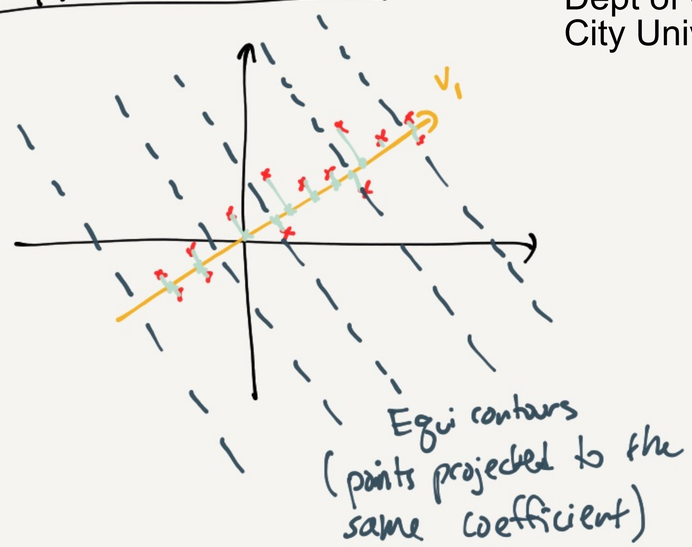
Lecture 11 - Non linear dimensionality Reduction

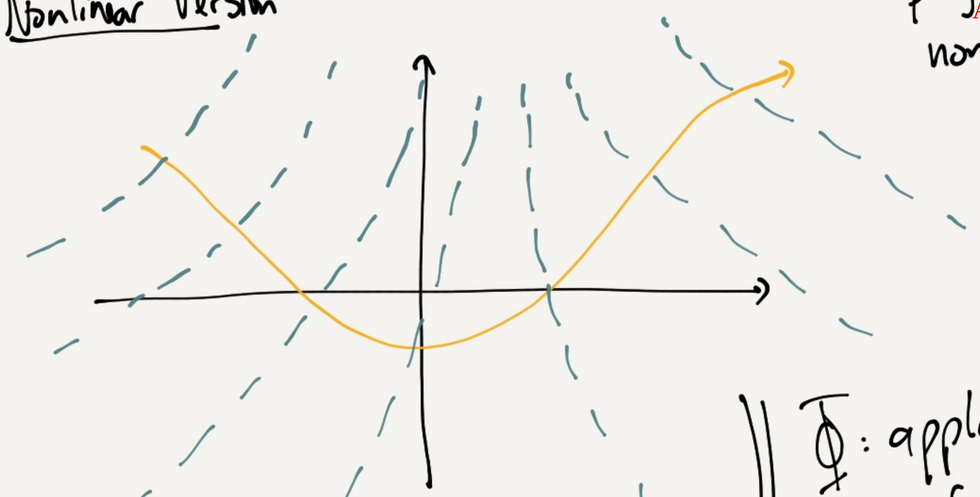
CS5487 Lecture Notes (2020)
Dr. Antoni B. Chan
Dept of Computer Science
City University of Hong Kong

PCA (linear dim reduction)

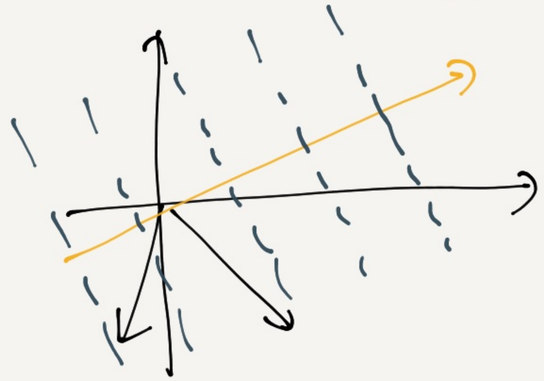


Nonlinear version

project onto nonlinear function.



Φ: apply feature xform



linear xform in high-dim space.

Kernel PCA

- Data: $\{x_1, \dots, x_n\}$
- Apply feature xform to $x_i \rightarrow \phi(x_i)$
  $X \rightarrow \Phi$
- Assume the xformed data is zero-mean (centered)
  i.e. $\sum_i \phi(x_i) = 0$
- Covariance matrix: (assuming zero-mean) $C = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)\,\phi(x_i)^T$
- Find Eigenvectors/values: (v_j, λ_j) $\quad C v_j = \lambda_j v_j$

$$\frac{1}{n} \sum_{m=1}^{n} \phi(x_m)\,\phi(x_m)^T v_j = \lambda_j v_j$$

($\phi(x_m)^T v_j$: scalar $a_{j,m}$)

$$\lambda_j v_j = \frac{1}{n} \sum_{m=1}^{n} a_{j,m}\, \phi(x_m)$$

Thus v_j has the form:

$$v_j = \sum_i a_{j,i}\, \phi(x_i) \quad \text{for some coefficients } a_{j,i}$$
$$= \Phi a_j, \quad a_j = \begin{bmatrix} a_{j,1} \\ \vdots \\ a_{j,n} \end{bmatrix}$$

Substitute for v_j

$$\Rightarrow \frac{1}{n} \sum_m \phi(x_m)\,\phi(x_m)^T \Phi a_j = \lambda_j \Phi a_j$$

($\sum_m \phi(x_m)\,\phi(x_m)^T = \Phi\Phi^T$)

$$\Rightarrow \frac{1}{n} \Phi\Phi^T \Phi a_j = \lambda_j \Phi a_j$$

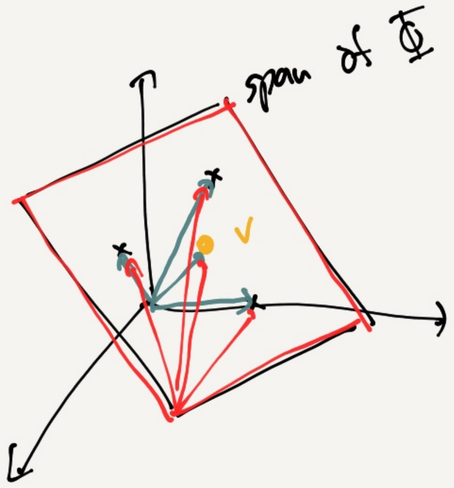
$$\frac{1}{n}\Phi\Phi^T\Phi a_j = \lambda_j \Phi a_j$$

eigenvector is a linear combo of $\phi(x_1) \dots \phi(x_n)$

Pre-multiply by Φ^T:

(equivalent set of equations by projecting onto the span of Φ)

$$\Rightarrow \frac{1}{n}\Phi^T\Phi\Phi^T\Phi a_j = \lambda_j \Phi^T\Phi a_j$$



Apply kernel trick

$$\frac{1}{n}KKa_j = \lambda_j K a_j$$

$$\Rightarrow KKa_j = n\lambda_j K a_j \quad (1)$$

Two possibilities:

1) K is invertible $\Rightarrow$ multiply by K^{-1}

$\Rightarrow$ Solve $Ka_j = n\lambda_j a_j$

2) K is not invertible $\Rightarrow$ the only difference is eigenvectors with $\lambda_j = 0 \Rightarrow$ do not affect the PCA solution.

$\Rightarrow$ Solve $Ka_j = n\lambda_j a_j$

Verification

Suppose (a_j, λ_j) are an eigenpair of K s.t.

✓ $Ka_j = n\lambda_j a_j$, $\lambda_j \neq 0$, $a_j^T a_j = 1$

Then looking at (1):

$$KKa_j = n\lambda_j Ka_j \quad \checkmark$$

($KKa_j = n\lambda_j a_j$, $n\lambda_j Ka_j = n\lambda_j n\lambda_j a_j$)

$$n\lambda_j Ka_j = n^2\lambda_j^2 a_j$$

($n\lambda_j Ka_j = n\lambda_j n\lambda_j a_j$)

$$n^2\lambda_j^2 a_j = n^2\lambda_j^2 a_j \quad \checkmark$$

PC v_j should be normalized

$$v_j^T v_j = 1$$

$$a_j^T \Phi^T \Phi a_j = 1$$

$$a_j^T K a_j = 1$$

($K a_j = n\lambda_j a_j$)

$$n\lambda_j a_j^T a_j = 1$$

$$\Rightarrow a_j \leftarrow \frac{1}{\sqrt{n\lambda_j}} a_j$$

Then $v_j^T v_j = 1$

Kernel Centering - How to center the data in the feature space $\phi(x)$?

Consider x_i, to center $\phi(x_i)$: (1 = vectors of ones)

$$\tilde{\phi}(x_i) = \phi(x_i) - \frac{1}{n}\sum_j \phi(x_j) = \phi(x_i) - \frac{1}{n}\Phi 1$$

Center all data:

$$\tilde{\Phi} = \Phi - \left[\frac{1}{n}\Phi 1\right] 1^T = \Phi\left(I - \frac{1}{n} 1 1^T\right)$$

Centered Kernel:

$$\tilde{K} = \tilde{\Phi}^T \tilde{\Phi} = \left(I - \frac{1}{n} 1 1\right)^T \Phi^T \Phi \left(I - \frac{1}{n} 1 1^T\right)$$

$$\boxed{\tilde{K} = \left(I - \frac{1}{n} 1 1\right)^T K \left(I - \frac{1}{n} 1 1^T\right)}$$

Summary KPCA (kernel PCA)

Given data X:

training:

1) calculate kernel matrix: $K = [k(x_i, x_j)]_{ij}$
2) center kernel: $\tilde{K} = (I - \frac{1}{n} 1 1^T) K (I - \frac{1}{n} 1 1^T)$
3) find the top D eigenvectors of $\tilde{K}$: $\tilde{K} a_j = \lambda_j a_j$
4) scale $a_j \leftarrow \frac{1}{\sqrt{\lambda_j}} a_j$

testing:

5) project x_*: $z_{*j} = \tilde{\phi}(x_*)^T \tilde{\Phi} a_j = \tilde{K}_*^T a_j$

$$\tilde{K}_* = K_* - \frac{1}{n} k_*^T 1 - \frac{1}{n} K 1 + \frac{1}{n^2} 1^T K 1$$

$$K_* = [k(x_*, x_i)]_i$$ ← kernel function btwn new point x_* & data X.

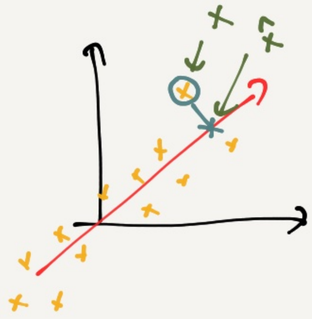
- Same properties of PCA in $\mathcal{H}$
  - the first D PC carry the most variance (in $\mathcal{H}$) minimize the reconstruction error (in $\mathcal{H}$)
  - PC are orthogonal (in $\mathcal{H}$)

Final: the original problem needs a d-dim eigenvector (d = # of features)
⇒ kernelized problem needs a N-dim eigenvector. (N = # of data)

- Kernel version of FLD ⇒ kernel discriminant analysis (10.15)

Pre-image Problem

Given PCA coefficients z, we can reconstruct $\hat{x}$.
(e.g. denoising)



given z,

$$\hat{x} = \sum_j v_j z_j = Vz$$

What about KPCA?

- given a z, what is the high-dim feature?

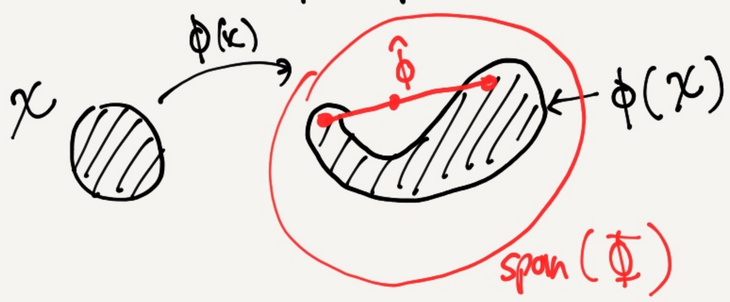
$$\hat{\phi} = \sum_j v_j z_j = \sum_j \Phi a_j z_j = \Phi\left(\sum_j a_j z_j\right)$$

$\hat{\phi}$ ← reconstruction

linear combo of $\phi(x_i)$'s

$\hat{\phi} \in \text{span}(\Phi)$

- But... not all points in $\text{span}(\Phi)$ have a corresponding x in the input space.



Example: let $\phi(x) = \begin{bmatrix} x^2 \\ x \end{bmatrix}$

$x_1 = 1 \Rightarrow \phi(x_1) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$x_2 = 2 \Rightarrow \phi(x_2) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

$\hat{\phi} = \phi(x_1) + \phi(x_2) = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$

what is the $\hat{x}$ that gives $\hat{\phi}$?

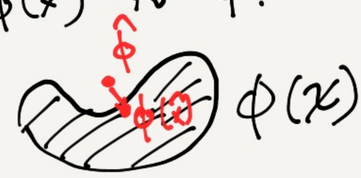
There is no $\phi(\hat{x}) = \hat{\phi}$

Approximate Pre-Image

- Consider the general problem:

$$\hat{\phi} = \Phi\alpha \quad \leftarrow \text{given vector}$$
$$= \sum_i \alpha_i \phi(x_i)$$

- Goal: find $\hat{x}$ that gives the closest $\phi(x)$ to $\hat{\phi}$.
  minimum norm reconstruction.



$$\hat{x} = \underset{x \in \mathcal{X}}{\text{argmin}} \; \| \phi(x) - \hat{\phi} \|^2 \qquad (\hat{\phi} = \Phi\alpha)$$

$$= \underset{x \in \mathcal{X}}{\text{argmin}} \; k(x,x) - 2\sum_i \alpha_i k(x, x_i) + \underbrace{\sum_i \sum_j \alpha_i \alpha_j k(x_i, x_j)}_{\text{constant wrt } x}$$

$$\boxed{\hat{x} = \underset{x \in \mathcal{X}}{\text{argmin}} \; k(x,x) - 2\sum_i \alpha_i k(x, x_i)}$$

Soln 1: (nearest neighbor): select $x \in X$ (dataset)

$$\hat{x} = \underset{j}{\text{argmin}} \; k(x_j, x_j) - 2k_j^T\alpha$$

where k_j = j^{th} column of kernel matrix K

Soln 2: solve the minimization directly w/ optimizer.

Soln 3: Suppose $k(x,x) = 1$, $k(x_i, x_j) \geq 0$ (e.g. Gaussian kernel)
and k is homogeneous, $k(\|x_i - x_j\|)$

$$\hat{x} = \underset{}{\text{argmin}} \; -\sum_i \alpha_i k(x_i, x) = \underset{}{\text{argmax}} \; \sum_i \alpha_i k(x_i, x)$$
$$= \underset{}{\text{argmax}} \; \left(\sum_i \alpha_i k(x_i, x)\right)^2$$

$\Rightarrow$ iterative algorithm:
(like mean-shift w/ weights α_i on each x_i)

$$\hat{x} \leftarrow \frac{\sum_i \alpha_i k'(\|x_i - \hat{x}\|^2)\, x_i}{\sum_i \alpha_i k'(\|x_i - \hat{x}\|^2)}$$