Dimensionality Reduction

- Turquoise slides: Alpaydin
- Numbered blue slides: Haykin, *Neural Networks: A Comprehensive Foundation*, Second edition, Prentice-Hall, Upper Saddle River:NJ, 1999.
- Unnumbered blue slides: None of the above.

Why Reduce Dimensionality?

- Reduces time complexity: Less computation
- Reduces space complexity: Less parameters
- Saves the cost of observing the feature
- Simpler models are more robust on small datasets
- More interpretable; simpler explanation
- Data visualization (structure, groups, outliers, etc) if plotted in 2 or 3 dimensions

Feature Selection vs Extraction

- Feature selection: Choosing $k<d$ important features, ignoring the remaining $d-k$

  Subset selection algorithms
- Feature extraction: Project the

  original x_i , $i=1,...,d$ dimensions to

  new $k<d$ dimensions, z_j , $j=1,...,k$

  Principal components analysis (PCA), linear discriminant analysis (LDA), factor analysis (FA)

Subset Selection

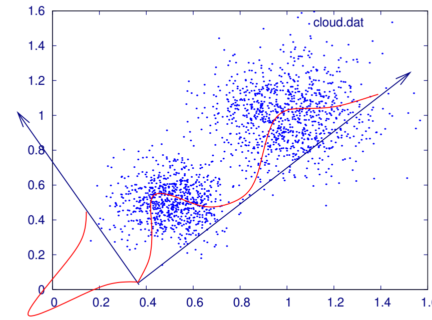
- There are 2^d subsets of d features
- Forward search: Add the best feature at each step
  - Set of features F initially Ø.
  - At each iteration, find the best new feature

    $j = \text{argmin}_i E (F \cup x_i)$
  - Add x_j to F if $E (F \cup x_j) < E (F)$
- Hill-climbing O(d^2) algorithm
- Backward search: Start with all features and remove one at a time, if possible.
- Floating search (Add k, remove l)

Principal Components Analysis (PCA)

Note: $\mathbf{Q}$ means eigenvector matrix of the covariance matrix, in Haykin slides.

Motivation



- How can we project the given data so that the variance in the projected points is maximized?

Eigenvalues/Eigenvectors

- For a square matrix $\mathbf{A}$, if a vector $\mathbf{x}$ and a scalar value λ exists so that

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0$$

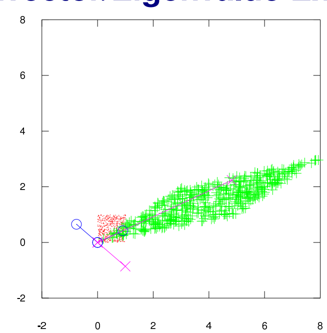
  then $\mathbf{x}$ is called an **eigenvector** of $\mathbf{A}$ and λ an **eigenvalue**.

- Note, the above is simply

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

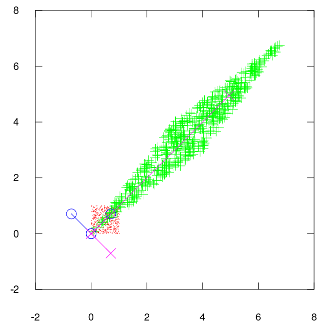
- An intuitive meaning is: $\mathbf{x}$ is the direction in which applying the linear transformation $\mathbf{A}$ only changes the magnitude of $\mathbf{x}$ (by λ) but not the angle.
- There can be as many as n eigenvector/eigenvalue for an $n \times n$ matrix.

Eigenvector/Eigenvalue Example



- Red: original data $\mathbf{x}$
- Green: projected data using $A = \begin{bmatrix} 3 & 5 \\ 2 & 1 \end{bmatrix}$.
- Blue: Eigenvectors $\mathbf{v}_1$=(0.91, 0.42), $\mathbf{v}_2$=(-0.76,0.65), $\lambda_1 = 5.3, \lambda_2 = -1.3$. Octave/Matlab code: `[V,Lamba]=eig(A)`
- Magenta: A times eigenvectors.

Eigenvector/Eigenvalue Example 2



- Red: original data $\mathbf{x}$
- Green: projected data using $A = \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix}$.
- Blue: Eigenvectors; Magenta: A times eigenvectors.
- A is a symmetric matrix, so eigenvectors are orthogonal.

Principal Components Analysis (PCA)

- Find a low-dimensional space such that when $\boldsymbol{x}$ is projected there, information loss is minimized.
- The projection of $\boldsymbol{x}$ on the direction of $\boldsymbol{w}$ is: $z = \boldsymbol{w}^T\boldsymbol{x}$
- Find $\boldsymbol{w}$ such that Var(z) is maximized

$$\begin{aligned}\text{Var}(z) &= \text{Var}(\boldsymbol{w}^T\boldsymbol{x}) = \text{E}[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})^2] \\ &= \text{E}[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})] \\ &= \text{E}[\boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{w}] \\ &= \boldsymbol{w}^T \text{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T]\boldsymbol{w} = \boldsymbol{w}^T \Sigma \boldsymbol{w}\end{aligned}$$

where Var($\boldsymbol{x}$)= $\text{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T] = \Sigma$

- Maximize Var(z) subject to $||\boldsymbol{w}||=1$

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \Sigma \mathbf{w}_1 - \alpha\left(\mathbf{w}_1^T\mathbf{w}_1 - 1\right)$$

$\Sigma \boldsymbol{w}_1 = \alpha \boldsymbol{w}_1$ that is, $\boldsymbol{w}_1$ is an eigenvector of Σ
Choose the one with the largest eigenvalue for Var(z) to be max

- Second principal component: Max Var(z_2), s.t., $||\boldsymbol{w}_2||=1$ and orthogonal to $\boldsymbol{w}_1$

$$\max_{\mathbf{w}_2} \mathbf{w}_2^T \Sigma \mathbf{w}_2 - \alpha\left(\mathbf{w}_2^T\mathbf{w}_2 - 1\right) - \beta\left(\mathbf{w}_2^T\mathbf{w}_1 - 0\right)$$

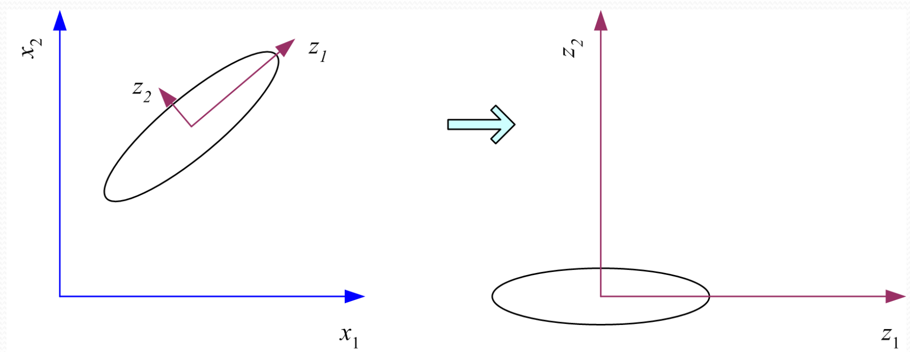
$\Sigma \boldsymbol{w}_2 = \alpha \boldsymbol{w}_2$ that is, $\boldsymbol{w}_2$ is another eigenvector of Σ
and so on.

What PCA does

$$\boldsymbol{z} = \mathbf{W}^T(\boldsymbol{x} - \boldsymbol{m})$$

where the columns of $\mathbf{W}$ are the eigenvectors of Σ, and $\boldsymbol{m}$ is sample mean

Centers the data at the origin and rotates the axes



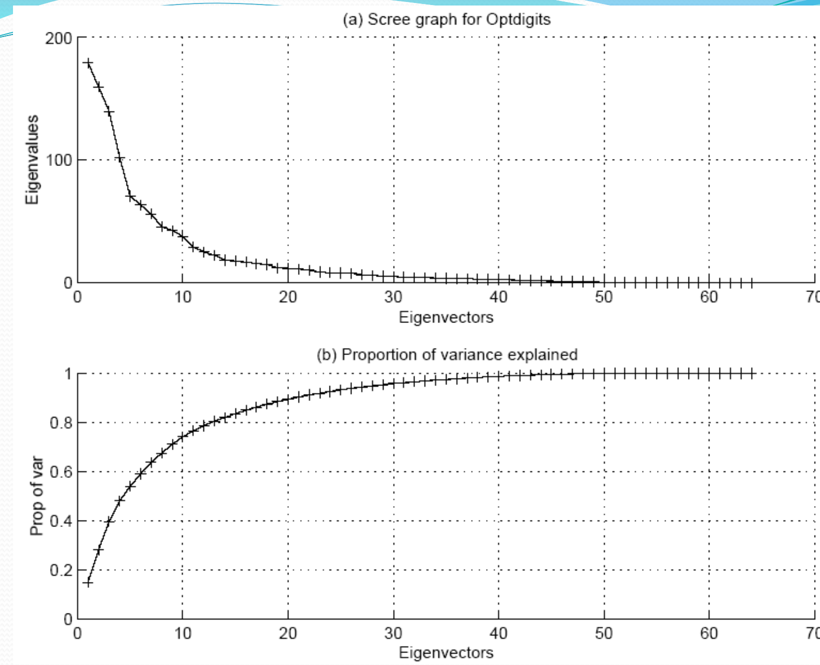
How to choose k ?

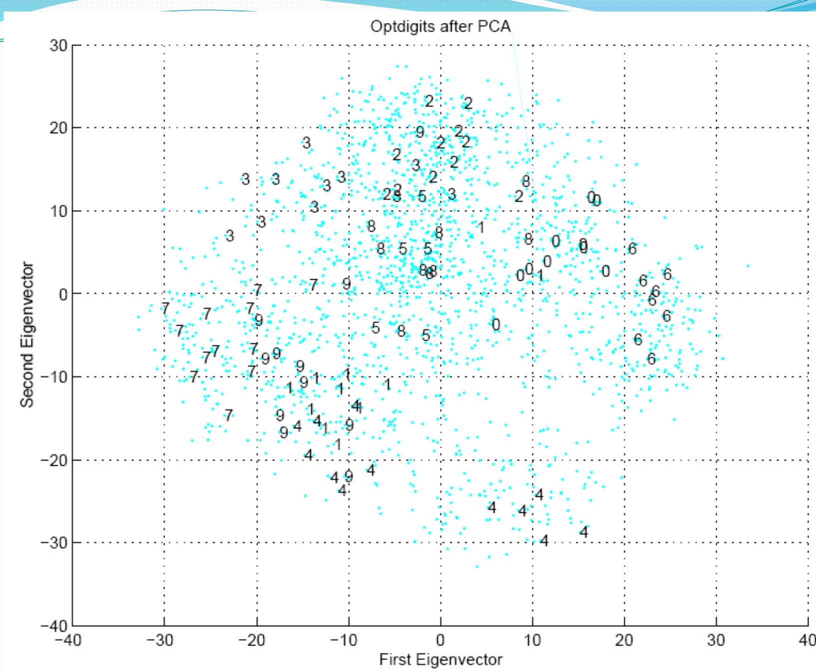
- Proportion of Variance (PoV) explained

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when λ_i are sorted in descending order

- Typically, stop at PoV>0.9
- Scree graph plots of PoV vs *k*, stop at "elbow"





PCA: Usage

- Project input $\mathbf{x}$ to the principal directions:

$$\mathbf{a} = \mathbf{Q}^T\mathbf{x}.$$

- We can also recover the input from the projected point $\mathbf{a}$:

$$\mathbf{x} = (\mathbf{Q}^T)^{-1}\mathbf{a} = \mathbf{Q}\mathbf{a}.$$

- Note that we don't need all m principal directions, depending on how much variance is captured in the first few eigenvalues: We can do dimensionality reduction.

PCA: Dimensionality Reduction

- **Encoding**: We can use the first l eigenvectors to encode $\mathbf{x}$.

$$[a_1, a_2, ..., a_l]^T = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_l]^T \mathbf{x}.$$

- Note that we only need to calculate l projections $a_1, a_2, ..., a_l$, where $l \leq m$.
- **Decoding**: Once $[a_1, a_2, ..., a_l]^T$ is obtained, we want to reconstruct the full $[x_1, x_2, ..., x_l, ..., x_m]^T$.

$$\mathbf{x} = \mathbf{Q}\mathbf{a} \approx [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_l][a_1, a_2, ..., a_l]^T = \hat{\mathbf{x}}.$$

Or, alternatively

$$\hat{\mathbf{x}} = \mathbf{Q}[a_1, a_2, ..., a_l, \underbrace{0, 0, ..., 0}_{m-l \text{ zeros}}]^T.$$

PCA: Total Variance

- The total variance of th em components of the data vector is

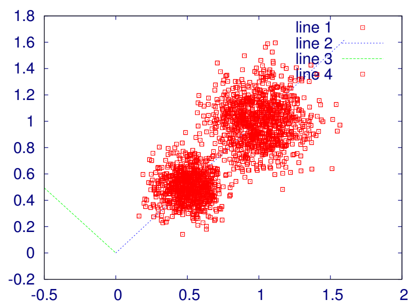
$$\sum_{j=1}^{m} \sigma_j^2 = \sum_{j=1}^{m} \lambda_j.$$

- The truncated version with the first l components have variance

$$\sum_{j=1}^{l} \sigma_j^2 = \sum_{j=1}^{l} \lambda_j.$$

- The larger the variance in the truncated version, i.e., the smaller the variance in the remaining components, the more accurate the dimensionality reduction.

PCA Example




```
inp=[randn(800,2)/9+0.5;randn(1000,2)/6+ones(1000,2)];
```

$$\mathbf{Q} = \begin{bmatrix} 0.70285 & -0.71134 \\ 0.71134 & 0.70285 \end{bmatrix}$$

$$\boldsymbol{\lambda} = \begin{bmatrix} 0.14425 & 0.00000 \\ 0.00000 & 0.02161 \end{bmatrix}$$

Factor Analysis

- Find a small number of factors $\boldsymbol{z}$, which when combined generate $\boldsymbol{x}$:

$$x_i - \mu_i = v_{i1}z_1 + v_{i2}z_2 + ... + v_{ik}z_k + \varepsilon_i$$

where z_j, j =1,...,k are the latent factors with

$E[\, z_j \,]=0$, $Var(z_j)=1$, $Cov(z_i, z_j)=0$, $i \neq j$,

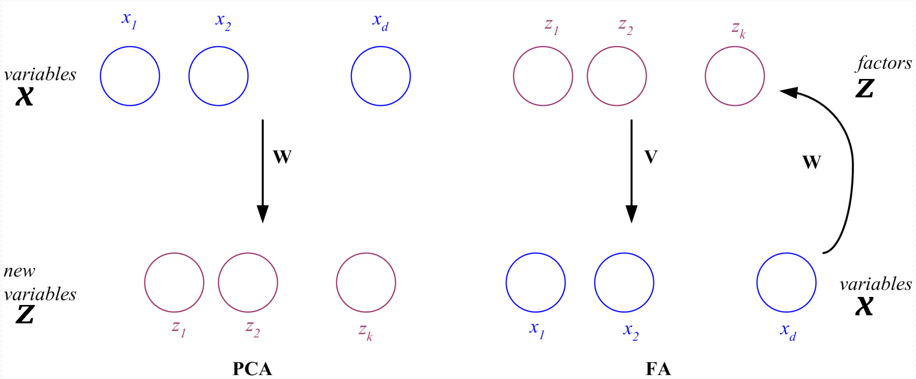
ε_i are the noise sources

$E[\, \varepsilon_i \,]= \psi_i$, $Cov(\varepsilon_i, \varepsilon_j) =0$, $i \neq j$, $Cov(\varepsilon_i, z_j) =0$,

and v_{ij} are the factor loadings

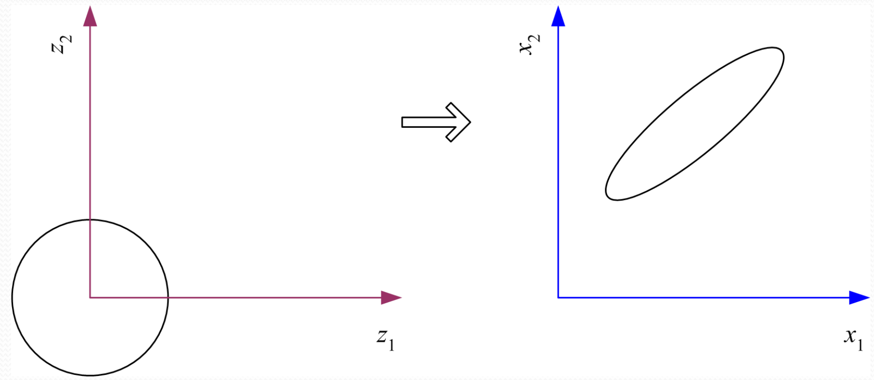
PCA vs FA

- PCA From $\boldsymbol{x}$ to $\boldsymbol{z}$ $\quad \boldsymbol{z} = \mathbf{W}^T(\boldsymbol{x} - \boldsymbol{\mu})$
- FA $\quad$ From $\boldsymbol{z}$ to $\boldsymbol{x}$ $\quad \boldsymbol{x} - \boldsymbol{\mu} = \mathbf{V}\mathbf{z} + \boldsymbol{\varepsilon}$



Factor Analysis

- In FA, factors z_j are stretched, rotated and translated to generate $\boldsymbol{x}$



Multidimensional Scaling

- Given pairwise distances between N points,

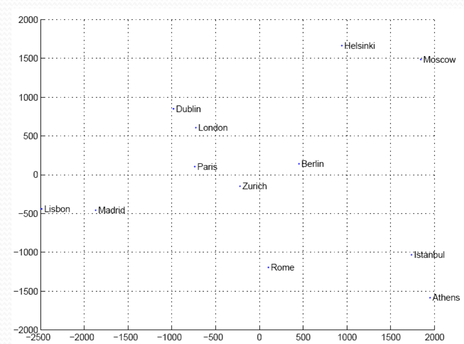
  d_{ij}, $i,j = 1,\dots,N$

  place on a low-dim map s.t. distances are preserved.
- $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{x} \mid \vartheta)$ $\quad$ Find ϑ that min Sammon stress

$$E(\theta \mid \mathcal{X}) = \sum_{r,s} \frac{\left(\left\|\mathbf{z}^r - \mathbf{z}^s\right\| - \left\|\mathbf{x}^r - \mathbf{x}^s\right\|\right)^2}{\left\|\mathbf{x}^r - \mathbf{x}^s\right\|^2}$$

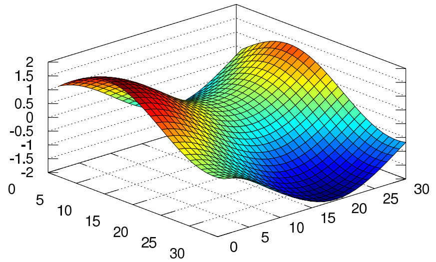
$$= \sum_{r,s} \frac{\left(\left\|\mathbf{g}(\mathbf{x}^r \mid \theta) - \mathbf{g}(\mathbf{x}^s \mid \theta)\right\| - \left\|\mathbf{x}^r - \mathbf{x}^s\right\|\right)^2}{\left\|\mathbf{x}^r - \mathbf{x}^s\right\|^2}$$

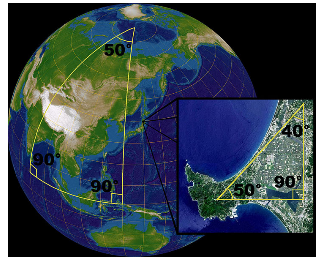
Map of Europe by MDS



Map from CIA – The World Factbook: http://www.cia.gov/

Manifolds

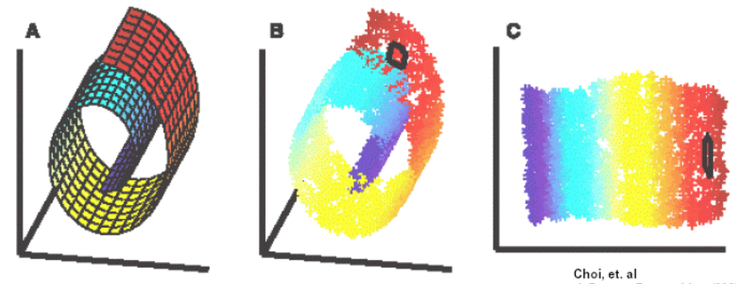




Lars H. Rohwedder, Wikimedia Commons

- A topological space that is locally Euclidean (flat, not curved).
- Dimensionality of the manifold = dimensionality of the Euclidean space it resembles, locally.
  - Straight line, wiggly curves, etc. are 1D manifolds.
  - Flat plane, surface of sphere, etc. are 2D manifolds.
- Detecting curvature of space: sum of internal angles of triangle = 180^{o}?

Manifold Learning

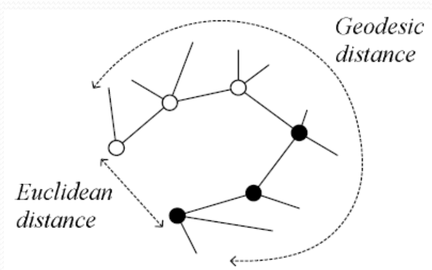


Choi, et. al
J. Pattern Recognition (2007)

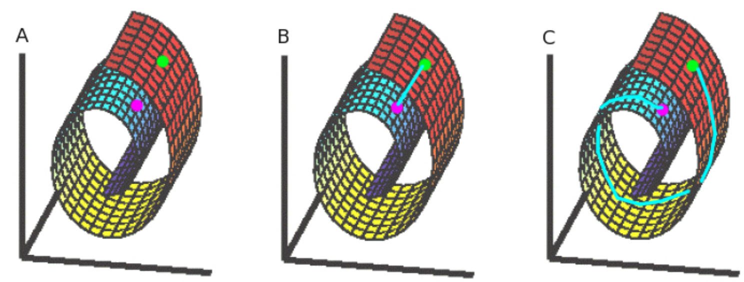
- A: 2D manifold embedded in 3D embedding space.
- B: Data points extraced from A.
- C: Recovered 2D structure.
- Task: recover C from B, without knowledge of A.

Isomap

- Geodesic distance is the distance along the manifold that the data lies in, as opposed to the Euclidean distance in the input space



Geodesic Distance



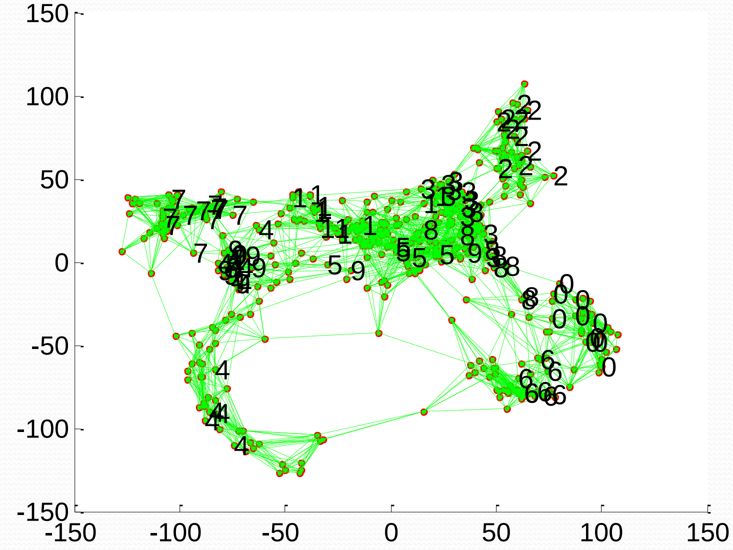
Geodesic distance = Shortest path.

- A: Manifold with two points.
- B: Euclidean distance between the two points.
- C: Geodesic distance between the two points.

Isomap

- Instances r and s are connected in the graph if
  $||\boldsymbol{x}^r - \boldsymbol{x}^s|| < \varepsilon$ or if $\boldsymbol{x}^s$ is one of the k neighbors of $\boldsymbol{x}^r$
  The edge length is $||\boldsymbol{x}^r - \boldsymbol{x}^s||$
- For two nodes r and s not connected, the distance is equal to the shortest path between them
- Once the *NxN* distance matrix is thus formed, use MDS to find a lower-dimensional mapping

Optdigits after Isomap (with neighborhood graph).



Matlab source from http://web.mit.edu/cocosci/isomap/isomap.html

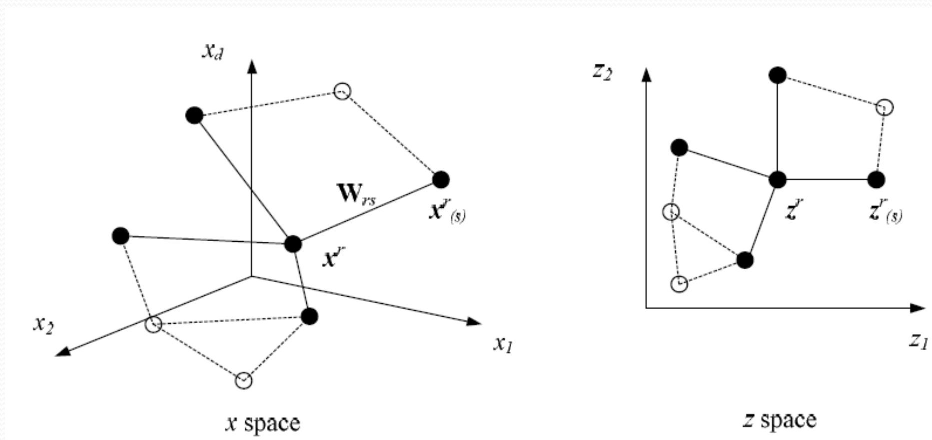
Locally Linear Embedding

1. Given $\boldsymbol{x}^r$ find its neighbors $\boldsymbol{x}^s_{(r)}$
2. Find $\mathbf{W}_{rs}$ that minimize

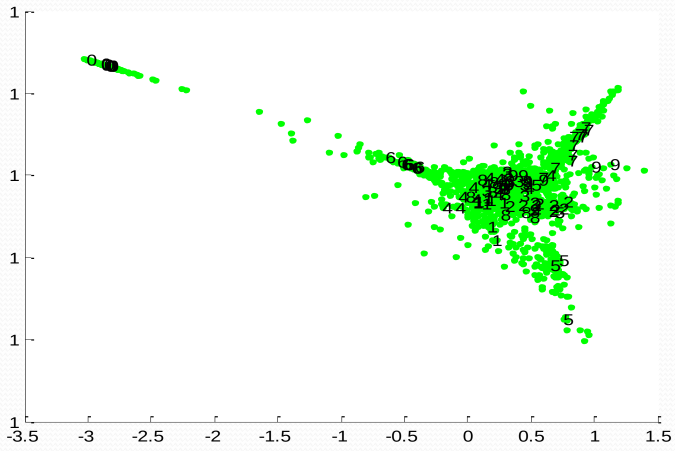
$$E(\mathbf{W} \mid X) = \sum_r \left\| \mathbf{x}^r - \sum_s \mathbf{W}_{rs} \mathbf{x}^s_{(r)} \right\|^2$$

3. Find the new coordinates $\boldsymbol{z}^r$ that minimize

$$E(\mathbf{z} \mid \mathbf{W}) = \sum_r \left\| z^r - \sum_s \mathbf{W}_{rs} z^s_{(r)} \right\|^2$$



LLE on Optdigits



Matlab source from http://www.cs.toronto.edu/~roweis/lle/code.html