Introduction

- The background
  - The data is high-dimensional
  - The desire of projecting those data onto a lower-dimensional subspace without losing important information regarding some characteristic of the original variables

- The methods
  - Accomplishing the reduction of dimensionality through variable selection, referred to as feature selection.
  - Accomplishing the reduction of dimensionality by creating a reduced set of linear or nonlinear transformations of the input variables, referred to as feature extraction.

Content

A. Linear dimensionality reduction
1. Principal component analysis (PCA)
2. Independent component analysis (ICA)
3. Factor analysis
4. Canonical variate analysis
5. Projection pursuit

B. Nonlinear dimensionality reduction
6. Polynomial PCA
7. Principal curves and surfaces
8. Kernel PCA
9. Multidimensional scaling
10. Nonlinear manifold learning
Principal component analysis (PCA)

- Principal component analysis (PCA) is also known as the singular value decomposition (SVD), the Karhunen-Loève transform, the Hotelling transform (1933), and the empirical orthogonal function (EOF) method.
- PCA reduces the dimension by finding a few orthogonal linear combinations (the PCs) of the original variables with the largest variance (PCA giảm số chiều của một số nhỏ các tổ hợp tuyến tính trực giao (PC) của các biến gốc với phương sai lớn nhất).
- The first PC, $s_1$, is the linear combination with the largest variance
  
  $s_1 = x^T w_1$, where coefficient vector $w_1 = (w_{11}, ..., w_{1p})^T$ solves
  
  $w_1 = \arg \max_{\|w\|=1} \text{Var}(x^T w)$

- The second PC, $s_2$, is the linear combination with the second largest variance and orthogonal to the first PC, and so on. There are as many PCs as the number of the original variables.

- The first several PCs may explain most of the variance, so that the rest can be disregarded with minimal loss of information.

- As variance depends on the variable scale, first standardize each variable to have mean zero and standard deviation one. Assuming a standardized data with the empirical covariance matrix

  $\Sigma_{p \times p} = \frac{1}{n} XX^T$

- We can use the spectral decomposition theorem to write $\Sigma$ as

  $\Sigma = U \Lambda U^T$

  where $\Lambda = \text{diag}(\lambda_1, ..., \lambda_p)$ is the diagonal matrix of the ordered eigenvalues $\lambda_1 \leq \cdots \leq \lambda_p$, and $U$ is a $p \times p$ orthogonal matrix containing the eigenvectors.

- By plotting the cumulative proportions in as a function of $k$, one can select the appropriate number of PCs to keep in order to explain a given percentage of the overall variation.

- An alternative way to reduce the dimension of a dataset using PCA: Instead of using the PCs as the new variables, this method uses the information in the PCs to find important variables in the original dataset.

- Another way: The number of PCs to keep is determined by first fixing a threshold $\lambda_0$, then only keeping the eigenvectors (at least four) such that their corresponding eigenvalues are greater than $\lambda_0$ (Jolliffe, 1972, 1973).
Principal component analysis

**Example**

- Nutritional data from 961 food items. The nutritional components of each food item are given by seven variables: fat (grams), food energy (calories), carbohydrates (grams), protein (grams), cholesterol (milligrams), weight (grams), and saturated fat (grams).
- PCA of the transformed data yields six principal components ordered by decreasing variances. The first three principal components, PC1, PC2, and PC3, which account for more than 83% of the total variance.

<table>
<thead>
<tr>
<th>Food Component</th>
<th>PC1</th>
<th>PC2</th>
<th>PC3</th>
<th>PC4</th>
<th>PC5</th>
<th>PC6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fat</td>
<td>0.557</td>
<td>0.099</td>
<td>0.275</td>
<td>0.130</td>
<td>0.455</td>
<td>0.617</td>
</tr>
<tr>
<td>Food energy</td>
<td>0.536</td>
<td>0.357</td>
<td>-0.137</td>
<td>0.075</td>
<td>0.273</td>
<td>-0.697</td>
</tr>
<tr>
<td>Carbohydrates</td>
<td>-0.925</td>
<td>0.672</td>
<td>-0.568</td>
<td>-0.286</td>
<td>-0.157</td>
<td>0.344</td>
</tr>
<tr>
<td>Protein</td>
<td>0.245</td>
<td>-0.374</td>
<td>-0.839</td>
<td>0.599</td>
<td>-0.154</td>
<td>0.119</td>
</tr>
<tr>
<td>Cholesterol</td>
<td>0.253</td>
<td>-0.521</td>
<td>-0.926</td>
<td>-0.717</td>
<td>0.210</td>
<td>-0.003</td>
</tr>
<tr>
<td>Saturated fat</td>
<td>0.531</td>
<td>-0.019</td>
<td>0.261</td>
<td>-0.150</td>
<td>-0.791</td>
<td>0.022</td>
</tr>
<tr>
<td>Variance</td>
<td>2.649</td>
<td>1.330</td>
<td>1.020</td>
<td>0.680</td>
<td>0.267</td>
<td>0.055</td>
</tr>
<tr>
<td>% Total Variance</td>
<td>44.1</td>
<td>22.2</td>
<td>17.0</td>
<td>11.3</td>
<td>4.4</td>
<td>0.9</td>
</tr>
</tbody>
</table>

---

Independent component analysis (ICA)

- A method that seeks linear projections, not necessarily orthogonal to each other, but nearly statistically independent as possible.
- The random variables $x = \{x_1, ..., x_p\}$ are uncorrelated, if for $\forall i \neq j$, $1 \leq i, j \leq p$, we have $\text{Cov}(x_i, x_j) = 0$. Independence requires that the multivariate probability density function factorizes, $f(x_1, ..., x_p) = f_1(x_1) ... f_p(x_p)$.

  independence $\Rightarrow$ uncorrelated, but uncorrelated $\not\Rightarrow$ independence.

- The noise-free ICA model for the $p$-dimensional random vector $x$ seeks to estimate the components of the $k$-dimensional vector $s$ and the $p \times k$ full column rank mixing matrix $A$.

  $$(x_1, ..., x_p)^T = A_{p \times k}(s_1, ..., s_k)^T$$

  such that the components of $s$ are as independent as possible, according to some definition of independence.

- The noisy ICA contains an additive random noise component,

  $$(x_1, ..., x_p)^T = A_{p \times k}(s_1, ..., s_k)^T + (u_1, ..., u_p)^T$$

  Estimation of such models is still an open research issue.

- In contrast with PCA, the goal of ICA is not necessarily dimension reduction. To find $k < p$ independent components, one needs to first reduce the dimension of the original data $p$ to $k$, by a method such as PCA.

- No order among the PCs of ICA. ICA can be considered a generalization of the PCA and the PP (project pursuit) concepts.

- ICA is applied to many different problems, including exploratory data analysis, blind source separation, blind deconvolution, and feature extraction.
Independent component analysis (ICA)

Perform ICA

Perform ICA

Factor analysis (FA)

- A linear method, based on the second-order data summaries. **Factor analysis** assumes that the measured variables depend on some unknown, and often unmeasurable, common factors.
- Typical examples include variables defined as various test scores of individuals, as such scores are thought to be related to a common “intelligence” factor.
- The goal of FA is to uncover such relations, and thus can be used to reduce the dimension of datasets following the factor model.
- The zero-mean $p$-dimensional random vector $x_p \times 1$ with covariance matrix $\Sigma$ satisfies the $k$-factor model
  \[ x = \Lambda f + u \]
  where $\Lambda_{p \times k}$ is a matrix of constants, and $f_{k \times 1}$ and $u_{p \times 1}$ are the random common factors and specific factors, respectively.

Canonical variate analysis (CVA, Hotelling, 1936)

- A method for studying **linear relationships** between two vector variates, $X = (X_1, ..., X_r)^T$ and $Y = (Y_1, ..., Y_s)^T$ which have different dimensions.
- CVA seeks to replace the two sets of correlated variables, $X$ and $Y$, by $t$ pairs of new variables,
  \[(\xi_i, \omega_i), i = 1, ..., t; \quad t \leq \min(r,s)\]
  where
  \[ \xi_i = g_i^T X = \sum_{k=1}^{r} g_{ki} X_k, \quad \omega_i = h_i^T Y = \sum_{k=1}^{s} h_{ki} Y_k \]

Canonical variate and correlation analysis

**Least-squares optimality of CVA**

- The task
  \[(X_1, ..., X_r, Y_1, ..., Y_s)^T \rightarrow ((\xi_1, \omega_1), ..., (\xi_t, \omega_t))^T\]
  Linear projections by $(t \times r)$-matrix $G$ and $(t \times s)$-matrix $H$ with $(1 \leq t \leq \min(r,s))$:
  \[
  \xi = GX, \quad \omega = HY,
  \]
  Least-square error criterion, to find $v$, $G$, and $H$ so that
  \[ HY \approx v + GX \]
  to minimize
  \[ E((HY - v + GX)(HY - v + GX)^T) \]
  which measure how well we can reconstruct $X$ and $Y$ from pairs of $(\xi_i, \omega_i)$.
- The goal is
  To choose the best $v$, $G$, and $H$ in the least-square sense.
Projection Pursuit

**The motivation**
The desire to discover “interesting” low-dimensional (typically, one- or two-dimensional) linear projections of high-dimensional data.

**The origin**
The desire to expose specific non-Gaussian features (variously referred to as “local concentration,” “clusters of distinct groups,” “clumpiness,” or “clottedness”) of the data.

**The strategy**
1. Set up a projection index \( \mathcal{I} \) to judge the merit of a particular one or two-dimensional (or sometimes three-dimensional) projection of a given set of multivariate data.
2. Use an optimization algorithm to find the global and local extrema of that projection index over all \( m \)-dimensional projections of the data.

Projection Pursuit

**Projection Indexes**

- Projection indexes should be chosen to possess certain computational and analytical properties, especially that of affine invariance (location and scale invariance).
- A special case of PP occurs when the projection index is the **variance**. Maximizing the variance reduces PP to PCA, and the resulting projections are the leading principal components of \( \mathbf{X} \).
- Maximizing the variance is equivalent to minimizing the corresponding Gaussian log-likelihood; in other words, the projection is most interesting (in a variance sense) when \( \mathbf{X} \) is least likely to be Gaussian.

Typical PP:
- Cumulant-based index
- Polynomial-based indexes
- Shannon negentropy
- Optimizing the projection index

Introduction

- The linear projection methods can be extremely useful in discovering low-dimensional structure when the data actually lie in a linear (or approximately linear) lower-dimensional subspace \( \mathcal{M} \) (called a **manifold**) of input space \( \mathbb{R}^p \).
- What can we do if we know or suspect that the data actually lie on a low dimensional **nonlinear manifold**, whose structure and dimensionality are both assumed unknown?
- Dimensionality reduction \( \rightarrow \) Problem of **nonlinear manifold learning**.
- When a linear representation of the data is unsatisfactory, we turn to specialized methods designed to recover nonlinear structure. Even so, we may not always be successful.
- **Key ideas**: Generalizing linear multivariate methods. Note that, these equivalences in the linear case do not always transfer to the nonlinear case. (tổng quát hóa các pp tùy tính đa biến dů không luôn thành công).

Polynomial PCA

- How should we generalize PCA to the nonlinear case? One possibility is to transform the set of input variables using a quadratic, cubic, or higher degree polynomial, and then apply linear PCA.
- **Focused on the smallest** few eigenvalues for nonlinear dimensionality reduction.
- **Quadratic PCA**: the \( r \)-vector \( \mathbf{X} \) is transformed into an extended \( r \)-vector \( \mathbf{X}' \), where \( r = 2r + r(r - 1)/2 \), \( \mathbf{X} = (X_1, X_2) \rightarrow \mathbf{X}' = (X_1, X_2, X_1^2, X_2^2) \).
- **Some problems inevitably arise when using quadratic PCA**.
  - First, the variables in \( \mathbf{X} \) will not be uniformly scaled, especially for large \( r \), and so a standardization of all \( r \) variables may be desirable.
  - Second, the size of the extended vector \( \mathbf{X} \) for quadratic PCA increases quickly with increasing \( r \) when \( r = 10, r = 65 \), and when \( r = 20, r = 230 \).
Principal curves and surfaces

- Suppose $X$ is a continuous random $r$-vector having density $p_X$, zero mean, and finite second moments. Suppose further that the data observed on $X$ lie close to a smooth nonlinear manifold of low dimension.
- A principal curve (Hastie, 1984; Hastie and Stuetzle, 1989) is a smooth one-dimensional parameterized curve $f$ that passes through the "middle" of the data, regardless of whether the "middle" is a straight line or a nonlinear curve.
- A principal surface is a generalization of principal curve to a smooth two- (or higher-) dimensional curve.
- We use an analogue of least-squares optimality as the defining characteristic: we determine the principal curve or surface by minimizing the average of the squared distances between the data points and their projections onto that curve.

Content

A. Linear dimensionality reduction
   1. Principal component analysis (PCA)
   2. Independent component analysis (ICA)
   3. Factor analysis
   4. Canonical variate analysis
   5. Projection pursuit

B. Nonlinear dimensionality reduction
   6. Polynomial PCA
   7. Principal curves and surfaces
   8. Kernel PCA
   9. Multidimensional scaling
   10. Nonlinear manifold learning

Introduction

- A map including cities and towns → table where each cell shows the degree of "closeness" (or proximity, sự gần gũi) of a row city to a column city. Proximity could have different meanings: straight-line distance or as shortest traveling distance.
- Consider entities as objects, products, a nations, a stimulus, etc. and we can talk about proximity of any two entities as measures of association (e.g., absolute value of a correlation coefficient), confusion frequency (i.e., to what extent one entity is confused with another in an identification exercise), or measure of how alike (or how different), etc.
- Multidimensional scaling (MDS): [tái dùng bản đồ góc nhiều chiều] given a table of proximities of entities, reconstruct the original map of entities as closely as possible.
- MDS is a family of different algorithms, each designed to arrive at an optimal low-dimensional configuration for a particular type of proximity data.

Problem: Re-create the map that yielded the table of airline distances.

Airline distances (km) between 18 cities.
Examples of MDS applications

- **Marketing**: Derive “product maps” of consumer choice and product preference (e.g., automobiles, beer) so that relationships between products can be discerned.
- **Ecology**: Provide “environmental impact maps” of pollution (e.g., oil spills, sewage pollution, drilling-mud dispersal) on local communities of animals, marine species, and insects.
- **Molecular Biology**: Reconstruct the spatial structures of molecules (e.g., amino acids) using biomolecular conformation (3D structure). Interpret their interrelations, similarities, and differences. Construct a 3D “protein map” as a global view of the protein structure universe.
- **Social Networks**: Develop “telephone-call graphs,” where the vertices are telephone numbers and the edges correspond to calls between them. Recognize instances of credit card fraud and network intrusion detection.

Proximity matrices

- The focus on pairwise comparisons of entities is fundamental to MDS.
- The “closeness” of two entities is measured by a proximity measure, defined in a number of different ways.
  - A continuous measure of how physically close one entity is to another or a subjective judgment recorded on an ordinal scale, but where the scale is well-calibrated as to be considered continuous.
  - In perception study, proximity is not quantitative but a subjective rating of similarity (or dissimilarity) recorded on a pair of entities.
- “Closeness” of one entity to another could be measured by a small or large value. Importance is a monotonic relationship between the “closeness” of two entities.

Proximity matrices

- Collection of \( n \) entities. Let \( \delta_{ij} \) represent the dissimilarity of the \( i \)th entity to the \( j \)th entity. Consider \( m \) dissimilarities, \( m = \binom{n}{2} = \frac{1}{2} n(n - 1) \), and \((m \times m)\) proximity matrix

\[
\Delta = (\delta_{ij})
\]

The proximity matrix is usually displayed as a lower-triangular array of nonnegative entries, with the understanding that the diagonal entries are all zeroes and the matrix is symmetric: for all \( i, j = 1, \ldots, n \),

\[
\delta_{ij} \geq 0, \delta_{ii} = 0, \delta_{ji} = \delta_{ij}
\]

In order for a dissimilarity measure to be regarded as a metric distance, we require that \( \delta_{ij} \) satisfy the triangle inequality,

\[
\delta_{ij} \leq \delta_{ik} + \delta_{kj} \quad \text{for all } k
\]

Comparing protein sequences

*Optimal sequence alignment*

- About 100,000 different proteins in the human body.
- Proteins carry out important bodily functions: supporting cell structure, protecting against infection from bacteria and viruses, aiding movement, transporting materials (hemoglobin for oxygen), regulating control (enzymes, hormones, metabolism, insulin) of the body.
- “protein map”: how existing protein families relate to one another, structurally and functionally.
- Might be able to predict the functions of newly discovered proteins from their spatial locations and proximities to other proteins, where we would expect neighboring proteins to have very similar biochemical properties.
Comparing protein sequences

**Optimal sequence alignment**

- Key idea in computing the proximity of two proteins is amino acids can be altered by random mutations (đột biến) over a long period of evolution.
- Mutations can take various forms: deletion or insertion of amino acids, ... For an evolving organism to survive, structure/functionality of the most important segments of its protein would have to be preserved.
- Compute a similarity value between two sequences that have different lengths and different amino acid distributions.
- **Trick:** Align the two sequences so that as many letters in one sequence …ƒ“‡Dzƒ–…Ї†dz™‹–Š
  - Several methods for sequence alignment:
    - **Global alignment** aligns all the letters in entire sequences assuming that the two sequences are very similar from beginning to end;
    - **Local alignment** assumes that the two sequences are highly similar only over short segments of letters.

Alignment methods use dynamic programming algorithms as the primary tool. BLAST and FASTA are popular tools for huge databases.

- An “optimal” alignment if maximizing an alignment score. For example, an alignment score is the sum of a number of terms, such as identity (high positive), substitution (positive, negative or 0).
- **Substitution score:** “cost” of replacing one amino acid (aa). Scores for all 210 possible aa pairs are collected to form a (20 × 20) substitution matrix. One popular is BLOSUM62 (BLOcks Substitution Matrix), assuming no more than 62% of letters in sequences are identical (Henikoff, 1996).
- A “gap” (indel), an empty space (“-“), penalizes an insertion or a deletion of an aa. Two types of gap penalties, starting a gap and extending the gap.
- The alignment score \( s \) is the sum of the identity and substitution scores, minus the gap score.

**Example: Two hemoglobin chains**

Given \( n \) proteins, let \( s_{ij} \) be the alignment score between the \( i \)th and \( j \)th protein. We have \( \delta_{ij} = s_{\text{max}} - s_{ij} \), where \( s_{\text{max}} \) is the largest alignment score among all pairs. The proximity matrix is then given by \( \Delta = (\delta_{ij}) \).

- Compare the hemoglobin alpha chain protein HBA HUMAN having length 141 with the related hemoglobin beta chain protein HBB HUMAN having length 146.
- We would obtain different optimal alignments and alignment scores.

\[ \text{LSTACTYSLKLVKQAIAVAVAYM} \cdots \text{NLFLLFNLPPPSQIEK} \cdots \text{QYVBRPSLSEKLVAIAVNTY} \]

86 positive substitution scores (the 25 “+”s and the 61 identities). The alignment score is \( s = 259 \).

Comparing protein sequences

**Optimal sequence alignment**

String matching

**Edit distance**

- In pattern matching, we study the problem of finding a given pattern within a body of text. If a pattern is a single string, the problem is called string matching, used extensively in text-processing applications.
- A popular numerical measure of the similarity between two strings is edit distance (also called Levenshtein distance).
- The usual definition of edit distance is the fewest number of editing operations (insertions, deletions, substitutions) which would be needed to transform one string into the other.
- An insertion inserts a letter into the sequence, a deletion deletes a letter from the sequence, and a substitution replaces one letter in the sequence by another letter. Identities (or matches) are not counted in the distance measure. Each editing operation can be assigned a cost.
- Used to construct a phylogenetic tree — a diagram laying out a possible evolutionary history — of a single protein.
Suppose we are given \(n\) points \(X_1, \ldots, X_n \in \mathbb{R}^r\). From these points, we compute an \((n \times n)\) proximity matrix \(\Delta = (\delta_{ij})\) of dissimilarities, where

\[
\delta_{ij} = \|X_i - X_j\| = \left\{\sum_{k=1}^{r}(X_{ik} - X_{jk})^2\right\}^{1/2} \quad (2)
\]

Many kinds of distance can be considered: the Minkowski or \(L_p\) distance is given by

\[
\delta_{ij} = \left\{\sum_{k=1}^{p}(X_{ik} - X_{jk})^p\right\}^{1/p}
\]

- \(p = 1\): city-block or Manhattan distance,
- \(p = 2\): Euclidean distance.

From dissimilarities to principal coordinates

\[\text{Wish to find a } t \text{-dimensional representation, } Y_1, \ldots, Y_n \in \mathbb{R}^t \text{ (referred to as principal coordinates), of those } r \text{-dimensional points (with } t < r), \text{ such that the interpoint distances in } t \text{-space “match” those in } r \text{-space.}
\]

When dissimilarities are defined as Euclidean interpoint distances, this type of “classical” MDS is equivalent to PCA in that the principal coordinates are identical to the scores of the first \(t\) principal components of the \(\{X_i\}\).
Classical scaling and distance geometry

Airlines distances

Estimated and observed airline distances. The lefthand show the 2D solution and the right panels show the 3D solution. The top panels show the estimated distances plotted against the observed distances, and the bottom panels show the residuals from the fit (residual = estimated distance – observed distance) plotted against sequence number.

First three principal coordinates of the 18 cities in the airline distances example.

Distance scaling

- Given \( n \) items (or entities) and their dissimilarity matrix \( s, \Delta = (\delta_{ij}) \)
- Classical scaling problem is to find a configuration of points in a lower-dimensional space such that the interpoint distances \( \{d_{ij}\} \) satisfy \( d_{ij} \approx \delta_{ij} \)
- In distance scaling, this relationship is relaxed; we wish to find a suitable configuration for which \( d_{ij} \approx f(\delta_{ij}) \) where \( f \) is some monotonic function.
- The function \( f \) transforms the dissimilarities into distances. The use of “metric” or “nonmetric” distance scaling depends upon the nature of the dissimilarities.
- If the dissimilarities are quantitative we use metric distance scaling, whereas if the dissimilarities are qualitative we use nonmetric distance scaling. In the MDS literature, metric distance scaling is traditionally called metric MDS, nonmetric distance scaling is called nonmetric MDS.

Summary

- **Introduction**: Given a table of proximities, reconstruct the original map as closely as possible and the space dimension.
- **Proximity Matrices**: \( \Delta = (\delta_{ij}) \) measures “closeness” of pairwise objects, metric (triangle inequality)
- **Comparing Protein Sequences**
- **String Matching**: Edit distance
- **Classical Scaling**: require \( d_{ij} \approx \delta_{ij} \)
- **Distance Scaling**: \( d_{ij} \approx f(\delta_{ij}) \) where \( f \) is monotonic
  - **Metric Distance Scaling**: metric distance scaling
  - **Nonmetric Distance Scaling**: nonmetric distance scaling
Nonlinear manifold learning

**Manifold?**

- An ant at a picnic: The ant crawls all over the picnic items, as diminutive size, the ant sees everything on a very small scale as flat and featureless.
- A manifold can be thought of in similar terms, as a topological space that locally looks flat and featureless and behaves like Euclidean space.
- A manifold also satisfies certain topological conditions. A submanifold is just a manifold lying inside another manifold of higher dimension.

**Figure 16.6. Left panel: The S-curve, a two-dimensional S-shaped manifold embedded in three-dimensional space. Right panel: 2,000 data points randomly generated to be on the surface of the S-shaped manifold.**

Nonlinear manifold learning

- Many exciting new algorithmic techniques: Isomap, Local Linear Embedding, Laplacian Eigenmap, and Hessian Eigenmap. They aim to recover the full low-dimensional representation of an unknown nonlinear manifold \( M \).
- Having different philosophies for recovering nonlinear manifolds, each method consists of a three-step approach.
  1. Incorporating *neighborhood information* from each data point to construct a weighted graph having the data points as vertices.
  2. Taking the weighted neighborhood graph and *transforming* it into suitable input for the embedding step.
  3. Computing an \((n \times n)\)-eigenequation (*embedding step*).
- Manifold learning involves concepts from differential geometry: What means a *manifold* and what means to be *embedded* in a higher-dimensional space?

If a topological manifold \( M \) is continuously differentiable to any order (i.e., \( M \in C^\infty \)), we call it a *smooth* (or *differentiable*) manifold.

**Riemannian manifold** \( (M, d^M) \) is smooth manifold \( M \) with a metric \( d^M \). We take \( d^M \) to be a manifold metric defined by

\[
d^M(y, y') = \inf_c \{ L(c) | c \text{ is a curve in } M \text{ which joins } y \text{ and } y' \},
\]

where \( y, y' \in M \) and \( L(c) \) is the arc-length of the curve \( c \). Thus, \( d^M \) finds the shortest curve (or *geodesic, do dac*) between any two points on \( M \), and the arc-length of that curve is the *geodesic distance* between the points.

**Data assumption:** Finitely many data points, \( \{y_i\} \), are randomly sampled from a smooth \( t \)-dimensional manifold \( M \) with metric given by geodesic distance.

**ISOMAP**

- The *isometric feature mapping* (or ISOMAP) algorithm (Tenenbaum et al., 2000) assumes that the smooth manifold \( M \) is a convex region of \( \mathbb{R}^t \) \((t \ll r)\) and that the embedding \( \psi : M \rightarrow \mathcal{X} \) is an isometry.
- This assumption has two key ingredients:
  - **Isometry:** The geodesic distance is invariant under the map \( \psi \). For any pair of points on the manifold, \( y, y' \in M \), the geodesic distance between those points equals the Euclidean distance between their corresponding coordinates, \( x, x' \in \mathcal{X} \); i.e.,
    \[
d^M(y, y') = \|x - x'\|_\mathcal{X}
\]
    where \( y = \varphi(x) \) and \( y' = \varphi(x') \).
  - **Convexity:** The manifold \( M \) is a convex subset of \( \mathbb{R}^r \).
- ISOMAP regards \( M \) as a convex region that may have been distorted in any of a number of ways (e.g., by folding or twisting), e.g., Swiss roll.

*isometry is a distance-preserving map between metric spaces.*
Nonlinear manifold learning
Three steps of ISOMAP

1. Neighborhood graph.
   - Fix either an integer \( K \) or an \( \epsilon > 0 \). Calculate the distances,
     \[ d^M_{ij} = d^X(x_i, x_j) = \|x_i - x_j\|_X \]
   between all pairs of data points \( x_i, x_j \in \mathcal{X} \), \( i, j = 1, 2, \ldots, n \).
   Determine which data points are “neighbors” on the manifold \( M \) by connecting each point either to its \( K \) nearest neighbors or to all points lying within a ball of radius \( \epsilon \) of that point. Choice of \( K \) or \( \epsilon \) controls neighborhood size and also the success of ISOMAP.
   - We obtain weighted neighborhood graph \( \mathcal{G} = \mathcal{G}(\mathcal{V}, \mathcal{E}) \), where the set of vertices \( \mathcal{V} = \{x_1, \ldots, x_n\} \) are the input data points, and the set of edges \( \mathcal{E} = \{e_{ij}\} \) indicate neighborhood relationships between the points.

2. Compute graph distances
   - Estimate the unknown true geodesic distances, \( \{d^F_{ij}\} \), between pairs of points in \( M \) by graph distances, \( \{d^G_{ij}\} \), with respect to the graph \( \mathcal{G} \).
   - The graph distances are the shortest path distances between all pairs of points in the graph \( \mathcal{G} \). Points that are not neighbors of each other are connected by a sequence of neighbor-to-neighbor links.
   - An efficient algorithm for computing the shortest path between every pair of vertices in a graph is Floyd’s algorithm (Floyd, 1962), which works best with dense graphs (graphs with many edges).

3. Embedding via multidimensional scaling
   - Let \( D^G = \{d^G_{ij}\} \) be the symmetric \( (n \times n) \)-matrix of graph distances.
     Apply “classical” MDS to \( D^G \) to give the reconstructed data points in a \( t \)-dimensional feature space \( \mathcal{Y} \), so that the geodesic distances on \( M \) between data points is preserved as much as possible:
     - Form the “doubly centered,” symmetric, \( (n \times n) \)-matrix of squared graph distances.
     - The embedding vectors \( \{\tilde{y}_i\} \) are chosen to minimize \( \|A^G_n - A^G_{nt}\| \)
       where \( A^G_n = -\frac{1}{2}\mathbf{H} \mathbf{S}^G \mathbf{H} \) and \( A^G_{nt} = -\frac{1}{2}\mathbf{H} \left( [d^G_{ij}]^2 \right) \mathbf{H} \) and \( d^G_{ij} = \|y_i - y_j\| \).
     - The graph \( \mathcal{G} \) is embedded into \( \mathcal{Y} \) by the \( (t \times n) \)-matrix
       \[ \mathbf{Y} = (\tilde{y}_1, \ldots, \tilde{y}_n) = (\sqrt{\lambda_1} v_1, \ldots, \sqrt{\lambda_t} v_t) \]