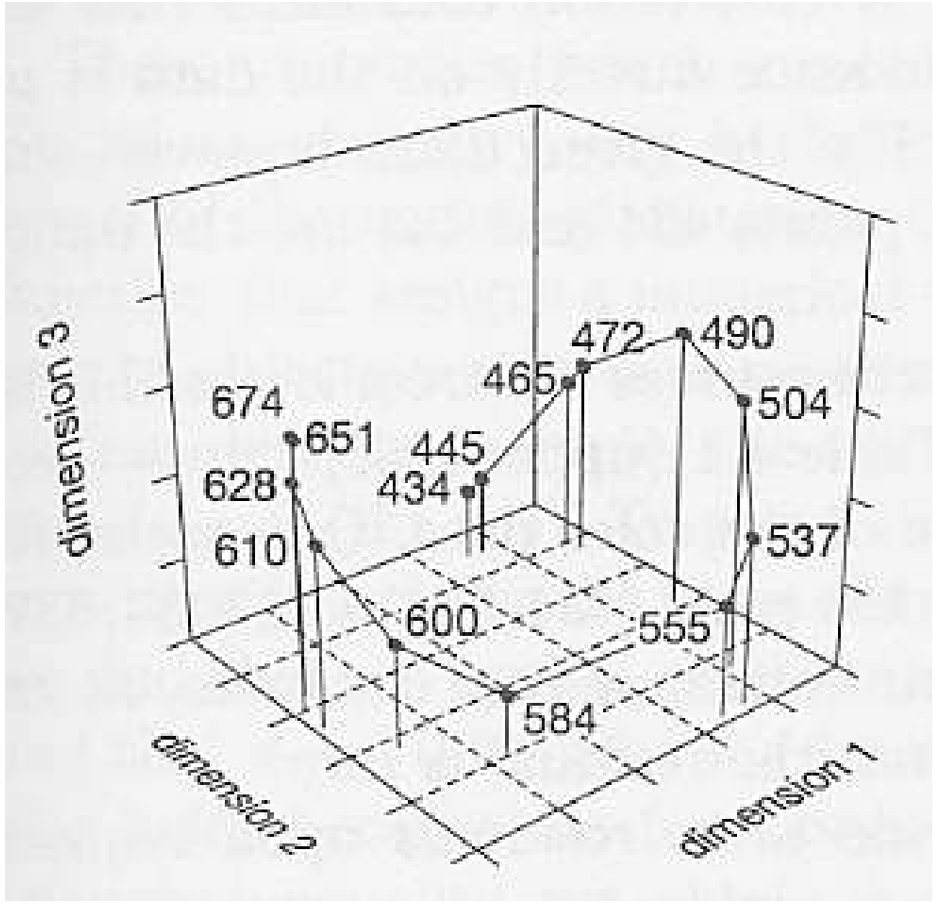
nm	434	445	465	472	490	504	537	555	584	600	610	628	651	674
434	–	.14	.17	.38	.22	-.73	-1.07	-1.21	-.62	-.06	.42	.38	.28	.26
445	.86	–	.25	.11	-.05	-.75	-1.09	-.68	-.35	-.04	.44	.65	.55	.53
465	.42	.50	–	.08	-.32	-.57	-.47	-.06	.00	-.32	.17	.12	.91	.82
472	.42	.44	.81	–	.12	-.36	-.26	.15	.00	-.11	.00	.33	.23	1.03
490	.18	.22	.47	.54	–	-.07	.08	.48	.40	.00	.22	.17	.07	.00
504	.06	.09	.17	.25	.61	–	.31	.28	.45	.68	.01	.00	.00	-.15
537	.07	.07	.10	.10	.31	.62	–	.13	.35	.09	.31	.00	.00	-.75
555	.04	.07	.08	.09	.26	.45	.73	–	-.05	.17	-.09	-.22	-.32	-.34
584	.02	.02	.02	.02	.07	.14	.22	.33	–	-.05	-.01	-.06	-.16	-.18
600	.07	.04	.01	.01	.02	.08	.14	.19	.58	–	.21	.07	-.39	-.40
610	.09	.07	.02	.00	.02	.02	.05	.04	.37	.74	–	-.08	-.13	-.11
628	.12	.11	.01	.01	.01	.02	.02	.03	.27	.50	.76	–	-.03	-.16
651	.13	.13	.05	.02	.02	.02	.02	.02	.20	.41	.62	.85	–	-.11
674	.16	.14	.03	.04	.00	.01	.00	.02	.23	.28	.55	.68	.76	–

Similarities of colors with different wavelengths (lower half, Ekman 1954) and residuals of 1D MDS representation (upper half) [B 4.1].

Color similarities

- The 14 colors where then projected by MDS (trying to preserve similarities) into 2D and 3D representations. The 2D representation shows that the red-violet (wavelength 434 nm) is perceived quite similar to blue-violet (wavelength 674 nm)





Ordinal MDS representations for color proximities in 2D and 3D [B 4.1, 4.3]

Multidimensional scaling (MDS)

- More formally, an MDS algorithm is given the original distances p_{ij} (called *proximities)* between data points *i* and *j*
- MDS algorithm then tries to find a low-dimensional (usually 2-3D) representation *X* for the points (*X* is just used to denote the Euclidean coordinates of the projected data points)
- More formally, MDS tries to find representation *X* that minimizes the error function (called *stress*, by convention)

$$\sigma_r = \sum_{i<j} \left(f(p_{ij}) - d_{ij}(X)\right)^2$$

where $d_{ij}(X)$ is the Euclidean distance between the data points *i* and *j* in representation *X*; and *f* is a function that defines the MDS model (next slide).

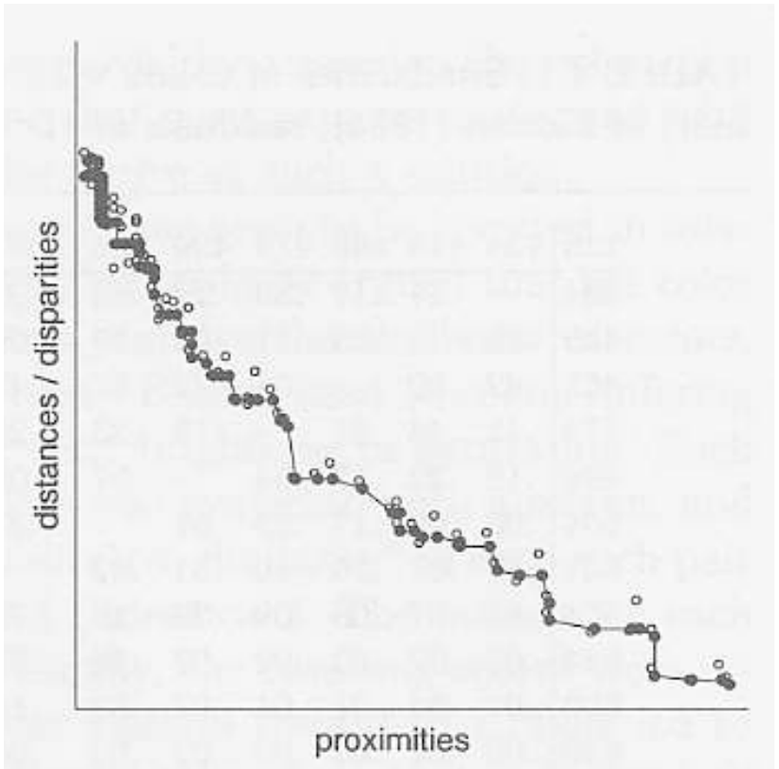
Multidimensional scaling (MDS)

$$\sigma_r = \sum_{i<j} \left(f(p_{ij}) - d_{ij}(X)\right)^2$$

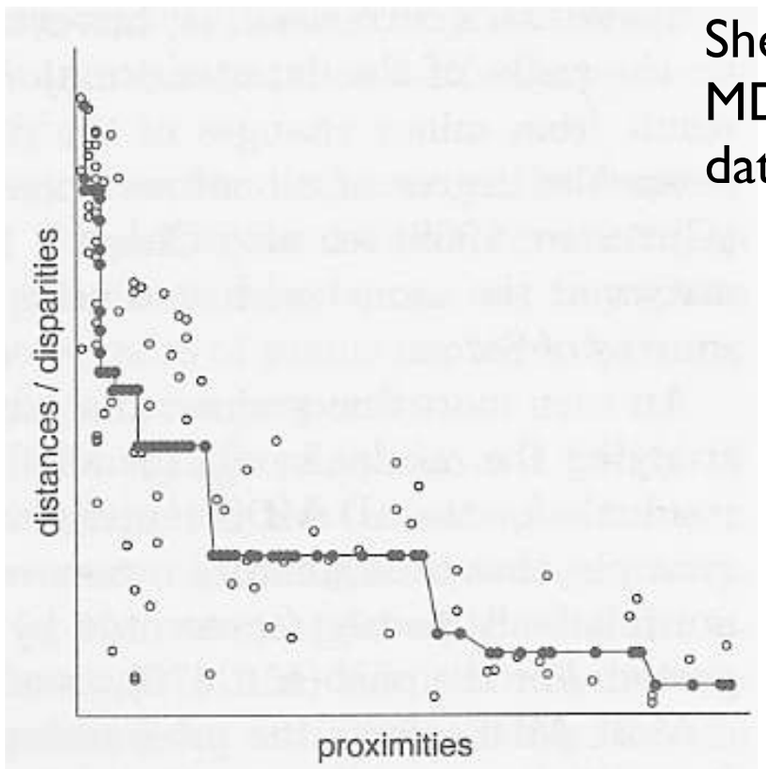
- The choice of *f* defines the MDS model. For example:
  - $f(p_{ij})=p_{ij}$ - absolute MDS (linear model, = PCA)
  - $f(p_{ij})=b\ p_{ij}$ - ratio MDS (linear model)
  - $f(p_{ij})=a+b\ p_{ij}$ - interval MDS (linear model)
  - $f(p_{ij})=a+b\ \log p_{ij}$ - useful in psychology
  - $f(p_{ij})$ is any monotonically increasing function (*ordinal* or *nonmetric MDS*) - this would be the most important special case of MDS
- The parameters of *f* (like *a* and *b* above) are optimized at the same time as the representation *X* (the details of the optimization algorithms is outside the scope of this course)
- It is conventional to denote the "transformed proximities", or "approximate distances", by d-hats, $\hat{d}_{ij} = f(p_{ij})$.

Shepard diagram

- There are two classical visualizations of MDS: *Shepard diagram* (shows the goodness of fit) and *Scree plot* (shows optimal dimensionality of the data)
- *Shepard diagram* shows the low-dimensional distances d_{ij} (white circles) and the target disparities $f(p_{ij})$ (filled circles) as a function of the original high-dimensional proximities p_{ij}.
- Scree plot shows the MDS cost function (stress) as a function of the number of dimensions



2D MDS



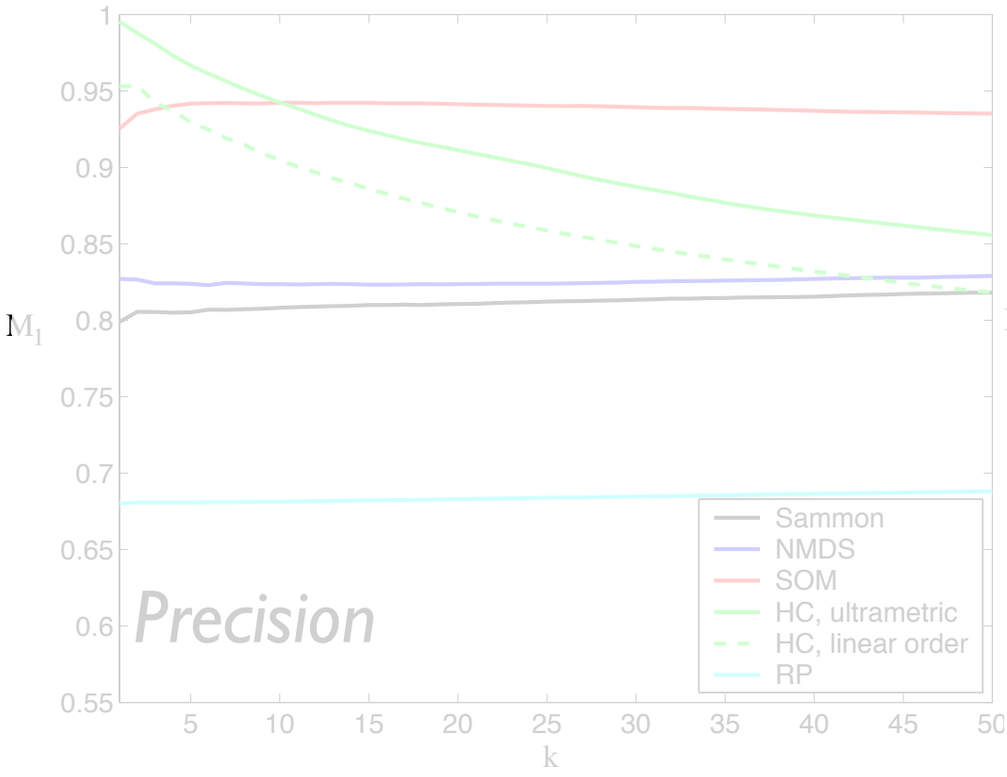
1D MDS

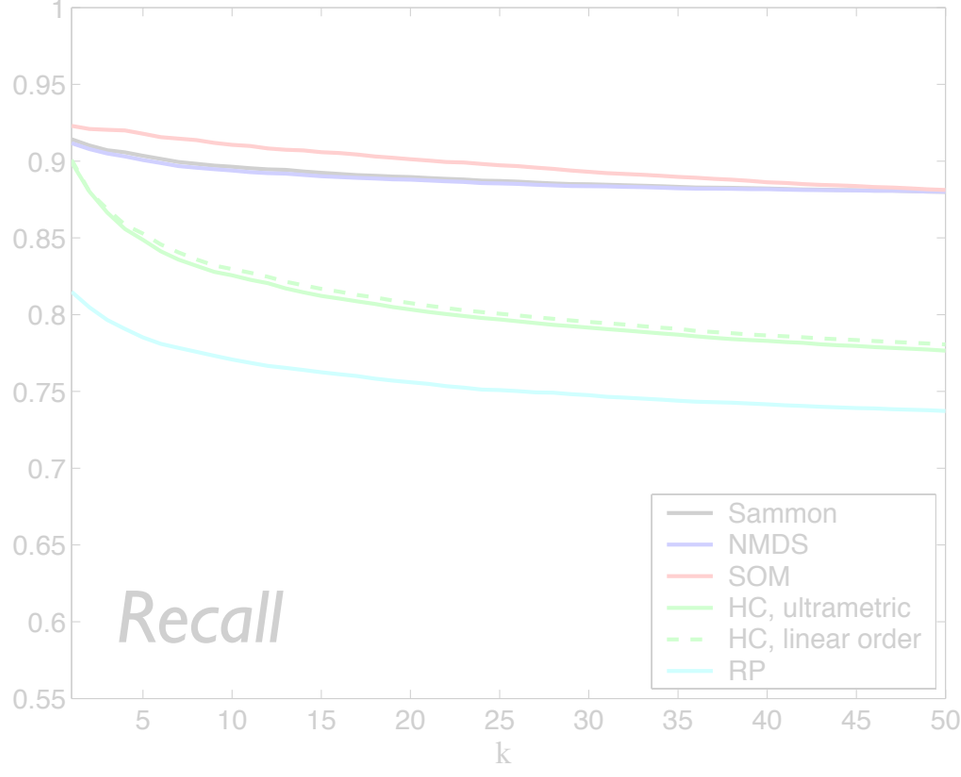
Shepard diagrams of 2D and 1D MDS projections of the color data.

- Each point is a pairwise comparison of two colors
- Horizontal axis positions are original similarities (not distances)
- Filled circles are transformations of similarities to distances,
- Empty circles are distances in the low-dimensional space.

Performance of MDS

- MDS is tries to preserve the large distances at the expense of small ones, hence, it can "collapse" some small distances on the expense of preserving large distances
- *Next lecture:* A projection is *trustworthy (precision)* if *k* closest neighbors of a sample on the projection are also close by in the original space. A projection *preserves the original neighborhoods* (*recall*) if all *k* closest neighbors of a sample in the original space are also close by in the projection.





Figures are from Kaski, Nikkilä, Oja, Venna, Törönen, Castrén, *Trustworthiness and metrics in visualizing similarity of gene expression*, BMC Bioinformatics 2003, 4:48.

Precision and recall as a function of the neighborhood size *k* for a yeast data set. Non-metric (ordinal) MDS (NMDS) is shown in blue. Larger precision and recall is better.

Performance of MDS

- Next lecture: Relatively better recall, worse precision
- MDS algorithms typically have running times of the order $O(N^2)$, where N is the number of data items.
- This is not very good: $N=1,000$ data items are ok, but $N=1,000,000$ is getting very slow.
  - Some solutions: use landmark points (i.e., use MDS only on a subset of data points and place the remaining points according to those, use MDS on cluster centroids etc.), use some other algorithm or modification of MDS.
- MDS is not guaranteed to find the global optimum of the stress (cost) function, nor it is guaranteed to converge to the same solution at each run (many of the MDS algorithms are quite good and reliable, though)

Performance of MDS

- Relatively better recall, worse precision
- MDS algorithms typically have running times of the order $O(N^2)$, where N is the number of data items.
- This is not very good: $N=1,000$ data items are ok, but $N=1,000,000$ is getting very slow.
- Some solutions: use landmark points (i.e., use MDS only on a subset of data points and place the remaining points according to those, use MDS on cluster centroids etc.), use some other algorithm or modification of MDS.
- MDS is not guaranteed to find the global optimum of the stress (cost) function, nor it is guaranteed to converge to the same solution at each run (many of the MDS algorithms are quite good and reliable, though)

Performance of MDS

- Relatively better recall, worse precision
- MDS algorithms typically have running times of the order $O(N^2)$, where N is the number of data items.
- This is not very good: $N=1,000$ data items are ok, but $N=1,000,000$ is getting very slow.
- Some solutions: use landmark points (i.e., use MDS only on a subset of data points and place the remaining points according to those, use MDS on cluster centroids etc.), use some other algorithm or modification of MDS.
- MDS is not guaranteed to find the global optimum of the stress (cost) function, nor it is guaranteed to converge to the same solution at each run (many of the MDS algorithms are quite good and reliable, though)

Classical Metric Multidimensional Scaling

First major steps by Young and Householder (1938):
Classical metric MDS is defined as linear generative model

$$\mathbf{x} = W\boldsymbol{\xi} \text{ with } W \in \mathbb{R}^{N\times M} \text{ and } W^\top W = \mathbf{I}_M$$

where the observed data **X** and the latent variables are assumed to be centered.
Pairwise affinities given by scalar products $S_{ij} = \langle \mathbf{x}^i, \mathbf{x}^j \rangle$

Gram matrix: $S = [S_{ij}]_{1\le i,j\le n} = \mathbf{X}^\top \mathbf{X} = (W\boldsymbol{\Xi})^\top (W\boldsymbol{\Xi})$

$= \boldsymbol{\Xi}^\top W^\top W \boldsymbol{\Xi} = \boldsymbol{\Xi}^\top \boldsymbol{\Xi}$

Find solution by eigenvalue decomposition of Gram matrix S:

$S = U\Lambda U^\top = (\Lambda^{1/2}U^\top)^\top(\Lambda^{1/2}U^\top)$ U = n x n orthonormal matrix
Λ= diagonal eigenvalue matrix

$\rightarrow \hat{\boldsymbol{\Xi}} = \mathbf{I}_{M\times n}\Lambda^{1/2}U^\top$ eigenvalues sorted in descending order

Classical Metric MDS and PCA

- PCA needs data coordinates **X**, not needed in metric MDS
- PCA decomposes covariance, which is proportional to $\mathbf{X}\mathbf{X}^\top$

$$\widehat{C_{XX}} \propto \mathbf{X}\mathbf{X}^\top = V\Lambda_{\mathrm{PCA}}V^\top \rightarrow \widehat{\mathbf{\Xi}}_{\mathrm{PCA}} = \mathbf{I}_{M\times n}V^\top\mathbf{X}$$

- metric MDS decomposes Gram matrix

$$S = \mathbf{X}^\top\mathbf{X} = U\Lambda_{\mathrm{MDS}}U^\top \rightarrow \widehat{\mathbf{\Xi}}_{\mathrm{MDS}} = \mathbf{I}_{M\times n}\Lambda_{\mathrm{MDS}}^{1/2}U^\top$$

- It is easy to show that

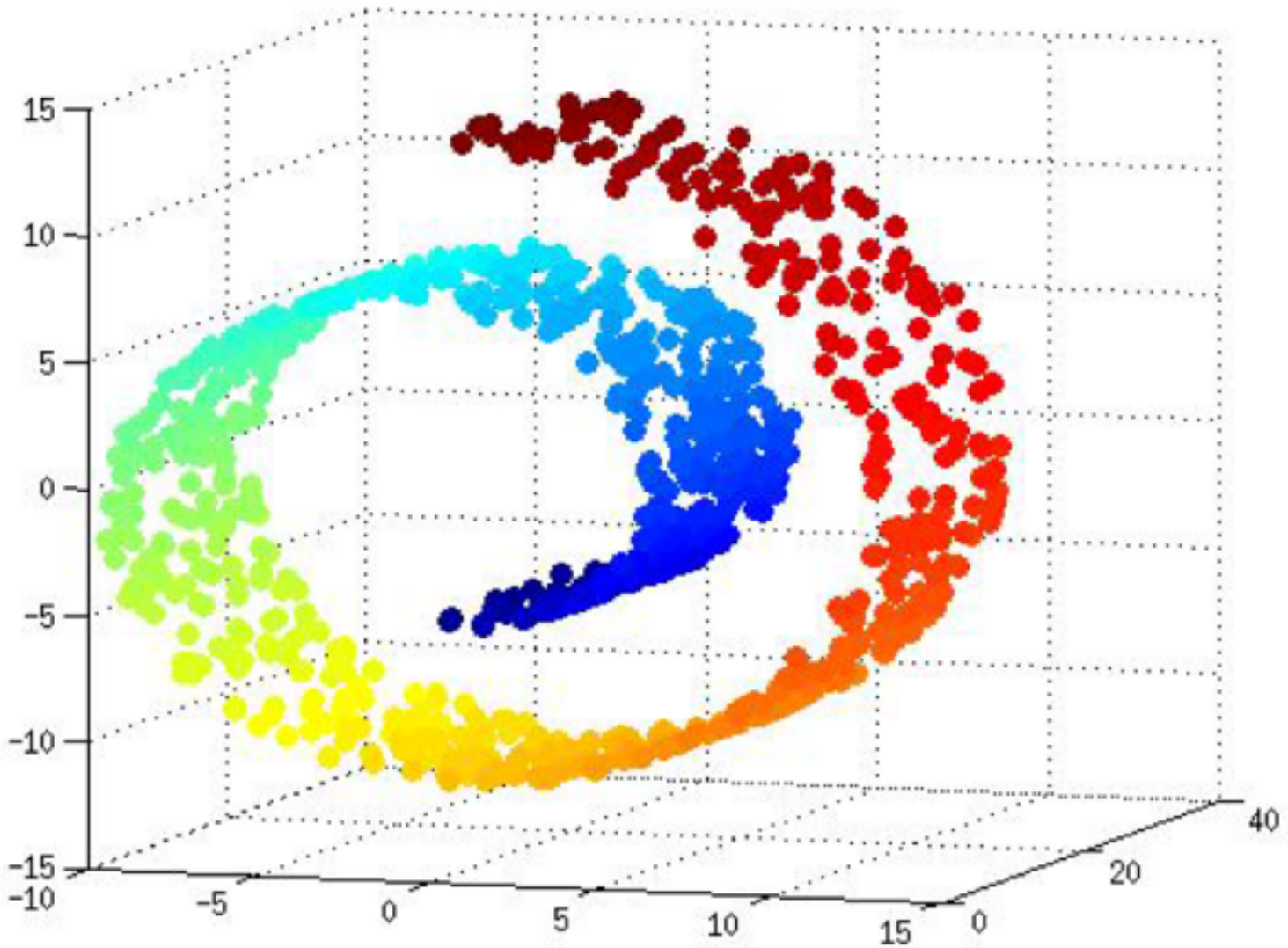
$$\widehat{\mathbf{\Xi}}_{\mathrm{PCA}} = \widehat{\mathbf{\Xi}}_{\mathrm{MDS}}$$
$$\mathbf{I}_{M\times n}V^\top\mathbf{X} = \mathbf{I}_{M\times n}\Lambda_{\mathrm{MDS}}^{1/2}U^\top$$

- To do so, replace **X** by its singular value decomposition

$$\mathbf{X} = V\Sigma U^\top$$

Example Data

swiss roll: 1000 three-dimensional samples



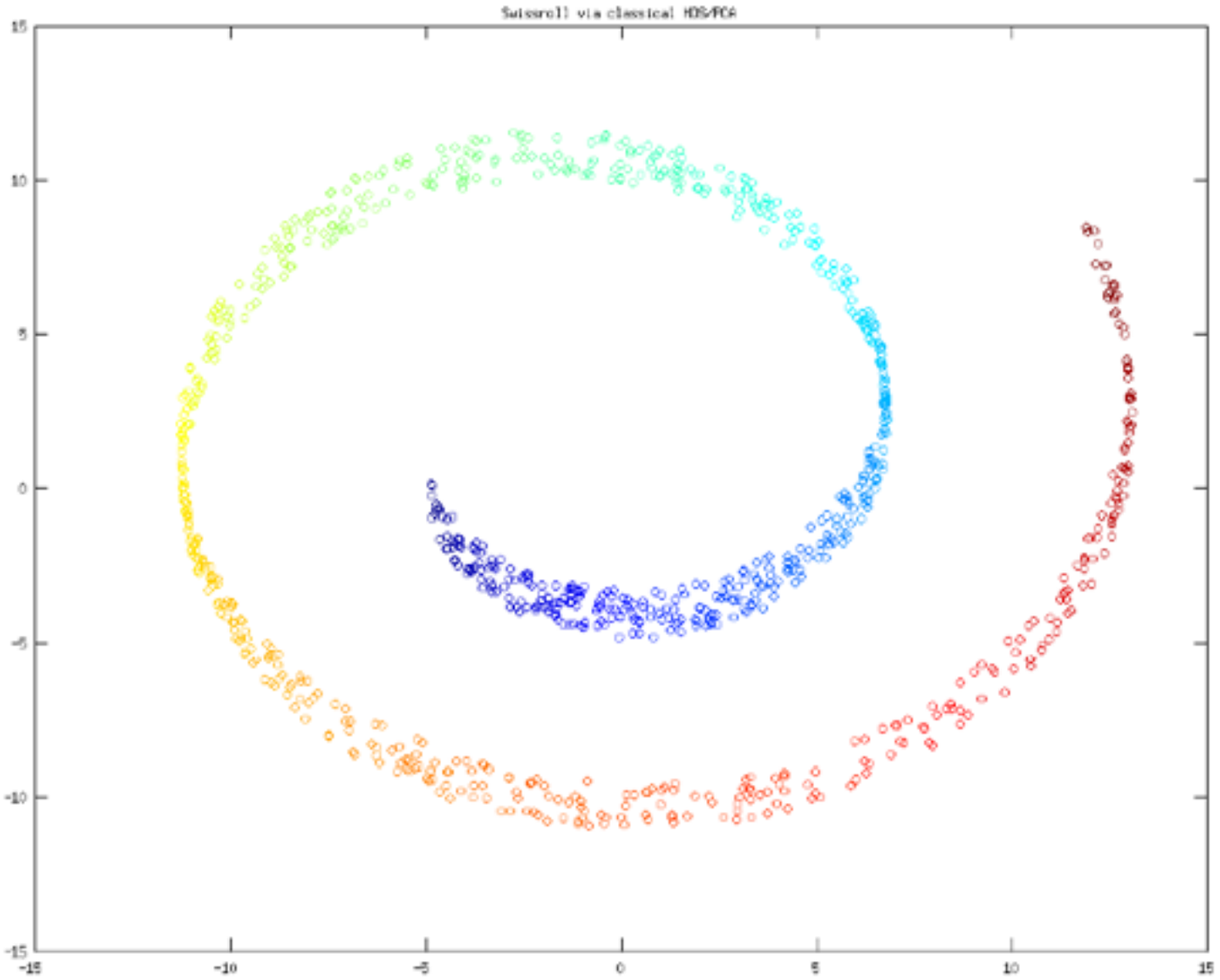
MDS

Pairwise Euclidean distances:

$$\mathrm{char}_{\mathcal{X}}(\mathbf{X}, \mathbf{x}) = (d_{\mathcal{X}}(\mathbf{x}^1, \mathbf{x}), \ldots, d_{\mathcal{X}}(\mathbf{x}^n, \mathbf{x}))$$

$$\mathrm{char}_{\mathcal{E}}(\mathbf{X\Xi}, (\mathbf{x}, \boldsymbol{\xi})) = (d_{\mathcal{E}}(\boldsymbol{\xi}^1, \boldsymbol{\xi}), \ldots, d_{\mathcal{E}}(\boldsymbol{\xi}^n, \boldsymbol{\xi}))$$

error = stress (least squared error (LSE))



Some MDS Variants

- **Nonmetric MDS** (Shepard 1962) and (Kruskal 1964) focuses on rank information ranks of closenesses between points instead of the specific interpoint distances:
- Proximities can be transformed to distances by $f(\delta(\mathbf{x}^i, \mathbf{x}^j)) \approx d_{\mathcal{X}}(\mathbf{x}^i, \mathbf{x}^j)$ with a monotonic transformation f

- Optimization done by minimizing

$$E_{\mathrm{nMDS}} = \sqrt{\frac{1}{a}\sum_{ij} w_{ij}(d_{\mathcal{X}}(\mathbf{x}^i, \mathbf{x}^j) - d_{\mathcal{E}}(\boldsymbol{\xi}^i, \boldsymbol{\xi}^j))^2}$$

where the normalizing constant is $a = \sum_{i,j}^{n} w_{ij} d_{\mathcal{X}}(\mathbf{x}^i, \mathbf{x}^j)$

Sammon Mapping

$$\sigma_r = \sum_{i<j} \frac{(p_{ij} - d_{ij}(X))^2}{p_{ij}}$$

- It is considered a non-linear approach as the projection cannot be represented as a linear combination of the original variables as possible in techniques such as principal component analysis.
- The minimization can be performed either by gradient descent. The number of iterations need to be experimentally determined and convergent solutions are not always guaranteed. Many implementations prefer to use the first Principal Components as a starting configuration.
- The Sammon mapping increases the importance of small distances and decreases the importance of large distances → nonlinear mapping

Sammon Mapping

$$\sigma_r = \sum_{i<j} \frac{(p_{ij} - d_{ij}(X))^2}{p_{ij}}$$

- It is considered a non-linear approach as the projection cannot be represented as a linear combination of the original variables as possible in techniques such as principal component analysis.
- The minimization can be performed either by gradient descent. The number of iterations need to be experimentally determined and convergent solutions are not always guaranteed. Many implementations prefer to use the first Principal Components as a starting configuration.
- The Sammon mapping increases the importance of small distances and decreases the importance of large distances → nonlinear mapping

Sammon Mapping

$$\sigma_r = \sum_{i<j} \frac{(p_{ij} - d_{ij}(X))^2}{p_{ij}}$$

- It is considered a non-linear approach as the projection cannot be represented as a linear combination of the original variables as possible in techniques such as principal component analysis.
- The minimization can be performed either by gradient descent. The number of iterations need to be experimentally determined and convergent solutions are not always guaranteed. Many implementations prefer to use the first Principal Components as a starting configuration.
- The Sammon mapping increases the importance of small distances and decreases the importance of large distances → nonlinear mapping

Summary

- How to project items into lower dimension when pairwise distance/similarity is known
- MDS, PCA try to preserve large distances
- No algorithm can generally preserve faithfully all features of the original data
- Next: SOM and CCA

References

K. Bunte, M. Biehl, and B. Hammer. **A general framework for dimensionality reducing data visualization using explicit mapping functions**. *Neural Computation*, 24(3):771–804, 2012.

J. A. Lee and M. Verleysen. ***Nonlinear dimensionality reduction***. Springer, New York; London, 2007.

David W. Scott. ***Multivariate Density Estimation: Theory, Practice, and Visualization (Wiley Series in Probability and Statistics)***. Wiley, 1 edition, September 1992. ISBN0471547700.

L. J. P. van der Maaten, E. O. Postma, and H. J. van den Herik. **Dimensionality Reduction: A Comparative Review**, 2008.

More literature on dimension reduction

- Additional reading on PCA: any book on matrix algebra
- Additional reading on MDS:
  - Borg, Kroenen, *Modern multidimensional scaling: theory and applications*. Springer 1997.
  - Buja, Swayne, Littman, Dean, *XGvis: interactive data visualization with multidimensional scaling,* 1998. (XGVis and GGobi are open source visualization tools that include MDS; MDS is also available, e.g., in SAS toolkit and GNU R [e.g., cmdscale and isoMDS in package MASS])
- NIPS 2005 tutorial by Saul, *Spectral methods for dimensionality reduction,* http://www.nips.cc/Conferences/2005/Tutorials/
- Jarkko Venna 2007, Academic Dissertation
- Lee & Verleysen, 2007. Nonlinear dimensionality reduction. Springer.
- Contents (today and Thursday):
  1. Dimension reduction methods: overview
  2. Principal component analysis (PCA)
  3. Multidimensional scaling (MDS)
  4. Self-organizing Maps
  5. CCA