Introduction to Machine Learning

Foundations and Applications

Paul J. Atzberger
University of California Santa Barbara

Unsupervised Learning: Dimension Reduction

Dimension Reduction

Motivations for Unsupervised Learning:
Given data set $S = \{x_1, x_2, \ldots, x_m\}$ what types of patterns or structure occur.

Insights into how features $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^N)$ relate to one another.

Insights into low-dimensional structure inherent in data.

Insights into groupings or clustering of data (number of classes).

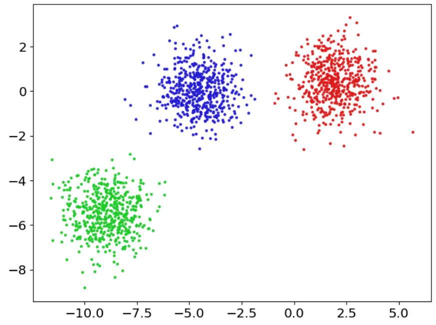
Many methods to consider depending on aims:

- **Clustering Methods (K-means)**
- **Principal Component Analysis (PCA / KPCA)**
- **Manifold Learning** + (many more methods)

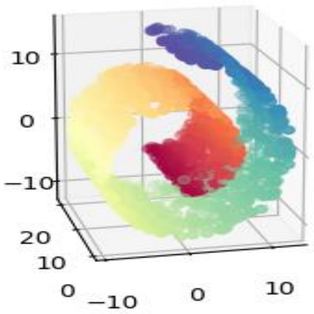
Overall methods share some common features.

Abstractly trying to learn characteristics of $\mathcal{D} \sim \mathcal{X}$.

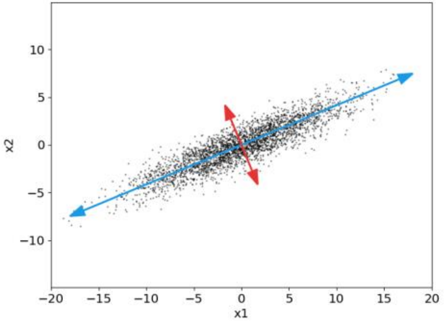
K-Means Clustering



Manifold Learning



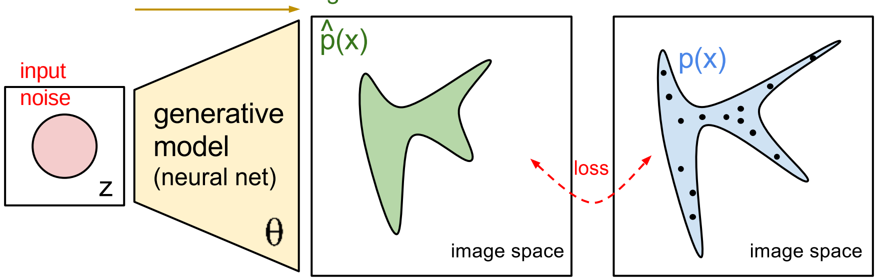
Principal Component Analysis



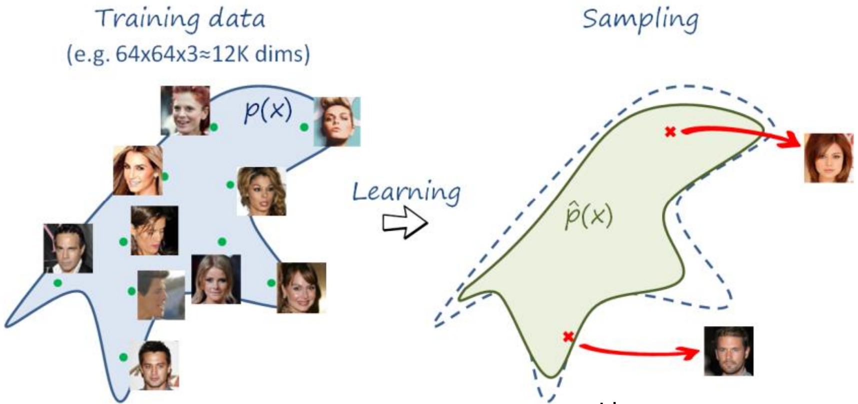
Motivations

Manifold-like structures in high dimensional spaces (natural images, audio, physical fields, PDE solutions).

Challenge: How to learn high dimensional probability distributions, generators G(z) for sampling?



open.ai



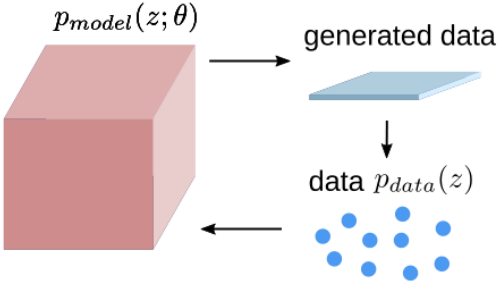
Lherranz 2018

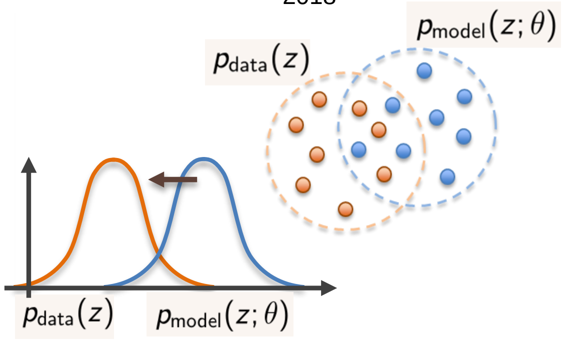
Generative Modeling

Approaches for learning models:

- Bayesian Methods
- Maximum Likelihood Estimation (MLE)
- Generative Adversarial Networks (GANs) (discussed in later lecture)
- … and many other approaches.

Challenge: How to do this in a tractable way?





Clustering: K-Means

K-Means Clustering

Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ find partition into k sets $\Omega = \{\Omega_1, \Omega_2, \ldots, \Omega_k\}$.

K-Means Clustering Optimization Problem:

$$\arg\min_{\Omega} \sum_{l=1}^{k} \sum_{x\in\Omega_l} \|\boldsymbol{x} - \mu_l\|^2$$

Challenge: Exact solution is NP-hard (requires considering all partitions).

Need to use approximate methods.

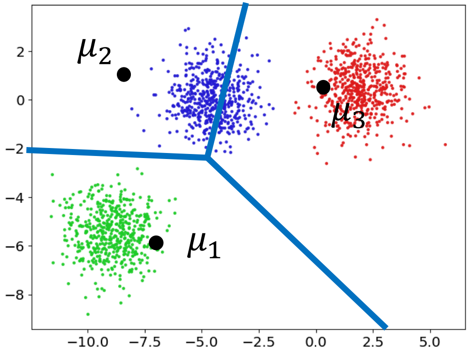
Voronoi Iteration (Lloyd's Algorithm):

- Randomly choose k initial seed points.
  - Compute Voronoi Cells of seed points and centroids.
  - Move seed points to centroid of points in each Voronoi Cell.
  - Repeat until termination criteria (seed points move less than ε).

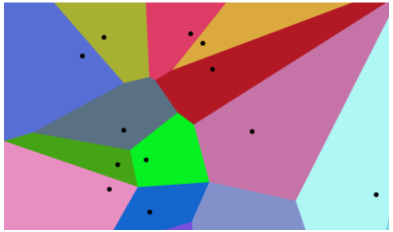
Voronoi iteration yields seeds tending toward distribution w/ more uniform cells.

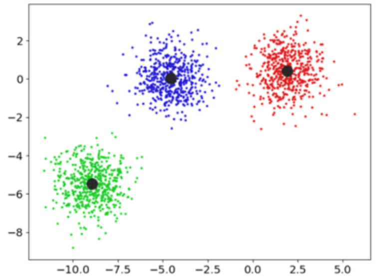
Seed points tend to migrate toward cluster centers giving approximate solution.

K-Means Clustering



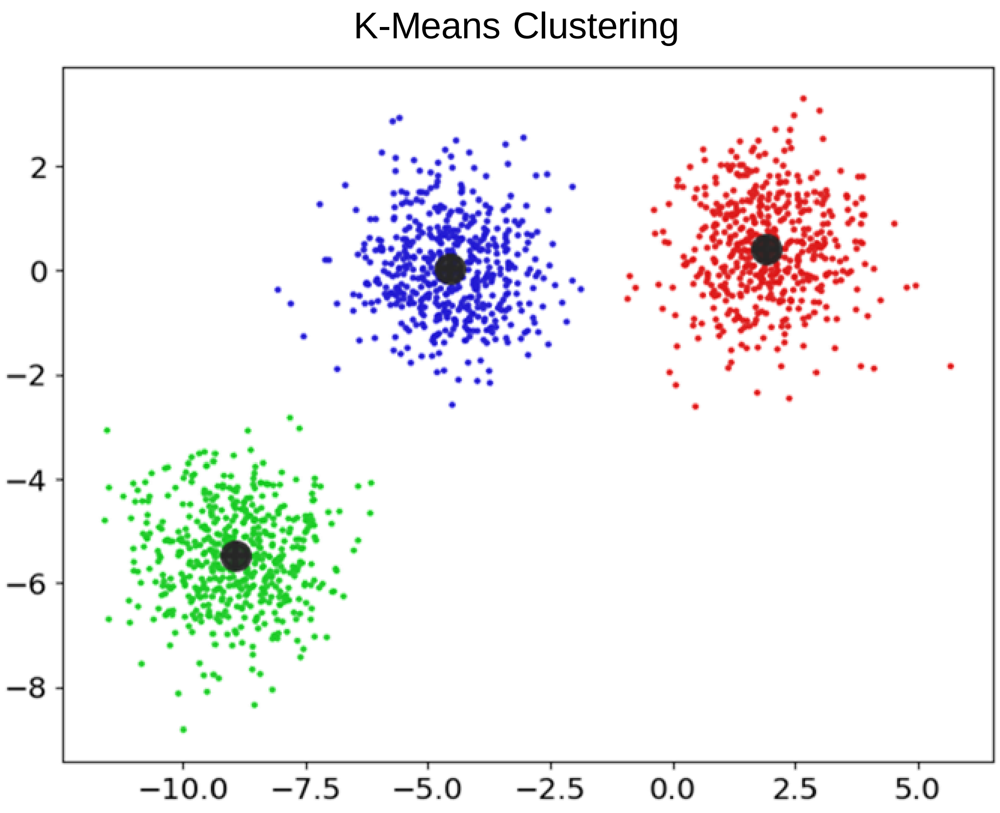
Voronoi Diagram





K-Means Clustering: Example

Example: $k = 3$ clusters, std. dev. $\sigma = 1.0$, $m = 1.5 \times 10^3$ samples.



K-Means Clustering: Example

Example: $k = 3$ clusters, std. dev. $\sigma = 1.0$, $m = 1.5 \times 10^3$ samples.

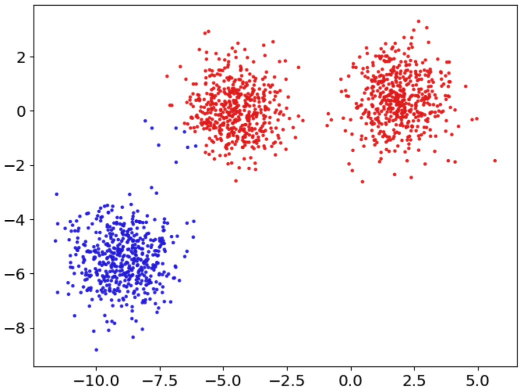
K-Means can be sensitive to hyperparameters and data distribution.

- Positing the wrong number k of clusters.
- Anisotropic cluster distribution.
- Unequal variance or size of clusters.

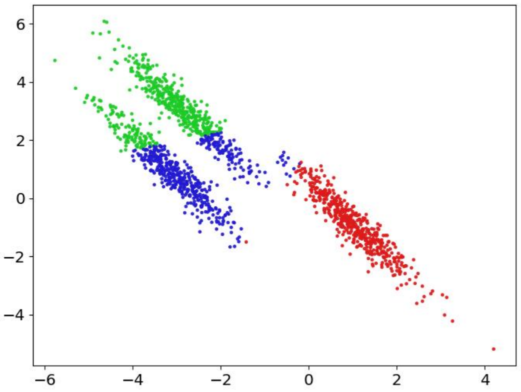
Cross validation can be used to help determine good hyperparameters.

Strong assumption that data is distributed in distinct clusters.

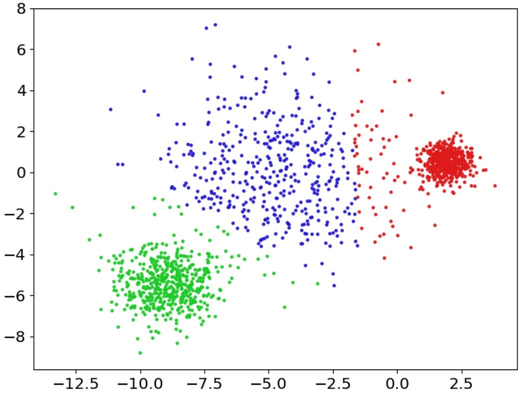
Wrong Number Clusters (k=2)



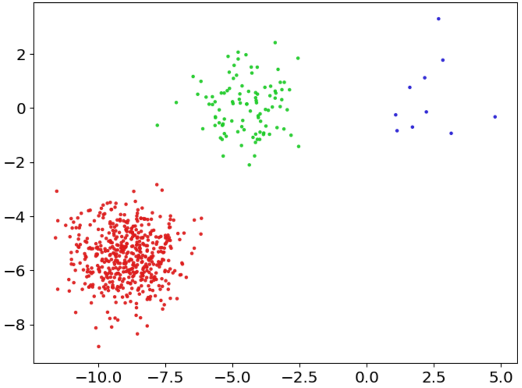
Anisotropic Clusters



Unequal Variance Clusters



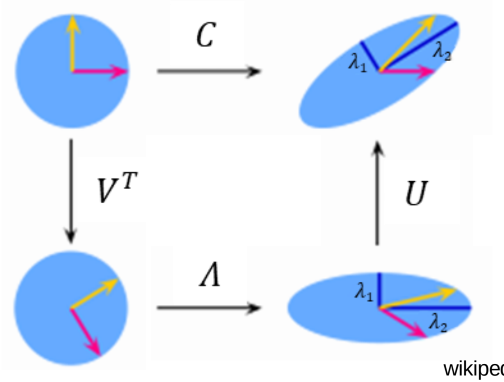
Unequal Cluster Size



Singular Value Decomposition (SVD)
and
Principal Component Analysis (PCA)

Singular Value Decomposition (SVD)

Singular Value Decomposition



wikipedia

Task: Given matrix C find the best approximating rank r matrix A_r.

Optimization Problem:

$$\min_A \|C - A\|_2$$

subject $rank(A) = r$.

Solution:

Singular Value Decomposition (SVD): $C = U\Lambda V^T \rightarrow A_r = U\Lambda_r V^T$

Features:

U, V **are orthonormal matrices.**

Λ **is diagonal matrix of singular values** $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$.

Rank r matrix $A_r = U\Lambda_r V^T$ where $\Lambda_r = diag(\lambda_1, \lambda_2, \ldots, \lambda_r, 0, \ldots 0)$ matrix of largest r singular values.

Best approximation in sense $\|C - Ar\|_2 \leq \|C - A\|_2$ for any A with $rank(A) = r$.

Many useful applications.

Example: Image Compression using SVD

Task: Given matrix C find the best approximating rank r matrix A?

Optimization Problem:

$$\min_{A} \|C - A\|_2$$
$$\text{subject } rank(A) = r.$$

Image Compression:

View image as matrix C of column vectors $\mathbf{x}_i$.

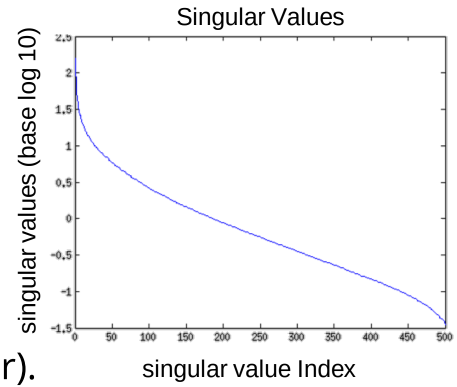
Singular Value Decomposition: $C = U\Lambda V^T$.

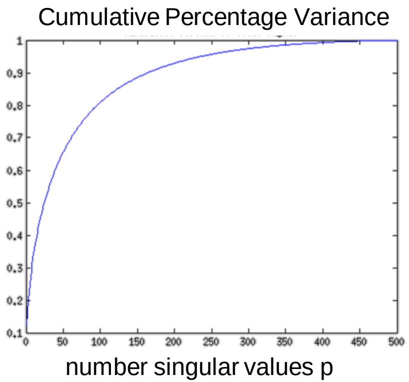
Compression → find p-vectors that span columns of **C**.

Keep only first p singular vectors of U.

Many other compression techniques for images (better).

image 800x500 pixels





Gibiansky 2015

Example: Image Compression using SVD

Task: Given matrix C find the best approximating rank r matrix A?

Optimization Problem:

$$\min_{A} \|C - A\|_2$$
$$\text{subject } rank(A) = r.$$

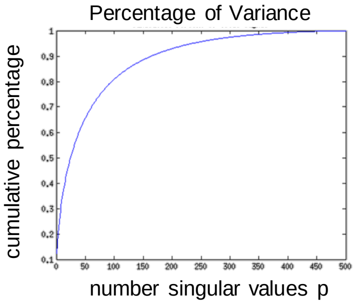


Image Compression:

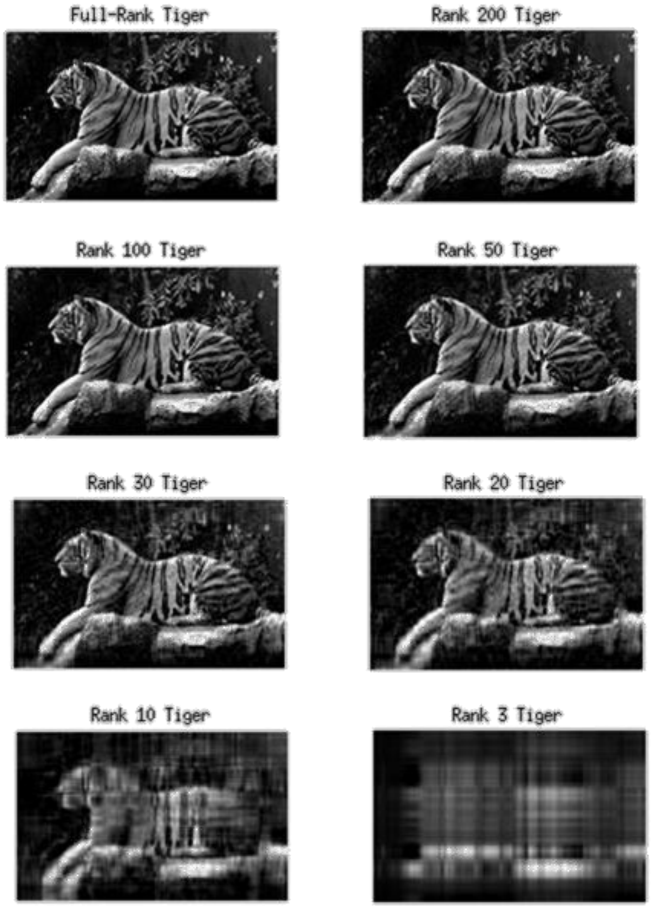
View image as matrix C of column vectors $\mathbf{x}_i$.

Singular Value Decomposition: $C = U \Lambda V^T$.

Compression → find p-vectors that span columns of **C**.

Keep only first p singular vectors of U.

Many other compression techniques for images (better).



Gibiansky 2015

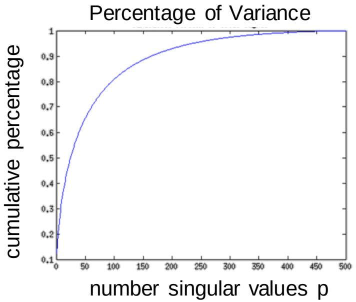
Example: Image Compression using SVD

Task: Given matrix C find the best approximating rank r matrix A?

Optimization Problem:

$$\min_{A} \|C - A\|_2$$
subject $rank(A) = r.$

Image Compression:



View image as matrix C of column vectors $\mathbf{x}_i$.

Singular Value Decomposition: $C = U\Lambda V^T$.

Compression → find p-vectors that span columns of **C**.

Keep only first p singular vectors of U.

Many other compression techniques for images (better).

Rank Full

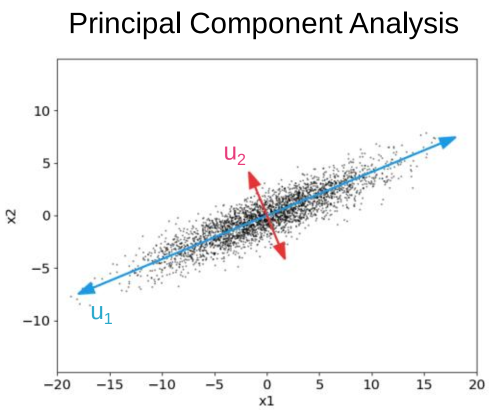
Rank 50

Rank 10

Rank 50 ~ 16% original size

Principal Component Analysis (PCA)

Principal Component Analysis (PCA)



Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ with $\boldsymbol{x}_i \in \mathbb{R}^N$ and $\sum_{i=1}^m \boldsymbol{x}_i = 0$ find the best approximating hyperplane h = span$\{w_1, w_2, \ldots w_p\}$.

Optimization Problem:

$$\min_{W,Z} \sum_{i=1}^{m} \|\boldsymbol{x}_i - W\boldsymbol{z}_i\|_2^2$$

subject $W^T W = I_{p\times p},\ W \in \mathbb{R}^{N\times p}$.

Finds optimal orthonormal basis w_i best accounting for **covariance** $C = \frac{1}{m}\sum_{i=1}^m x_i x_i^T \sim \sum_{a=1}^p \lambda_a w_a w_a^T$.

Closely related to the Singular Value Decomposition (SVD): $\mathrm{C} = U\Lambda V^T$.

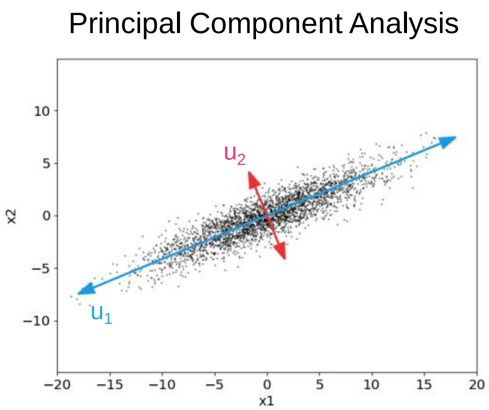
Solution: W= $[u_1,u_2,\ldots,u_p]$ first p singular vectors of U and $\mathbf{z}_i = \wp_W \mathbf{x}_i := W^T\mathbf{x}_i$, $\tilde{\mathbf{x}}_i = W\mathbf{z}_i = WW^T\mathbf{x}_i$.

Selects p directions of data $\{\mathbf{x}_i\}$ with **greatest variation.**

Projects data $\mathbf{x}_i$ to the **nearest hyperplane point** to yield $\mathbf{z}_i$.

Low-dimensional linear structure then expect **good fit** for p << N.

Principal Component Analysis (PCA)



Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m\}$ with $\boldsymbol{x}_i \in \mathbb{R}^N$ and $\sum_{i=1}^m \boldsymbol{x}_i = 0$ find the best approximating hyperplane h = span{w_1,w_2,…w_p}.

Optimization Problem:

$$\min_{W,Z} \sum_{i=1}^{m} \|\boldsymbol{x}_i - W\boldsymbol{z}_i\|_2^2$$

subject $W^T W = I_{p\times p}$, $W \in \mathbb{R}^{N\times p}$.

Finds optimal orthonormal basis w_i best accounting for **covariance** $C = \frac{1}{m}\sum_{i=1}^m x_i x_i^T \sim \sum_{a=1}^p \lambda_a w_a w_a^T$.

Closely related to the Singular Value Decomposition (SVD): $\mathrm{C} = U\Lambda V^T$

Solution: W= [u_1,u_2,…,u_p] first p singular vectors of U and $\mathbf{z}_\mathrm{i} = \wp_\mathrm{W}\mathbf{x}_\mathrm{i} \coloneqq \mathrm{W}\mathrm{W}^\mathrm{T}\mathbf{x}_\mathrm{i}$.

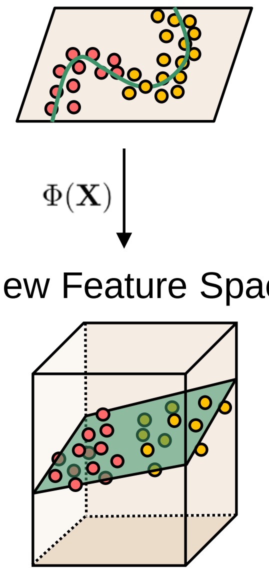
Example: m = 3x10^3 data points, exact: $\mathrm{u}_1 = [cos(\pi/8), \sin(\pi/8)]$, $\mathrm{u}_2 = [-sin(\pi/8), cos(\pi/8)]$, $\Lambda = diag(36.0,1.0)$.

Results: $\Lambda = diag(37.68, 1.01)$, w_1 = [-0.926, -0.378], w_2 = [-0.378, 0.926], $cos(\pi/8)$ = 0.924, $sin(\pi/8)$ = 0.383 → u_1=[0.924, -0.383], u_2 = [-0.383, 0.924].

Assumptions: Variation is indicative of importance of a feature, data distributed within a linear subspace.

Kernel Principal Component Analysis (KPCA):

Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ with $\boldsymbol{x}_i \in \mathbb{R}^N$, Φ feature map of K. Find in feature space $\mathcal{H}_{\text{RKHS}}$ the best approximating hyperplane h = span$\{v_1, v_2, \ldots v_p\}$ of $\tilde{S} = \{\Phi(\boldsymbol{x}_1), \Phi(\boldsymbol{x}_2), \ldots, \Phi(\boldsymbol{x}_m)\}$.



Optimization Problem:

$$\min_{W,Z} \sum_{i=1}^{m} \|\Phi(x_i) - Wzi\|_2^2$$

subject $W^T W = I_{p\times p}$, $W \in (\mathcal{H}_{\text{RKHS}})^p$.

Reformulate: $q_i = \Phi(x_i)$, $C = \frac{1}{m}\sum_{i=1}^m q_i q_i^T$. Solution using SVD, $C = U\Lambda U^T$.

$$\lambda_a u_a = C u_a = \frac{1}{m}\sum_{i=1}^m q_i q_i^T u_a = \frac{1}{m}\sum_{i=1}^m \left(q_i^T u_a\right) q_i \Rightarrow u_a = \sum_i \frac{q_i^T u_a}{m\lambda_a} q_i = \sum_i \alpha_i^a q_i, \; u_a \in \text{span}\{q_i\}.$$

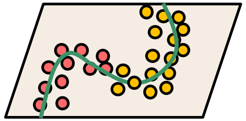
Project the equations using

$$q_i^T C u_a = \lambda_a q_i^T u_a, \quad \Rightarrow \quad q_i^T \frac{1}{m}\sum_k q_k q_k^T \sum_j \alpha_j^a q_j = \lambda_a q_i^T \sum_j \alpha_j^a q_j$$

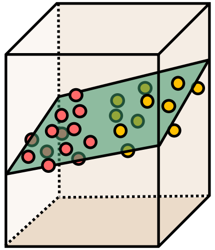
$$\Rightarrow \quad \frac{1}{m}\sum_{j,k} \alpha_j^a \left(q_i^T q_k\right)\left(q_k^T q_j\right) = \lambda_a \sum_j \alpha_j \left(q_i^T q_j\right).$$

Kernel Principal Component Analysis (KPCA):

Input Feature Space



New Feature Space



Optimization Problem:

$$\min_{W,Z} \sum_{i=1}^{m} \|\Phi(x_i) - Wzi\|_2^2$$

subject $W^T W = I_{p\times p},\ W \in (\mathcal{H}_{\text{RKHS}})^p.$

Reformulate: $q_i = \Phi(x_i),\ C = \frac{1}{m}\sum_{i=1}^{m} q_i q_i^T$. Solution using SVD, $C = U\Lambda U^T$.

$$\lambda_a u_a = C u_a = \frac{1}{m}\sum_{i=1}^{m} q_i q_i^T u_a = \frac{1}{m}\sum_{i=1}^{m}\left(q_i^T u_a\right) q_i \ \Rightarrow\ u_a = \sum_i \frac{q_i^T u_a}{m\lambda_a} q_i = \sum_i \alpha_i^a q_i,\ u_a \in \text{span}\{q_i\}.$$

Project the equations using

$$q_i^T C u_a = \lambda_a q_i^T u_a, \quad \Rightarrow \quad q_i^T \frac{1}{m}\sum_k q_k q_k^T \sum_j \alpha_j^a q_j = \lambda_a q_i^T \sum_j \alpha_j^a q_j$$

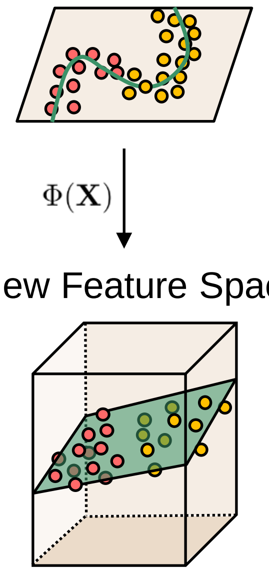
$$\Rightarrow \quad \frac{1}{m}\sum_{j,k} \alpha_j^a \left(q_i^T q_k\right)\left(q_k^T q_j\right) = \lambda_a \sum_j \alpha_j \left(q_i^T q_j\right).$$

We now use $K(x_i, x_j) = \Phi^T(x_i)\Phi(x_j) = q_i^T q_j$, $\Rightarrow\ K^2\boldsymbol{\alpha}^a = m\lambda_a K\boldsymbol{\alpha}^a$, $\Rightarrow\ K\boldsymbol{\alpha}^a = \tilde{\lambda}_a\boldsymbol{\alpha}^a$, $\tilde{\lambda}_a = m\lambda_a$.

Gives: $y_a = u_a^T q = \sum_i \boldsymbol{\alpha}_i^a q_i^T q = \sum_i \boldsymbol{\alpha}_i^a K(x_i, x)$.

Kernel Principal Component Analysis (KPCA):

Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ with $\boldsymbol{x}_i \in \mathbb{R}^N$, Φ feature map of K. Find in feature space $\mathcal{H}_{\text{RKHS}}$ the best approximating hyperplane h = span$\{v_1, v_2, \ldots v_p\}$ of $\tilde{S} = \{\Phi(\boldsymbol{x}_1), \Phi(\boldsymbol{x}_2), \ldots, \Phi(\boldsymbol{x}_m)\}$.



KPCA Reformulation with Centering:

$\mathrm{K}^c\,\boldsymbol{\alpha}^a = \tilde{\lambda}_a \boldsymbol{\alpha}^a$ where $K^c(\boldsymbol{x}_i, \boldsymbol{x}) = K(\boldsymbol{x}_i, \boldsymbol{x}) - \boldsymbol{\kappa}^T(\boldsymbol{x}_i)\mathbf{1} - \boldsymbol{\kappa}^T(\boldsymbol{x})\mathbf{1} + \mathbf{1}^T\boldsymbol{k}\mathbf{1}$,

Notation: $[K^c]_{ij} = K^c(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with $[\boldsymbol{\kappa}(\boldsymbol{x})]_l = \frac{1}{m} K(\boldsymbol{x}_l, \boldsymbol{x})$, $[\boldsymbol{k}]_{pl} = \frac{1}{m^2} K_{pl}$ and $[\mathbf{1}]_l = 1$.

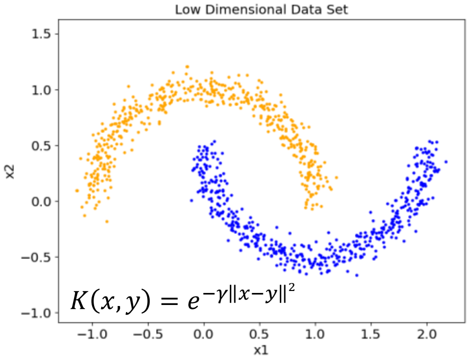
Principal components: $\boldsymbol{u}_a = \sum_{i=1}^m \alpha_i^a\, \Phi(\boldsymbol{x}_i)$ with $\boldsymbol{u}_a^T \Phi(\boldsymbol{x}) = \sum_{i=1}^m \alpha_i^a\, K^c(\boldsymbol{x}_i, \boldsymbol{x})$,
$K_{ij}^c = K_{ij} - \boldsymbol{\kappa}_i^T \mathbf{1} - \boldsymbol{\kappa}_j^T \mathbf{1} + \mathbf{1}^T \boldsymbol{k} \mathbf{1}$

Derivation: $K_{ij}^c = \left(\Phi(\boldsymbol{x}_i) - \frac{1}{m}\sum_{k=1}^m \Phi(\boldsymbol{x}_k)\right)^T \left(\Phi(\boldsymbol{x}_j) - \frac{1}{m}\sum_{l=1}^m \Phi(\boldsymbol{x}_l)\right)$

$$K_{ij}^c = \Phi(x_i)^T \Phi(x_j) - \frac{1}{m}\sum_{k=1}^m \Phi(x_k)^T \Phi(x_j) - \frac{1}{m}\sum_{\ell=1}^m \Phi(x_i)^T \Phi(x_\ell) + \frac{1}{m^2}\sum_{k,\ell} \Phi(x_k)^T \Phi(x_\ell).$$

Kernel Principal Component Analysis (KPCA): Example

Task: Given data set $S = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_m}\}$ with $\boldsymbol{x_i} \in \mathbb{R}^N$, Φ feature map of K. Find in feature space $\mathcal{H}_{\text{RKHS}}$ the best approximating hyperplane h = span$\{v_1, v_2, \ldots v_p\}$ of $\tilde{S} = \{\Phi(\boldsymbol{x_1}), \Phi(\boldsymbol{x_2}), \ldots, \Phi(\boldsymbol{x_m})\}$.



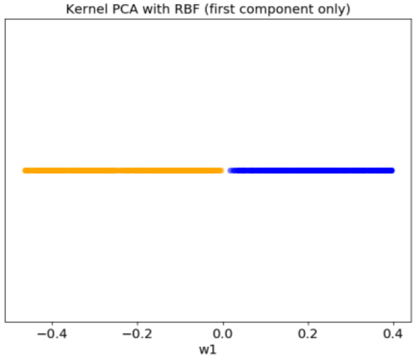
Example: Data-set S = {two semi-circles centered around (0,0) and (1,0)}, m = 10^3 samples, noise = 0.075, kernel= Radial Basis Function, $\gamma = 15$.

Nearly one dimensional structure (non-linear).

Radial basis function kernel corresponds to **similarity by proximity**.

Results:

First component w_1 := u_1 captures enough information to **separate data set.**

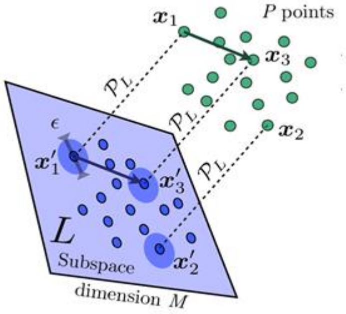


Kernel methods can be sensitive to tuning of **hyperparameters.**

Cross-validation to get good hyperparameters but need metric (two common)
- if unsupervised learning can use reconstruction errors.
- if used for supervised learning (classifier) can use validation training errors.

Random Projection

Random Projections



L, Jacques 2019

Random Projection

Consider the transform $\mathrm{x} = W\tilde{\mathrm{x}}$ with random $W \sim \mu\left(\mathbb{R}^{n\times d}\right)$ fixed for all $\tilde{\mathrm{x}} \in \mathbb{R}^d$.

We define the **distortion** r of a transformed vector x as

$$r(\mathrm{x}) := \left|\frac{\|W\mathrm{x}\|}{\|\mathrm{x}\|} - 1\right|$$

Lemma: Distortion of a Gaussian Random Projection

For a fixed $\mathrm{x} \in \mathbb{R}^d$, consider a random matrix $W \in \mathbb{R}^{n\times d}$ with each component $W_{ij} \sim \eta(0,1)$ an independent Gaussian random variable. For every $\epsilon \in (0,3)$, we have

$$\Pr\left\{\left|\frac{\|n^{-1/2}W\mathrm{x}\|^2}{\|\mathrm{x}\|^2} - 1\right| > \epsilon\right\} \leq 2\exp\left(-\epsilon^2 n/6\right).$$

Random Projections

Lemma: Distortion of a Gaussian Random Projection

For a fixed $\mathrm{x} \in \mathbb{R}^d$, consider a random matrix $W \in \mathbb{R}^{n \times d}$ with each component $W_{ij} \sim \eta(0,1)$ an independent Gaussian random variable. For every $\epsilon \in (0,3)$, we have

$$\Pr\left\{\left|\frac{\|n^{-1/2}W\mathrm{x}\|^2}{\|\mathrm{x}\|^2} - 1\right| > \epsilon\right\} \leq 2\exp\left(-\epsilon^2 n/6\right).$$

Proof: Let $\mathrm{z} = \mathrm{x}/\|\mathrm{x}\|$, then we can express the probability as

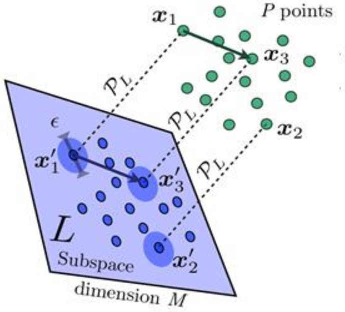
$$\Pr\left\{(1-\epsilon)\,n \leq \|W\mathrm{z}\|^2 \leq (1+\epsilon)\,n\right\} \geq 1 - 2\exp\left(-\epsilon^2 n/6\right).$$

For $\mathrm{w}_i = W_{i,*}$ the i^{th} row of W, the product $\langle \mathrm{w}_i, \mathrm{z}\rangle$ is a Gaussian random variable with mean 0 and variance $\sum_i z_j^2 = 1$. As a consequence, $\|W\mathrm{z}\|^2 = \sum_{i=1}^n \left(\langle \mathrm{w}_i, \mathrm{z}\rangle\right)^2$ is a sum of squared Gaussians.

This has the well-known χ_n^2 distribution, which satisfies $\Pr\left\{\chi_n^2 \leq (1-\epsilon)n\right\} \leq \exp\left(-\epsilon^2 n/6\right)$ and $\Pr\left\{\chi_n^2 \geq (1+\epsilon)n\right\} \leq \exp\left(-\epsilon^2 n/6\right)$.

This yields the result. ■

Random Projections



L, Jacques 2019

Lemma: Johnson-Lindenstrauss

Consider m data points $\mathrm{x}_i \in \mathbb{R}^d$ mapped to $\mathrm{y}_i = T\mathrm{x}_i \in \mathbb{R}^n$. For any $0 < \epsilon < 1/2$, $m > 4$, and $n = 20\log(m)/\epsilon^2$, consider the random projection $T : \mathbb{R}^d \to \mathbb{R}^n$ given by $T = n^{-1/2}W$ with $n < d$ and W having components $W_{ij} \sim \eta(0,1)$ i.i.d. Gaussians with mean 0 and variance 1. For any two data points x_i and x_j we have

$$(1-\epsilon)\left\|\mathrm{x}_i - \mathrm{x}_j\right\|^2 \leq \left\|T\mathrm{x}_i - T\mathrm{x}_j\right\|^2 \leq (1+\epsilon)\left\|\mathrm{x}_i - \mathrm{x}_j\right\|^2$$

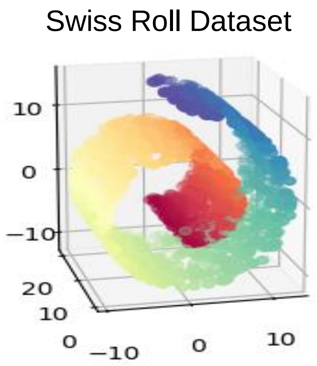
holds with probability $1-\delta$ where $\delta = 2\exp\left(-n\epsilon^2/40\right)$.

This shows **low distortion** T can be achieved provided the dimension $n \sim \log(m)/\epsilon^2$.

Random projections allow for **reducing the dimension** from d to n while **nearly preserving the pair-wise distances** between points.

Manifold Learning

Manifold Learning



Task: Given data set $S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ with $\boldsymbol{x}_i \in \mathbb{R}^N$, Find a p-dimensional manifold that approximates the data set.

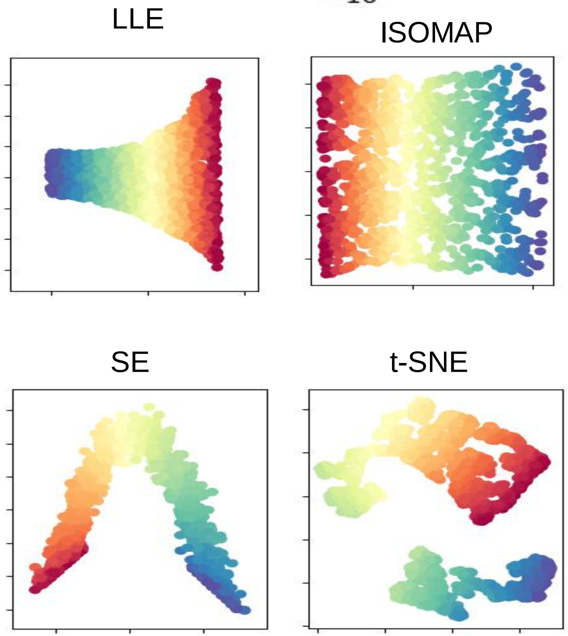
Kernel PCA is one form of manifold learning.

Many other (related) ways to try to approximate data by manifold.

- **Locally Linear Embedding** (LLE)
- **ISOMAP**
- **Spectral Embedding** (SE)
- **t-distributed Stochastic Neighbor Embedding (t-SNE)**

Can be viewed in some cases as a form of KPCA (i.e. SE, ISOMAP).

Treats data as having relevant features close to some smooth low-dimensional manifold for inference, data exploration, visualization.



Manifold Learning: t-SNE

Method:

Consider for all distinct pairs of points $\mathbf{x}_i$ and $\mathbf{x}_j$ the Gaussian probability distribution

$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2/2\sigma^2\right)}{\sum_{k\neq l}\exp\left(-\|x_k - x_l\|^2/2\sigma^2\right)}$$

Want mapping $x \in \mathbb{R}^N \to y \in \mathbb{R}^p$ into a smaller dimensional $p \ll N$.

Consider probability t-distribution for neighbors under mapping

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k\neq l}\left(1 + \|y_k - y_l\|^2\right)^{-1}}$$

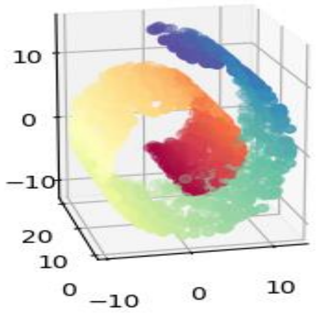
Optimization Problem:

$$\min_{y} KL(P||Q) = \sum_{i,j} p_{ij}\log\left(\frac{p_{ij}}{q_{ij}}\right)$$

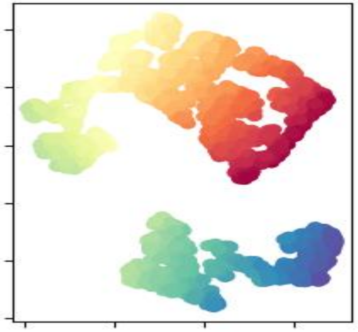
Finds points $\mathbf{y}_i$ minimizing Kullback–Leibler (KL) Divergence of pair distributions.

t-SNE used primarily for visualization (good at showing crowding / clustering).

Swiss Roll Dataset



t-SNE



Manifold Learning: Locally Linear Embedding (LLE)

Method:

Want mapping $x \in \mathbb{R}^N \to y \in \mathbb{R}^p$ into a smaller dimensional $p \ll N$.

Find for a point $\mathbf{x}_i$ the k-nearest neighbors x_j.

Optimization Problem (step I):

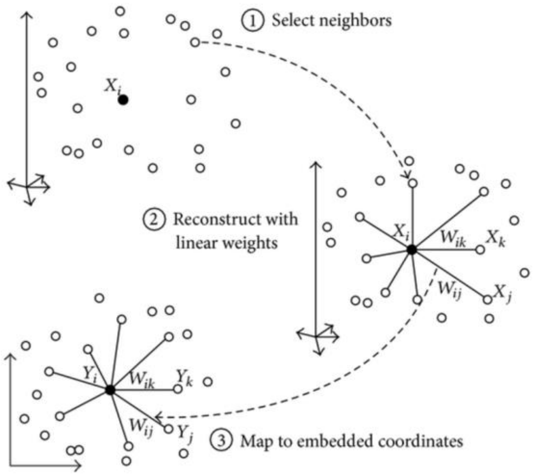
$$\min_{W} \sum_i \left\| x_i - \sum_{j \in \mathcal{N}(x_i)} W_{ij} x_j \right\|_2^2$$

$$\text{subject } W \cdot 1 = 1$$

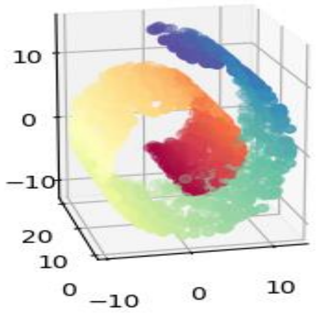
Optimization Problem (step II):

$$\min_{\boldsymbol{y}} \sum_i \left\| \boldsymbol{y}_i - \sum_{j \in \mathcal{N}(x_i)} W_{ij} \boldsymbol{y}_j \right\|_2^2$$

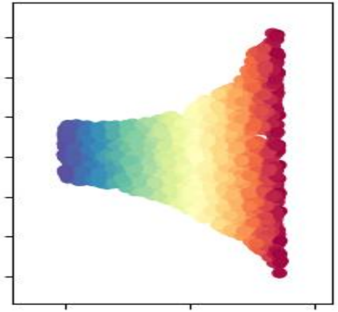
Finds points $\boldsymbol{y_i}$ so that embedding has similar linear weighting to reconstruct $\boldsymbol{y_i}$ from nearby points $\boldsymbol{y_j}$.



Swiss Roll Dataset



LLE



Manifold Learning: Isomap

Method:

Consider for $\mathbf{x}_i$ the neighborhood of points $\mathbf{x}_j$ within distance ε.

Form graph $\mathcal{G}$ with edges $\mathbf{x}_j$ in ε-neighborhood of $\mathbf{x}_i$.

Compute distance matrix Δ_{ij} using shortest path in graph between points $\boldsymbol{x}_i \to \boldsymbol{x}_j$. The Δ_{ij} ~ squared geodesic distance.

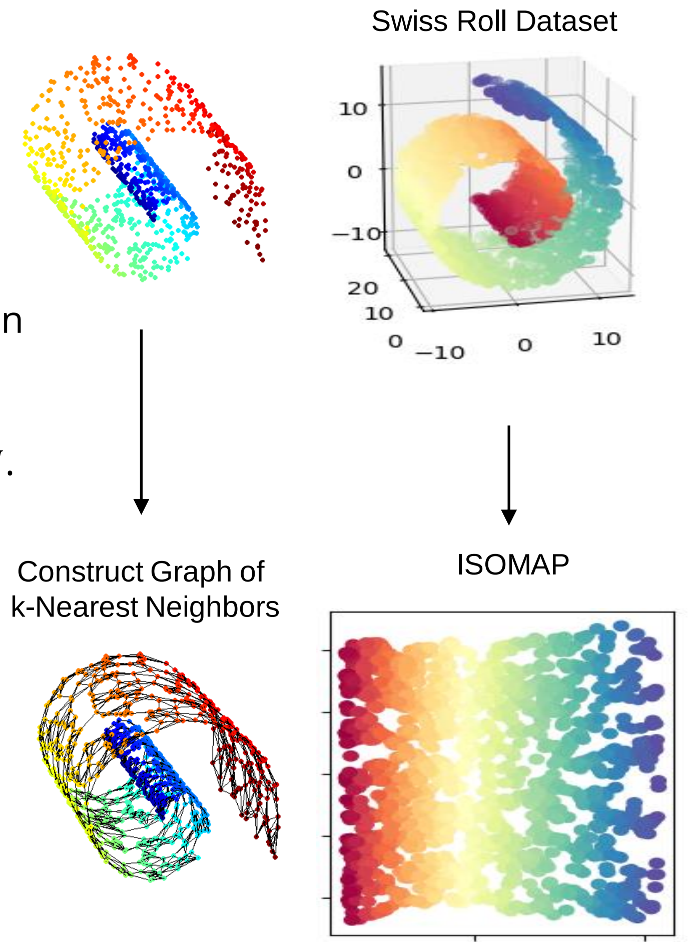
Want mapping $x \in \mathbb{R}^N \to y \in \mathbb{R}^p$ into a smaller dimensional $p \ll N$.

Optimization Problem:

$$\min_{\boldsymbol{y}} \sum_{i,j} \left(\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2 - \Delta_{ij} \right)^2$$

Finds points $\mathbf{y}_i$ so that embedding distance between pairs is close to Δ_{ij}.

Related to KCPA with kernel $\mathbf{K}_{\mathrm{Iso}} = -\frac{1}{2}\mathbf{H}\Delta\mathbf{H}$ and $\mathbf{H} = \mathbf{I}_m - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$.



Manifold Learning: Spectral Embedding

Method:

Consider for $\mathbf{x}_i$ the neighborhood of points $\mathbf{x}_j$ within distance ε.

Form graph $\mathcal{G}$ with edges $\mathbf{x}_j$ in ε-neighborhood of $\mathbf{x}_i$.

Compute weight matrix $W_{ij} = e^{-\|x_i - x_j\|_2^2/2\sigma^2}$

Want mapping $x \in \mathbb{R}^N \to y \in \mathbb{R}^p$ into a smaller dimensional $p \ll N$.

Optimization Problem:

$$\min_{\boldsymbol{y}} \sum_{i,j} W_{ij} \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2$$

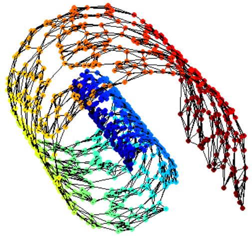
subject $y^T D y = 1$ with $D = \mathrm{diag}(W \cdot \mathbf{1})$.

Finds points $\mathbf{y}_i$ so that embedding distance between pairs minimized. Note the penalty W_{ij} is stronger for closer pairs of points and weaker for further pairs of points.

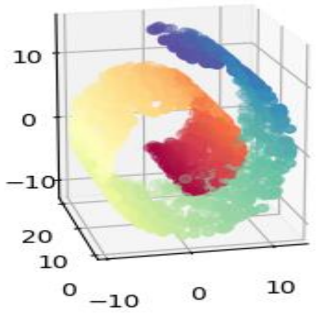
Related to KCPA with kernel $K_L = L^\dagger$ where graph Laplacian $L = D - W$ and $D = \mathrm{diag}(W \cdot \mathbf{1})$. Solution: $\boldsymbol{Y} = U_{L,p}$ from SVD of $L = U \Lambda U^T$.

Approximately preserves diffusion commute times on manifold from $\mathbf{x}_i$ to $\mathbf{x}_j$.

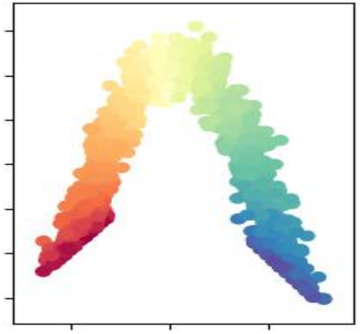
Construct Graph of ε-Neighborhoods



Swiss Roll Dataset



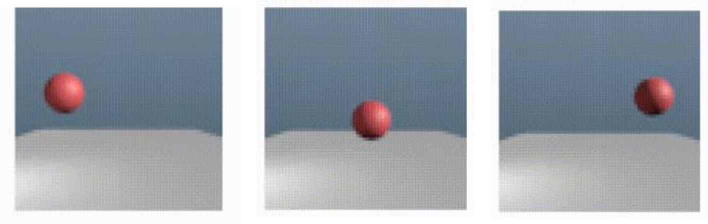
SE



Examples
Unsupervised Learning
Manifold Learning / Clustering

Pendulum Dynamics from Video: Spectral Embedding

Input: Image frames from video.

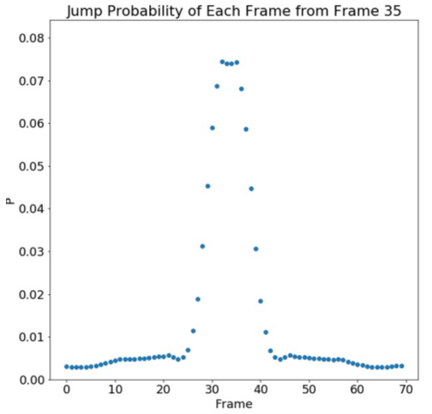


Spectral Embedding:

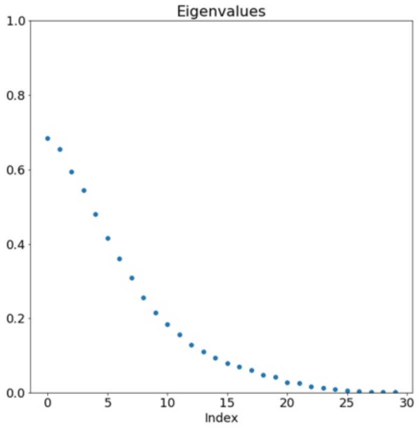
$$\min_{\boldsymbol{y}} \sum_{i,j} W_{ij} \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2$$

subject $y^T D y = 1$ with $D = \mathrm{diag}(W \cdot \mathbf{1})$.

Kernel for frames (similarity)

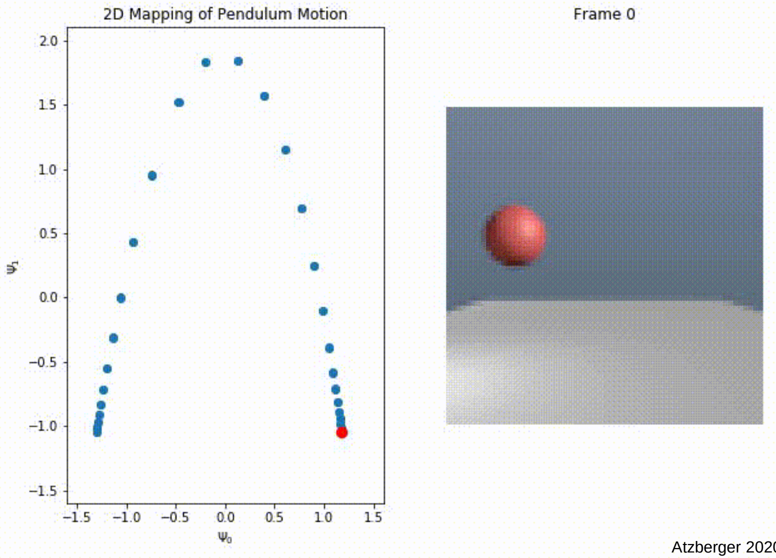


Eigenvalues Diffusion Process



Atzberger 2020

Results



Atzberger 2020

Manifold Learning and Clustering

MNIST Digits Database:

Widely used benchmark for classification methods.

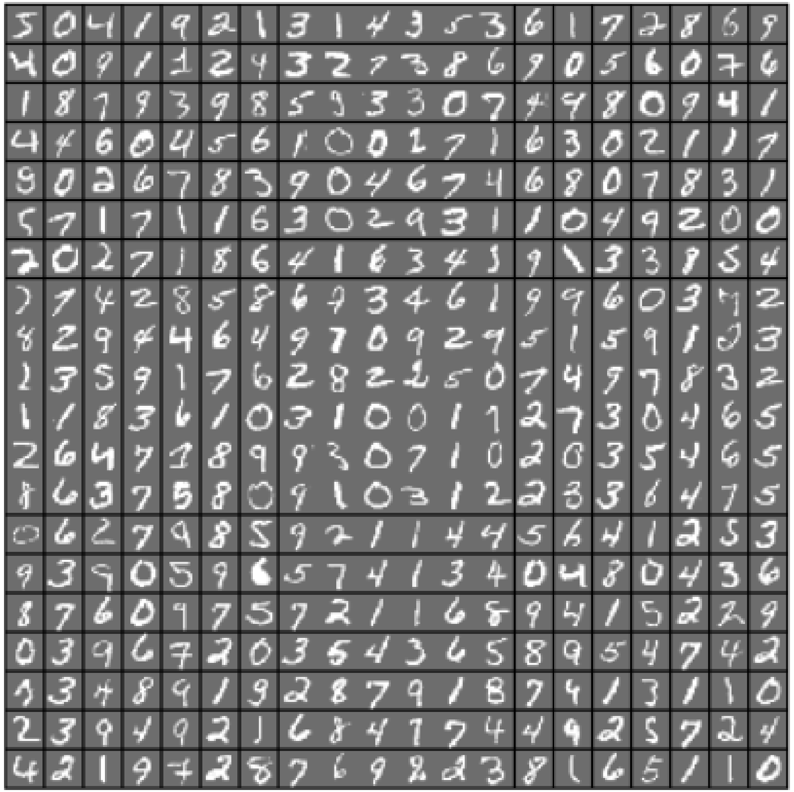
Task: Classify image to digit 0 - 9.

Data: train + test ~ 80K+ images 28x28.

Considered as points in 784-dimensional space.

Dimension reduction methods used to visualize, better understand structure, perform tasks.

t-SNE vs Local Linear Embedding (LLE).



Manifold Learning and Clustering: LLE vs t-SNE

MNIST Digits Database:

Widely used benchmark for classification methods.

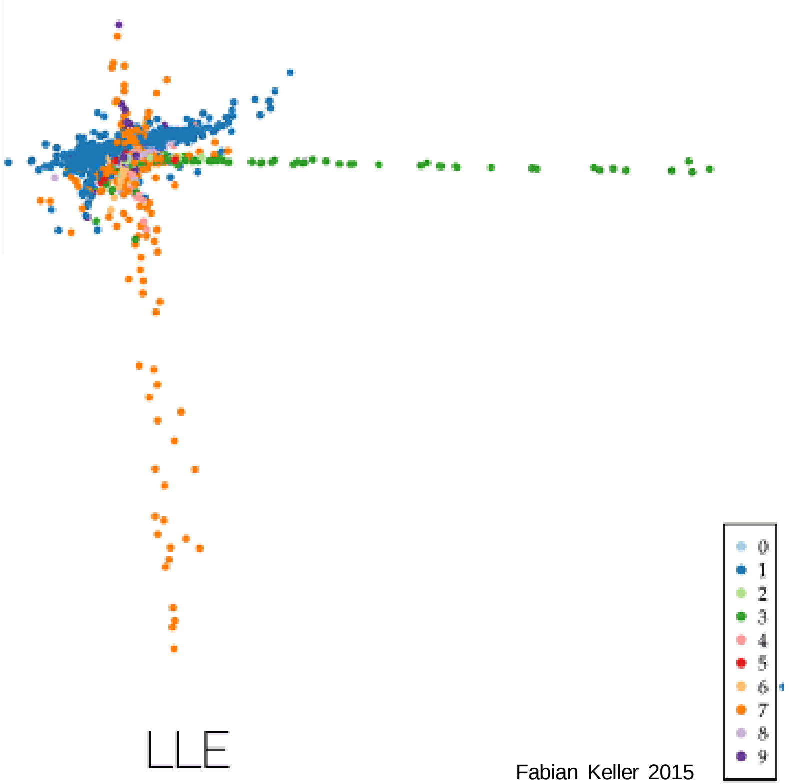
Task: Classify image to digit 0 - 9.

Data: train + test ~ 80K+ images 28x28.

Considered as points in 784-dimensional space.

Dimension reduction methods used to visualize, better understand structure, perform tasks.

t-SNE vs Local Linear Embedding (LLE).



Manifold Learning and Clustering: LLE vs t-SNE

MNIST Digits Database:

Widely used benchmark for classification methods.

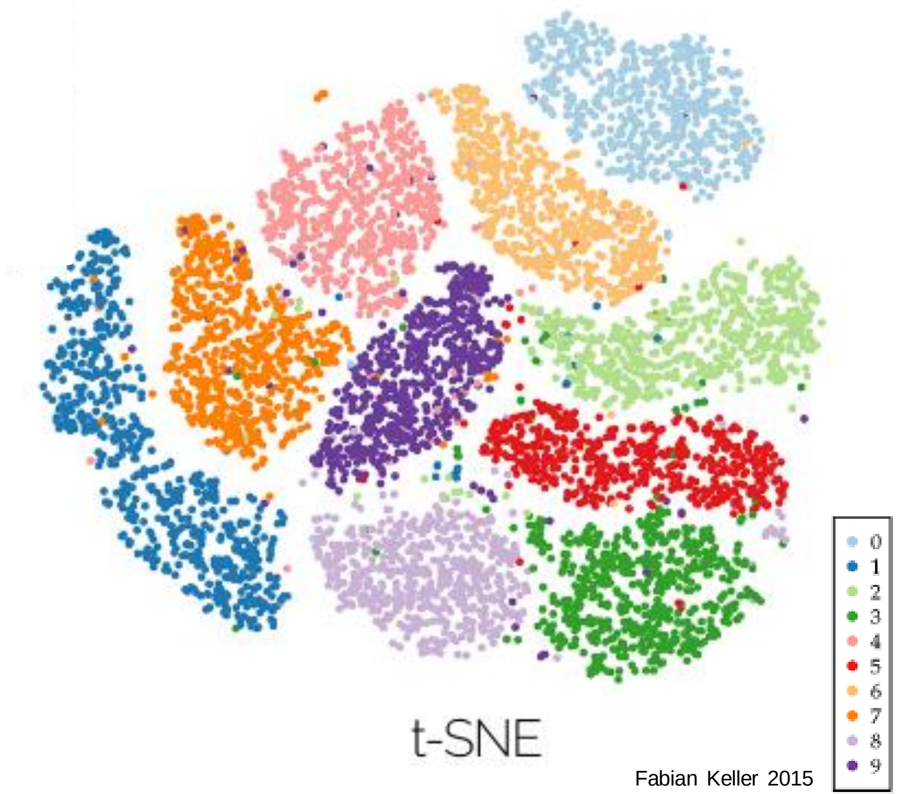
Task: Classify image to digit 0 - 9.

Data: train + test ~ 80K+ images 28x28.

Considered as points in 784-dimensional space.

Dimension reduction methods used to visualize, better understand structure, perform tasks.

t-SNE vs Local Linear Embedding (LLE).



Fabian Keller 2015

Manifold Learning: LLE

Face Database:

Local Linear Embedding (LLE).

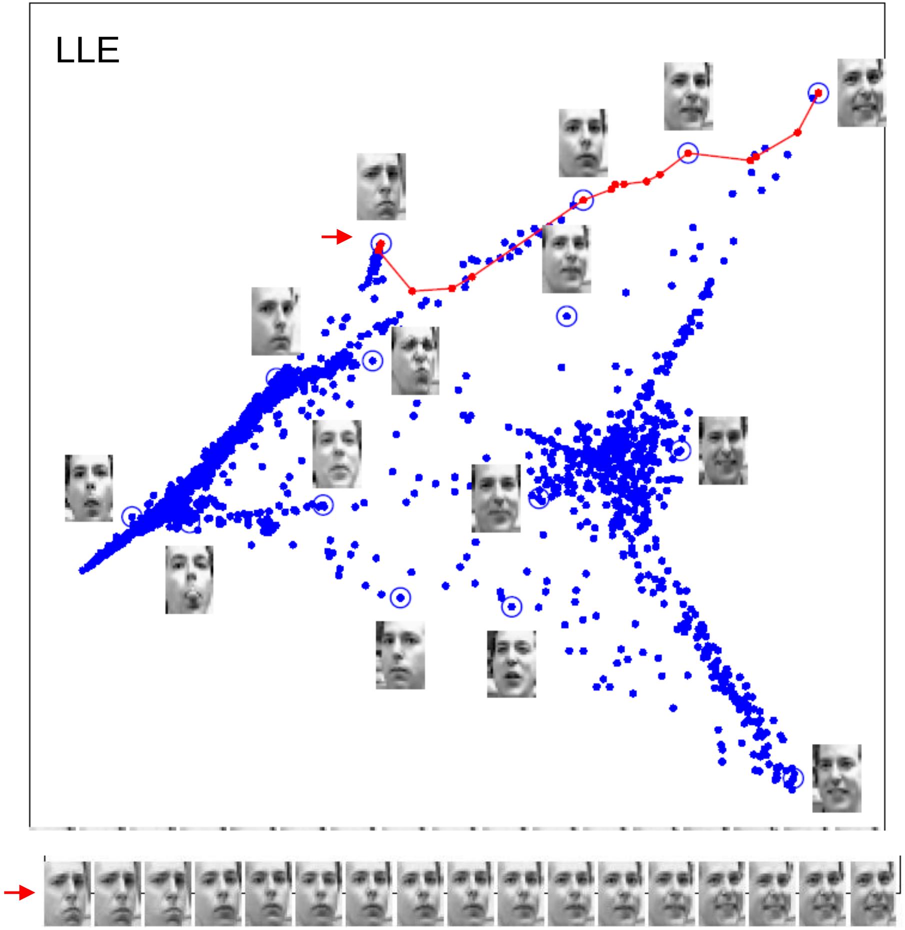
Task: Organize data set of faces by “similarity”

Data: 1965 images at 20 × 28.

Considered as points in 560-dimensional space.

Dimension reduction method useful
to visualize, better understand structure, classify, perform tasks.

Red path shows progression through embedding space.



Hastie 2017, Chen and Buja 2008.

Summary

Summary: Dimension Reduction

Motivations for Unsupervised Learning:

Abstractly trying to learn characteristics of $\mathcal{D} \sim \mathcal{X}$.

Find Structure and patterns in data $S = \{x_1, x_2, \ldots, x_m\}$.

A few features are often extremal in determining key properties.

Transforms input data allowing for incorporating **prior knowledge**.

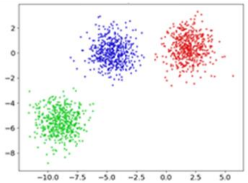
Filtering to capture essential aspects of data, improves generalization.

Methods (depends on the task):

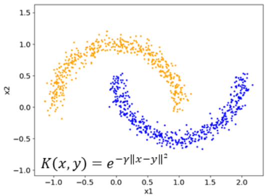
- **Clustering Methods (K-means, Spectral Graph)**
- **Principal Component Analysis (PCA / KPCA)**
- **Manifold Learning (Isomap, LLE, Spectral)**
- **Generative Adversarial Networks (GANs).**
- (many more methods)

Provides ways to learn structure from data, incorporate prior knowledge, improve generalization, improve computational efficiency.

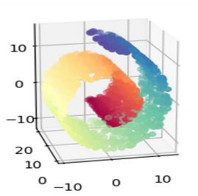
K-Means Clustering



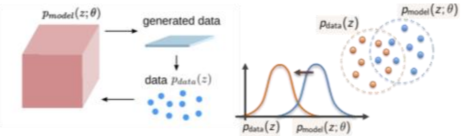
Kernel PCA



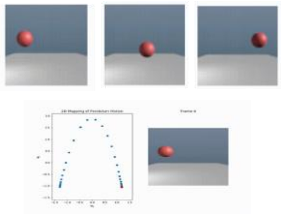
Manifold Learning



Generative Models



Spectral Embedding



GANs