

JPL-Caltech Virtual Summer School

Big Data Analytics



September 2 – 12, 2014

David R. Thompson

Jet Propulsion Laboratory, California Institute of Technology

Nonlinear Dimensionality Reduction: KPCA

Copyright 2014 California Institute of Technology. All Rights Reserved. US Government Support Acknowledged.

Objectives

1. Distinguish linear from nonlinear dimensionality reduction
2. Apply a kernel-based method (KPCA)
3. Know its advantages and pitfalls



The UN emblem is an approximate polar azimuthal equidistant projection



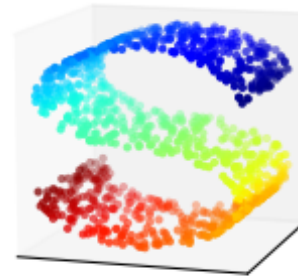
Thanks to Ali Ghodsi's tutorial for many examples and derivations!

wikipedia

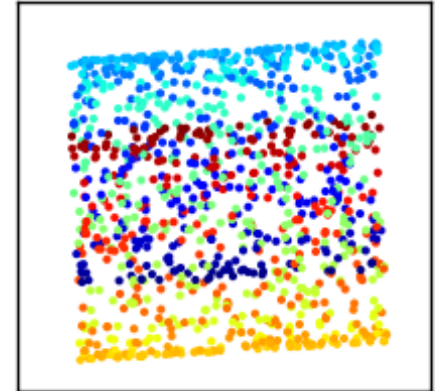
From subspaces to nonlinear manifolds

Linear subspaces may be inefficient for some datasets.

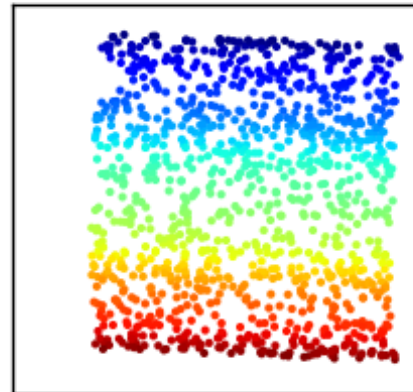
If the data is embedded on a manifold, we should capture that structure (unfolding).



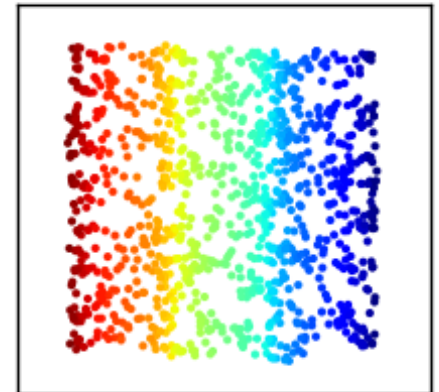
PCA projection



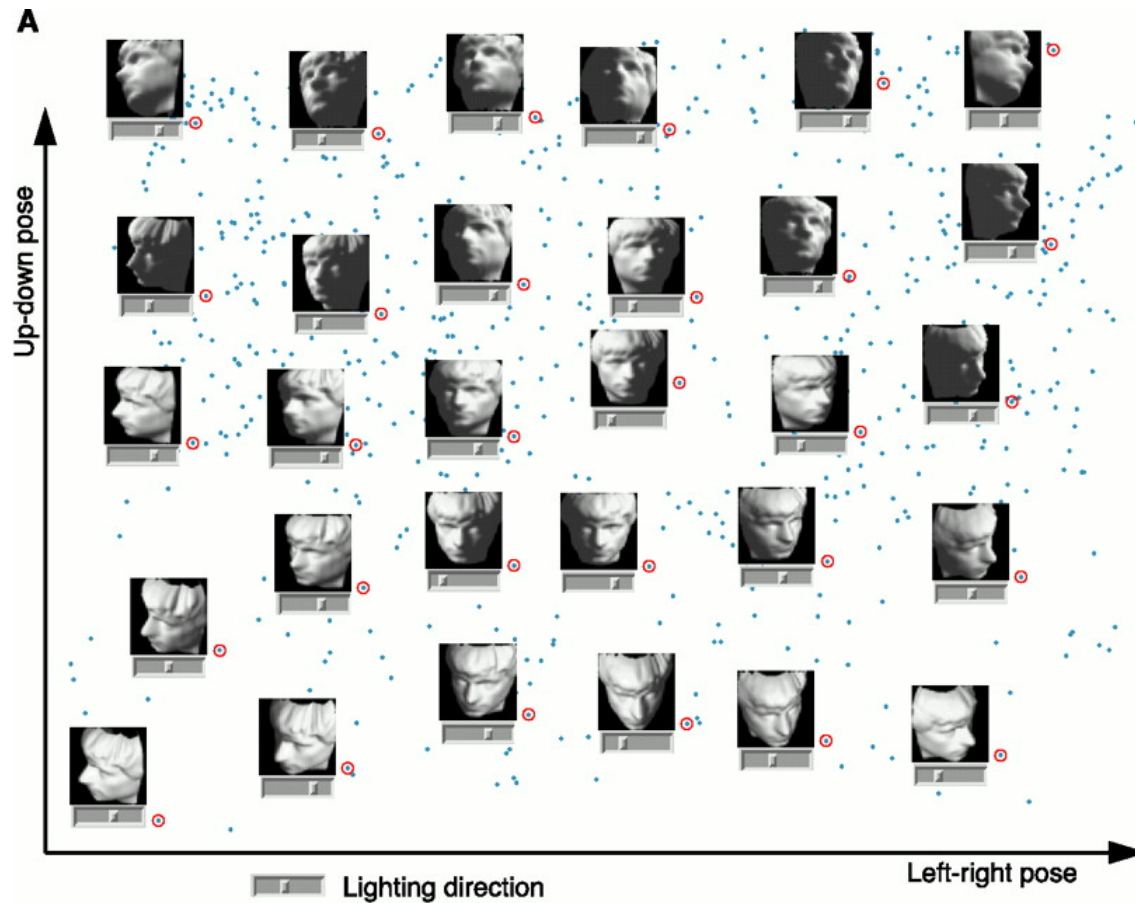
LtE projection



IsoMap projection

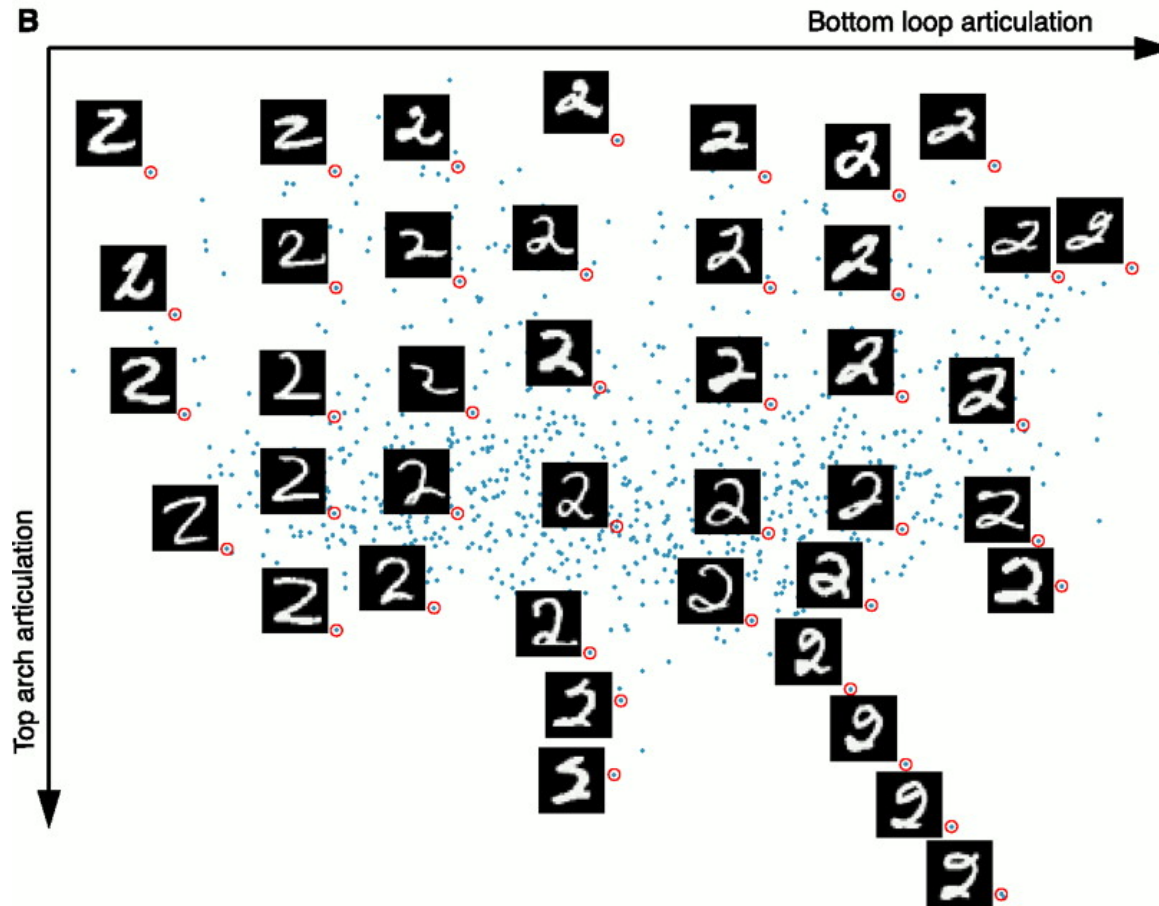


Other examples



[Tanenbaum et al., Science 22, 2009]

Other examples



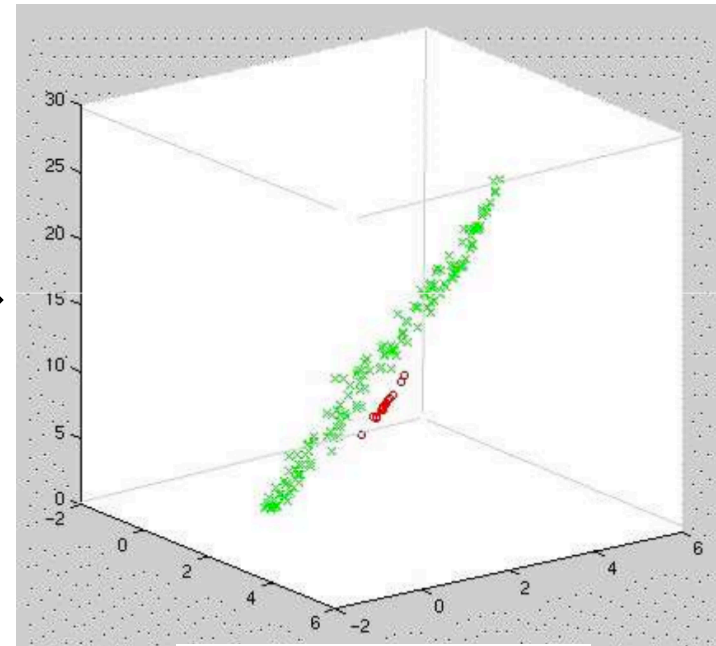
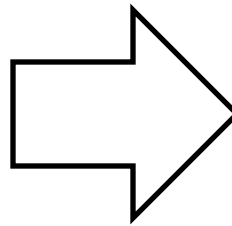
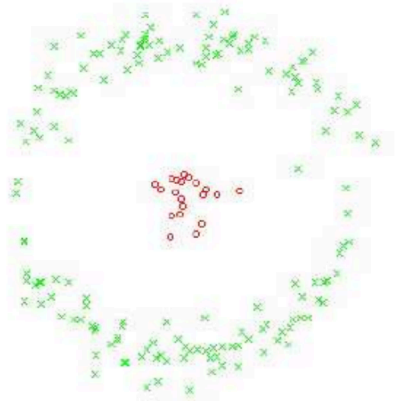
[Tanenbaum et al., Science 22, 2009]



Representing nonlinear structure

Projecting to higher dimensions can simplify data that is not linearly separable.

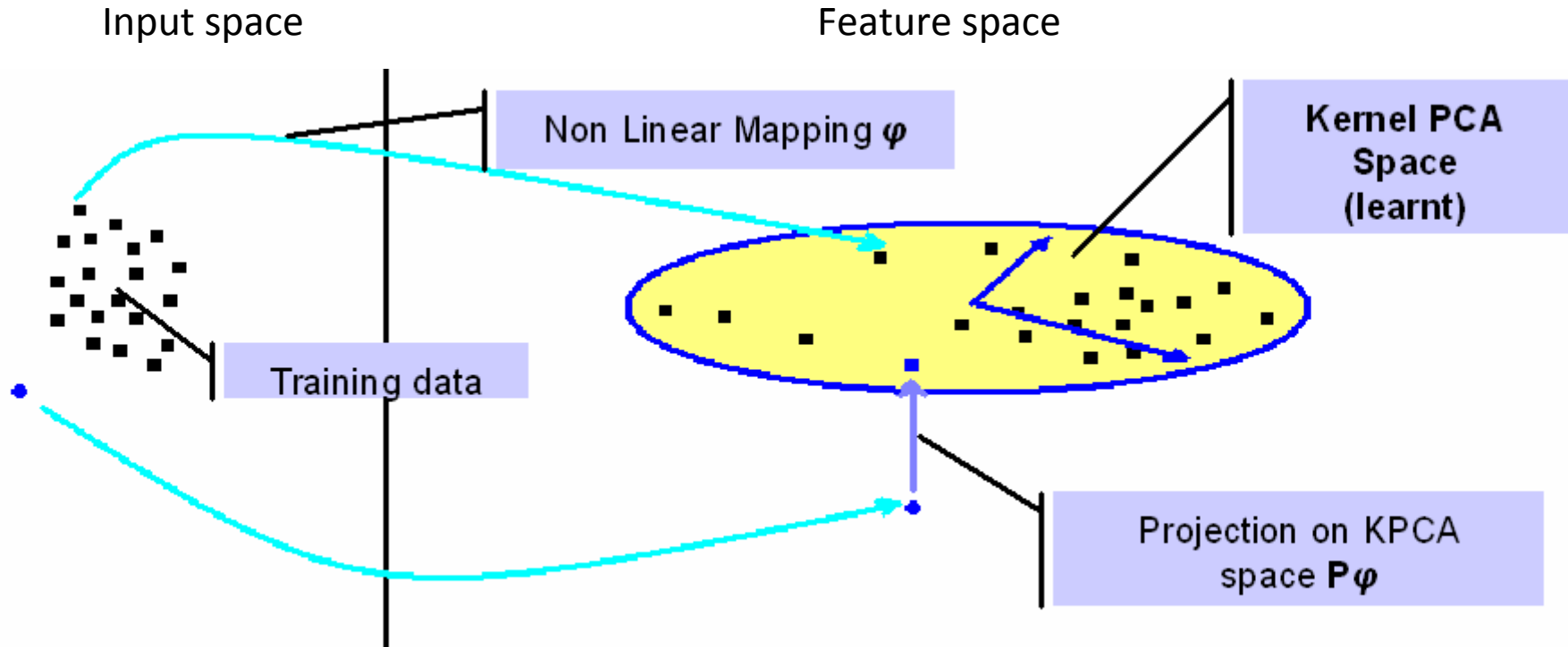
$$(x_1, x_2) \mapsto (x_1, x_2, x_1^2 + x_2^2)$$



$$\Phi : \mathbf{R}^2 \rightarrow \mathbf{R}^3$$



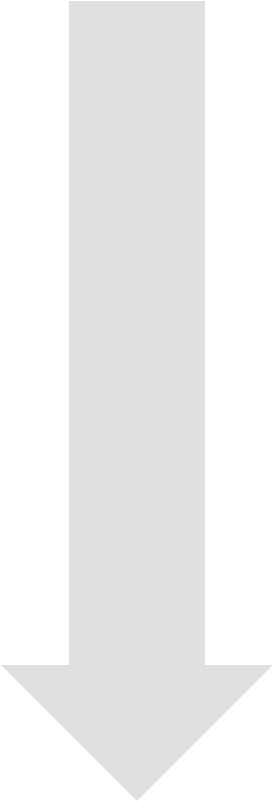
Kernel PCA



But how do we avoid having to design and manipulate these feature spaces?



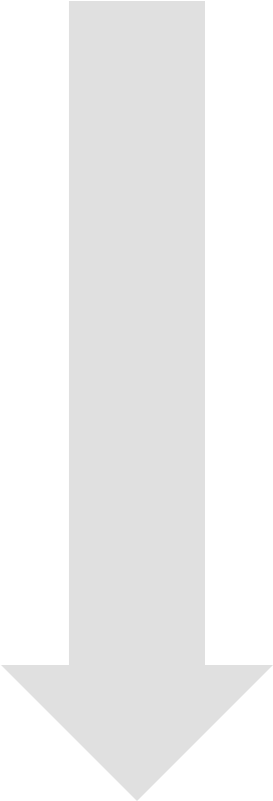
Kernel PCA objective follows PCA


$$\min \sum_i^t ||\Phi(x_i) - U_q U_q^T \Phi(x_i)||$$

Minimize
reconstruction error
in feature space, for
a *centered* dataset
 $\Phi(\mathbf{X})$



Kernel PCA objective follows PCA


$$\min \sum_i^t ||\Phi(x_i) - U_q U_q^T \Phi(x_i)||$$

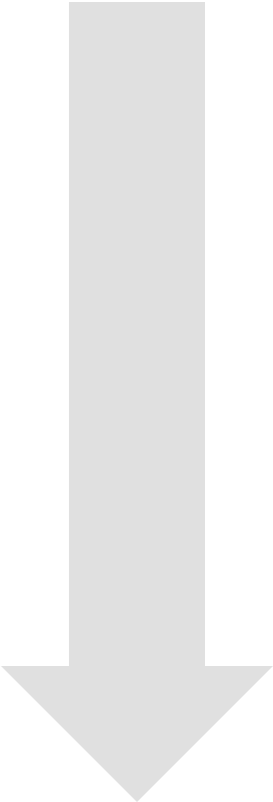
Minimize
reconstruction error
in feature space, for
a *centered dataset*
 $\Phi(\mathbf{X})$

$$\Phi(X) = U \Sigma V^T$$

Define U to be the left
singular vectors of $\Phi(\mathbf{x})$.



Kernel PCA objective follows PCA


$$\min \sum_i^t ||\Phi(x_i) - U_q U_q^T \Phi(x_i)||$$

Minimize reconstruction error in feature space, for a *centered* dataset $\Phi(\mathbf{X})$

$$\Phi(X) = U \Sigma V^T$$

Define U to be the left singular vectors of $\Phi(\mathbf{x})$.

$$\mathbf{K} = \Phi(X) \Phi(X)^T$$

Equivalently, calculate eigenvectors of the *centered* dot product matrix (as in PCA).



The “kernel trick”

Observation: PCA relies on eigenvectors of the sample covariance matrix \mathbf{XX}^T . This can be represented in the new feature space by the Kernel Matrix \mathbf{K} , composed of dot products:

$$\mathbf{K} = \Phi(X)\Phi(X)^T$$

Consequently we can *implicitly* model any nonlinear transformation $\Phi(\mathbf{x})$. The only requirement is that we can calculate the dot product $\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$ efficiently.



The kernel matrix \mathbf{K}

Contains all pairwise evaluations of the dot product

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \dots & k(\mathbf{x}_1, \mathbf{x}_d) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \dots & k(\mathbf{x}_2, \mathbf{x}_d) \\ \vdots & \vdots & & \vdots \\ k(\mathbf{x}_d, \mathbf{x}_1) & k(\mathbf{x}_d, \mathbf{x}_2) & \dots & k(\mathbf{x}_d, \mathbf{x}_d) \end{bmatrix}$$

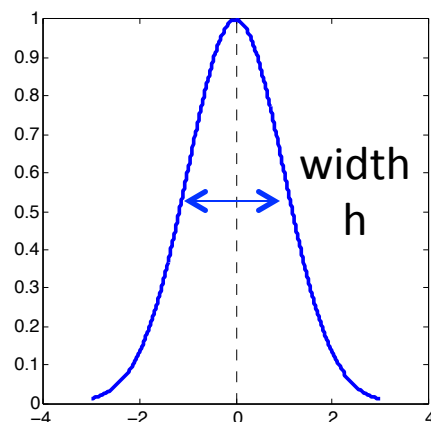
PCA's covariance matrix scales with the number of input dimensions n .

KPCA's kernel matrix scales with the number of datapoints d .



What kernel function to use?

A typical choice is the **Gaussian kernel**



$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{z} e^{-\frac{h(\mathbf{x}_i - \mathbf{x}_j)^2}{h}}$$

Reflects *local structure*: the intuition that points which are near each other in the input space should have large dot products.



Centering the dataset

Recall that our derivation required data $\Phi(\mathbf{X})$ to be zero mean. This is not the case in general.

We want:

$$\tilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{d} \sum_{k=1}^d \phi(\mathbf{x}_k)$$



Centering the dataset

Recall that our derivation required data $\Phi(\mathbf{X})$ to be zero mean. This is not the case in general.

We want:

$$\tilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{d} \sum_{k=1}^d \phi(\mathbf{x}_k)$$

Implications for the kernel matrix:

$$\begin{aligned} \tilde{K}(\mathbf{x}_i, \mathbf{x}_j) &= \tilde{\phi}(\mathbf{x}_i)^T \tilde{\phi}(\mathbf{x}_j) \\ &= \left(\phi(\mathbf{x}_i) - \frac{1}{d} \sum_{k=1}^d \phi(\mathbf{x}_k) \right)^T \left(\phi(\mathbf{x}_j) - \frac{1}{d} \sum_{k=1}^d \phi(\mathbf{x}_k) \right) \\ &= K(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{d} \sum_{k=1}^d K(\mathbf{x}_i, \mathbf{x}_k) - \frac{1}{d} \sum_{k=1}^d K(\mathbf{x}_i, \mathbf{x}_k) + \frac{1}{d^2} \sum_{l,k=1}^d K(\mathbf{x}_l, \mathbf{x}_k) \end{aligned}$$



KPCA Recipe

1. Pick a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$
2. Calculate the kernel matrix \mathbf{K}
3. Center the kernel matrix

$$\tilde{\mathbf{K}} = \mathbf{K} - 2(\mathbf{1}_{1/d})\mathbf{K} + (\mathbf{1}_{1/d})\mathbf{K}(\mathbf{1}_{1/d})$$

Matrix with entries $1/d$

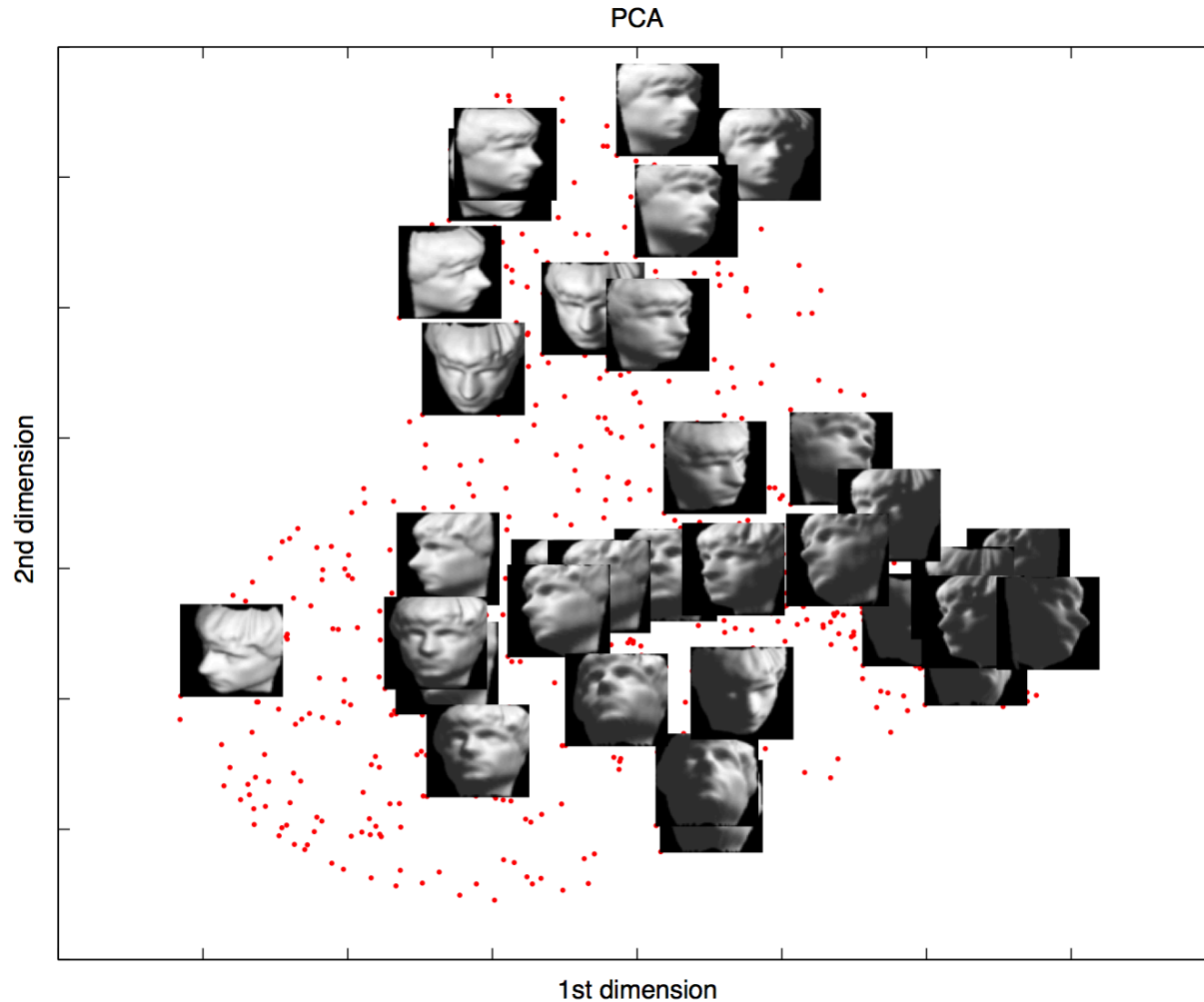
4. Solve the eigenproblem $\tilde{\mathbf{K}}\alpha_i = \lambda_i\alpha_i$
5. Project the data to each new dimension j

$$y_j = \sum_{i=1}^d \alpha_{ij} k(\mathbf{x}_i, \mathbf{x}) \quad \text{for } j = 1, 2, \dots, m$$



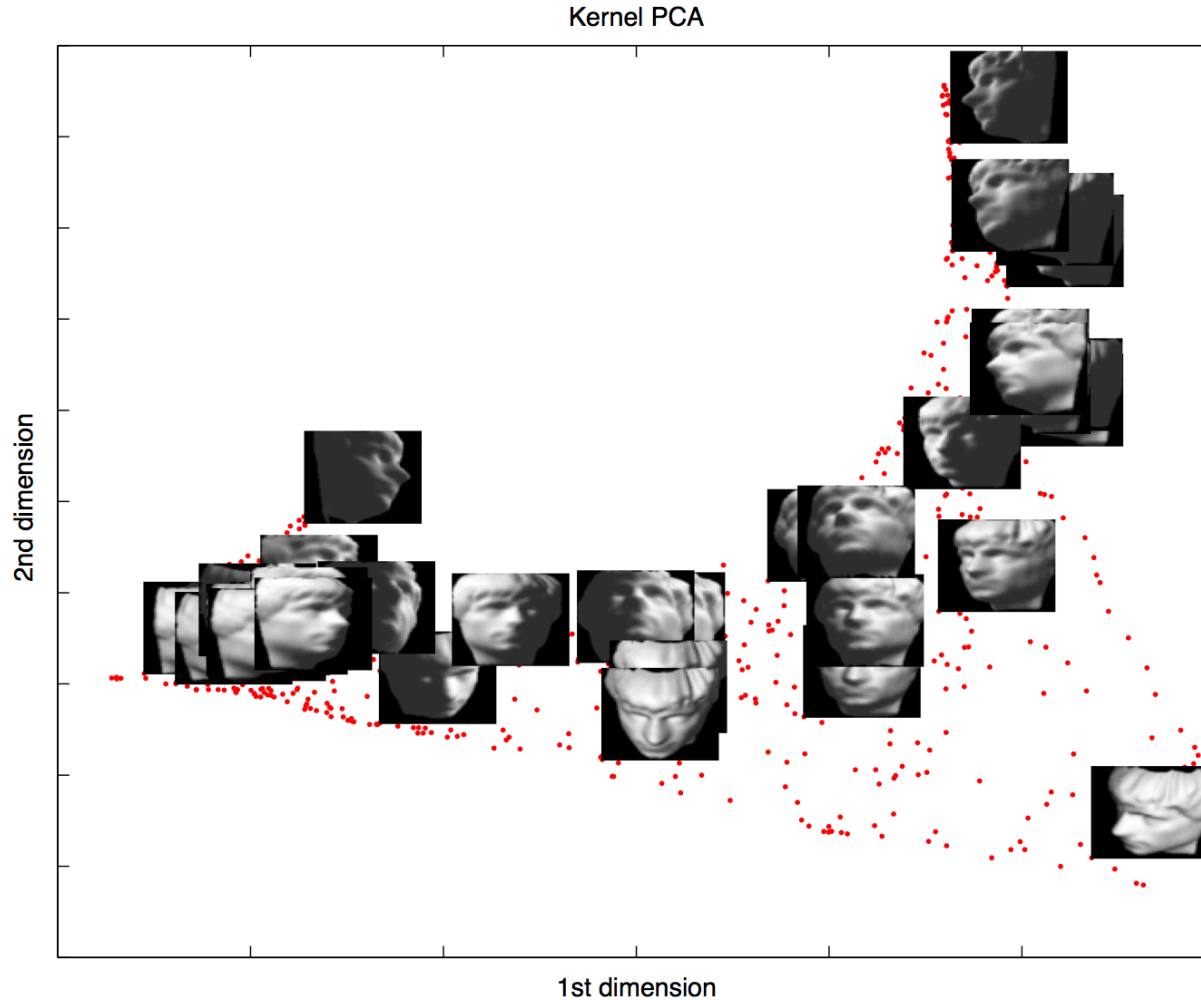
Thanks to Rita Osadchy, cs.haifa.ac.il

Example: PCA



[Ghods 2006]

Example: KPCA

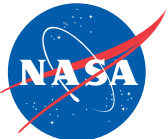


[Ghodsii 2006]



Other methods for nonlinear dimensionality reduction

- Laplacian Eigenmaps
- Local Linear Embedding
- Isomap
- Multidimensional Scaling
- Feedforward Autoencoders
- ...and more.



Nonlinear dimensionality reduction works well when...

- You have a lot of data (to fill the manifold)
- The intrinsic dimensionality is relatively low
- Your data is evenly distributed on the manifold



Summary

- Methods like KPCA can find non-linear structures by projecting data into high-dimensional spaces
- You can avoid the cost of this calculation with the kernel trick, e.g. computations based on dot products

