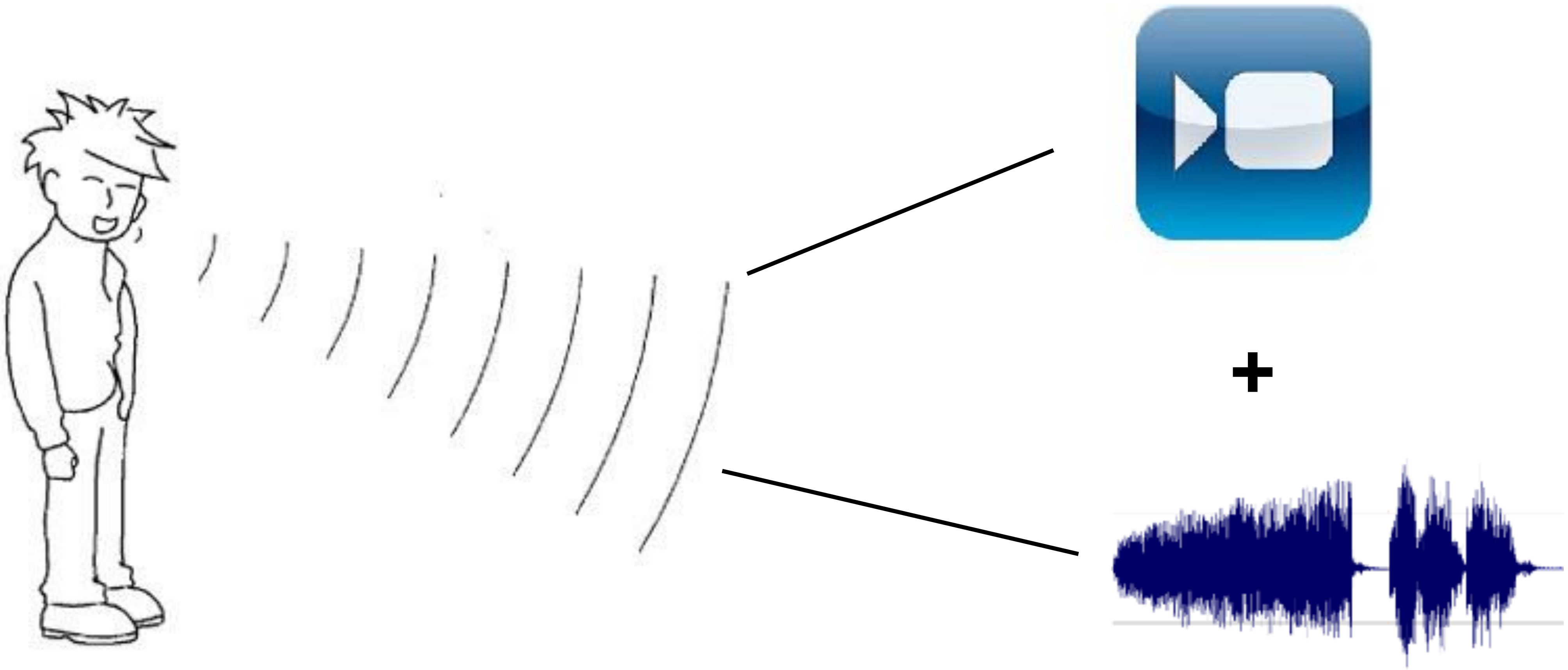
Machine Learning for Data Science (CS4786)
Lecture 6

Canonical Correlation Analysis
+
Kernel PCA

Example I: Speech Recognition



- Audio might have background sounds uncorrelated with video
- Video might have lighting changes uncorrelated with audio
- Redundant information between two views: the speech

Canonical Correlation Analysis

Canonical Correlation Analysis

Age

+ Gender

Candies per week

Favorite Cartoon

Maximizing Correlation Coefficient

- Say $\mathbf{w}_1$ and $\mathbf{v}_1$ are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

$$\frac{\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[1]\right)\cdot\left(\mathbf{y}'_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}'_t[1]\right)}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[1]\right)^2}\sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}'_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}'_t[1]\right)}}$$

Maximizing Correlation Coefficient

- Say $\mathbf{w}_1$ and $\mathbf{v}_1$ are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

$$\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[1]-\frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[1]\right)\cdot\left(\mathbf{y}'_t[1]-\frac{1}{n}\sum_{t=1}^{n}\mathbf{y}'_t[1]\right)$$

where $\mathbf{y}_t[1]=\mathbf{w}_1^\top\mathbf{x}_t$ and $\mathbf{y}'_t[1]=\mathbf{v}_1^\top\mathbf{x}'_t$

Maximizing Correlation Coefficient

- Say $\mathbf{w}_1$ and $\mathbf{v}_1$ are the directions we choose to project in views 1 and 2 respectively we want these directions to maximize,

$$\frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[1]\right)\cdot\left(\mathbf{y}'_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}'_t[1]\right)$$

$$\text{s.t. } \frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[1]\right)^2 = \frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{y}'_t[1] - \frac{1}{n}\sum_{t=1}^{n}\mathbf{y}'_t[1]\right) = 1$$

where $\mathbf{y}_t[1] = \mathbf{w}_1^\top \mathbf{x}_t$ and $\mathbf{y}'_t[1] = \mathbf{v}_1^\top \mathbf{x}'_t$

Canonical Correlation Analysis

- Hence we want to solve for projection vectors $\mathbf{w}_1$ and $\mathbf{v}_1$ that

$$\text{maximize } \frac{1}{n}\sum_{t=1}^{n} \mathbf{w}_1^\top(\mathbf{x}_t - \mu) \cdot \mathbf{v}_1^\top(\mathbf{x}_t' - \mu')$$

$$\text{subject to } \frac{1}{n}\sum_{t=1}^{n} (\mathbf{w}_1^\top(\mathbf{x}_t - \mu))^2 = \frac{1}{n}\sum_{t=1}^{n} (\mathbf{v}_1^\top(\mathbf{x}_t' - \mu'))^2 = 1$$

where $\mu = \frac{1}{n}\sum_{t=1}^{n} \mathbf{x}_t$ and $\mu' = \frac{1}{n}\sum_{t=1}^{n} \mathbf{x}_t'$

- Hence we want to solve for projection vectors $\mathbf{w}_1$ and $\mathbf{v}_1$ that

$$\begin{aligned} &\text{maximize } \mathbf{w}_1^\top \Sigma_{1,2} \mathbf{v}_1 \\ &\text{subject to } \mathbf{w}_1^\top \Sigma_{1,1} \mathbf{w}_1 = \mathbf{v}_1^\top \Sigma_{2,2} \mathbf{v}_1 = 1 \end{aligned}$$

Canonical Correlation Analysis

- Hence we want to solve for projection vectors $\mathbf{w}_1$ and $\mathbf{v}_1$ that

$$\text{maximize } \mathbf{w}_1^\top \Sigma_{1,2} \mathbf{v}_1$$
$$\text{subject to } \mathbf{w}_1^\top \Sigma_{1,1} \mathbf{w}_1 = \mathbf{v}_1^\top \Sigma_{2,2} \mathbf{v}_1 = 1$$

$$\Sigma = \begin{matrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{matrix} = \mathrm{cov}\left(X \; X' \right)$$

CCA Algorithm

CCA Algorithm

1. $$X = \left({}^{n}\underbrace{X_1}_{d_1}, \underbrace{X_2}_{d_2} \right)$$

CCA Algorithm

1. $$X = \left(\underset{d_1}{n \; X_1} , \underset{d_2}{X_2} \right)$$

2. $$\Sigma = \begin{matrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{matrix} = \mathrm{cov}\left(X \right)$$

CCA Algorithm

1. $$X = \left(n \; X_1 , \; X_2 \right)$$
   d_1 $\quad$ d_2

2. $$\Sigma = \begin{matrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{matrix} = \mathrm{cov}\left(X \right)$$

3. $$W = \mathrm{eigs}\left(\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}, K \right)$$

CCA Algorithm

1. $X = \left({}^{n} \boxed{X_1}_{d_1}, \boxed{X_2}_{d_2} \right)$

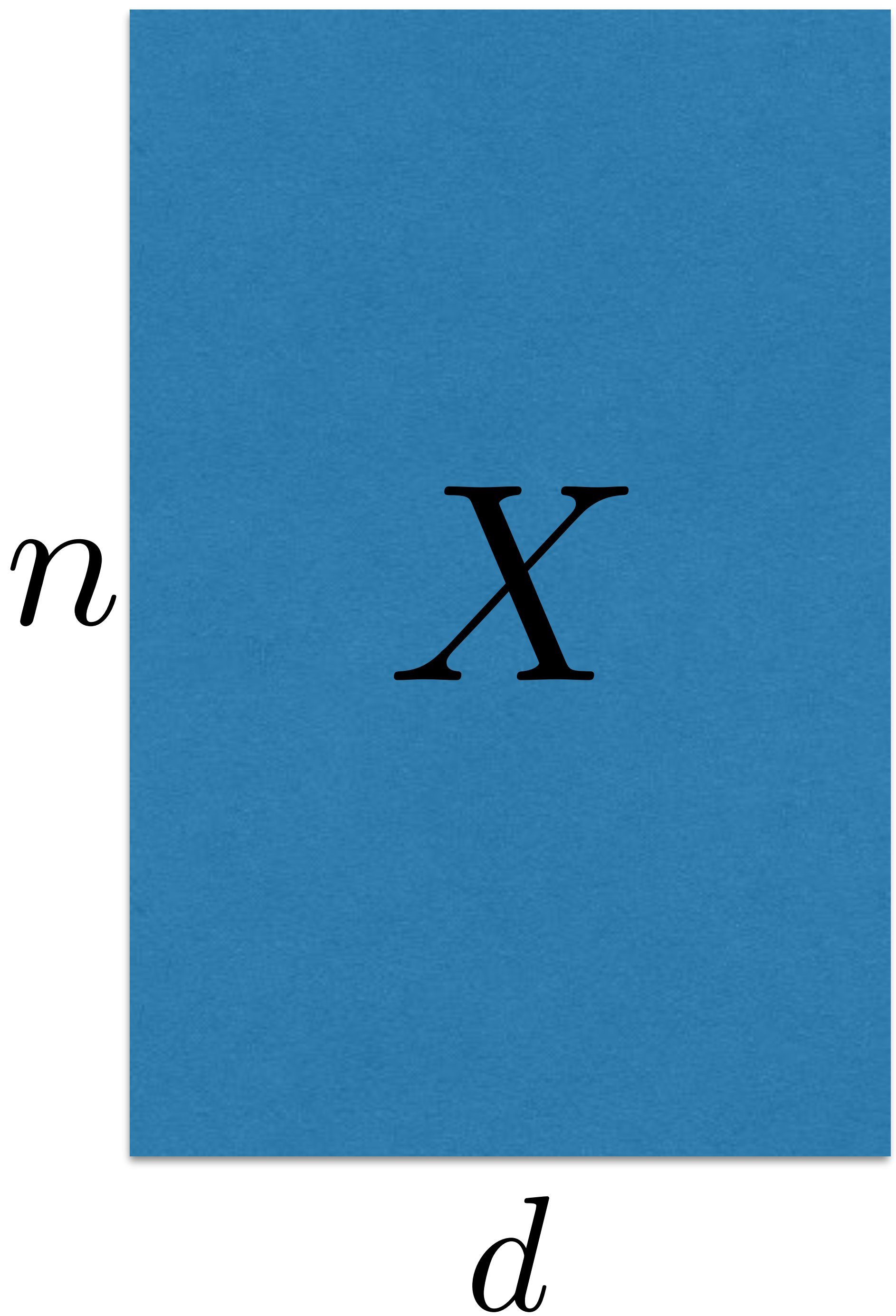
2. $\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} = \mathrm{cov}\left(X \right)$

3. $W = \mathrm{eigs}\left(\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}, K \right)$

4. $Y_1 = X_1 - \mu_1 \times W$

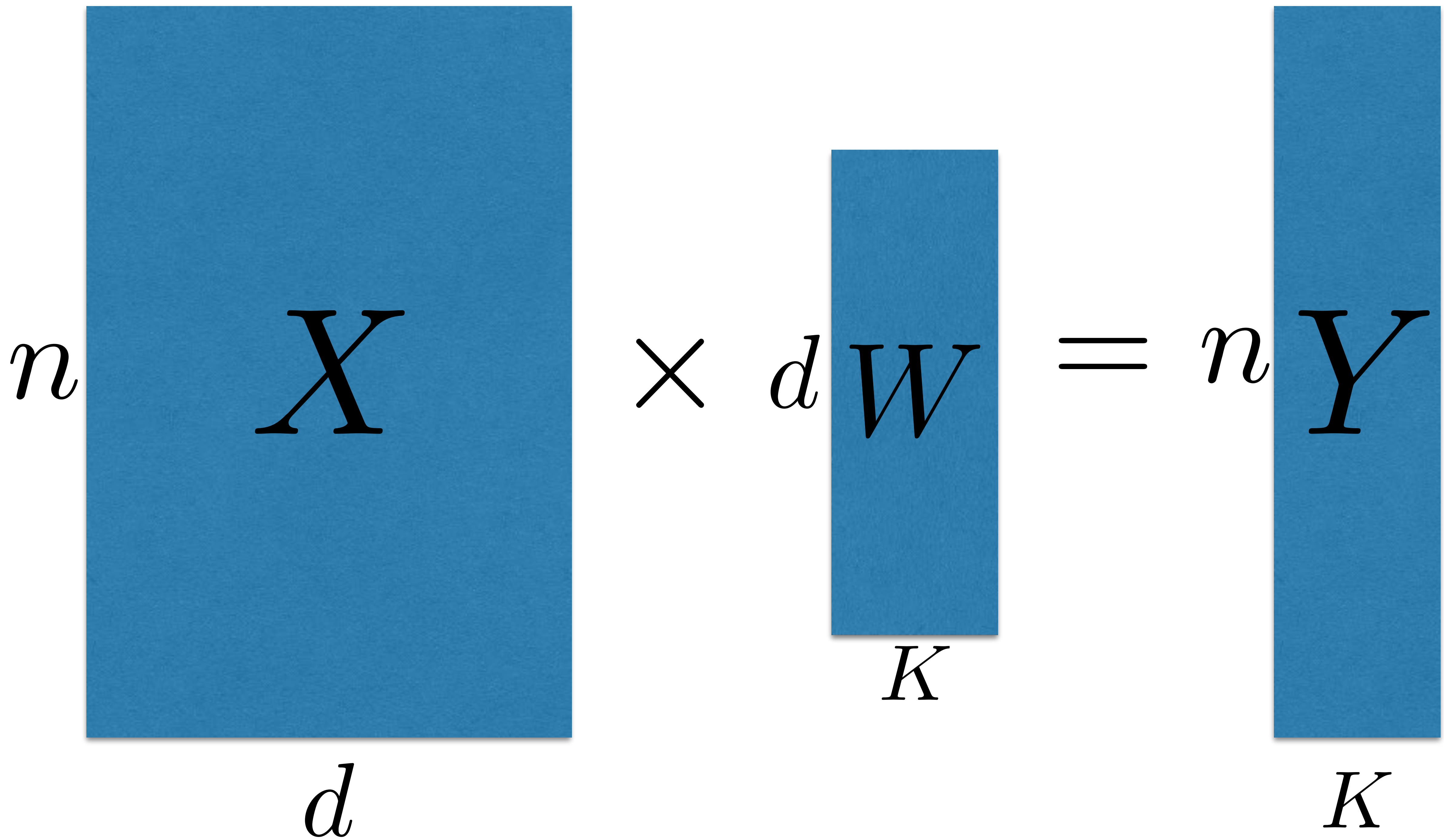
Linear Projections

Linear Projections

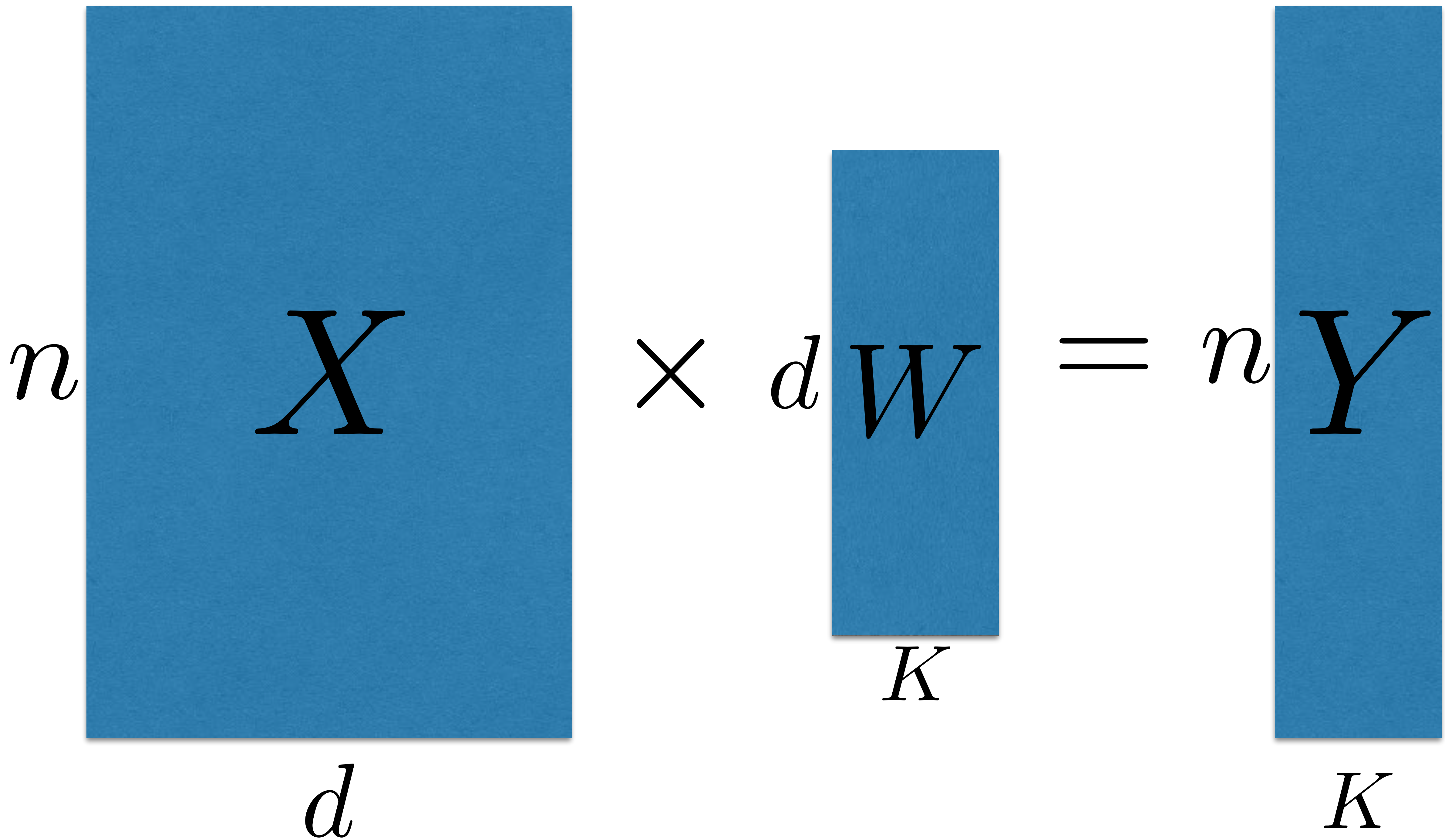


Linear Projections

LINEAR PROJECTIONS



Linear Projections



Works when data lies in a low dimensional linear sub-space

Kernel Trick

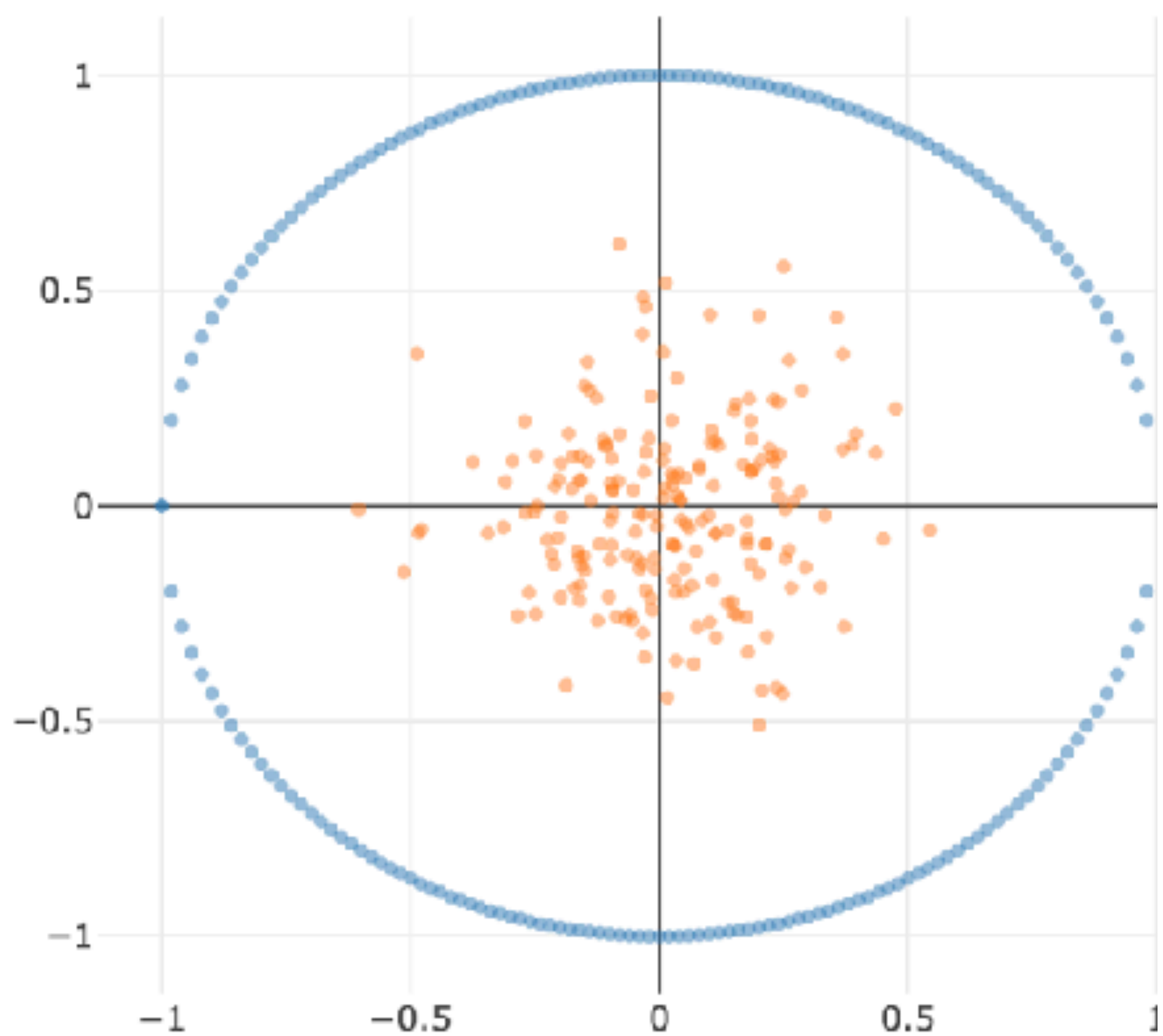
- We have have nice methods for linear dimensionality reduction
- Can we use this beyond the linear realm?

Kernel Trick

- Lift to higher dimensions (introduces non-linearity)
- Perform linear dimensionality reduction in this high dimensional space

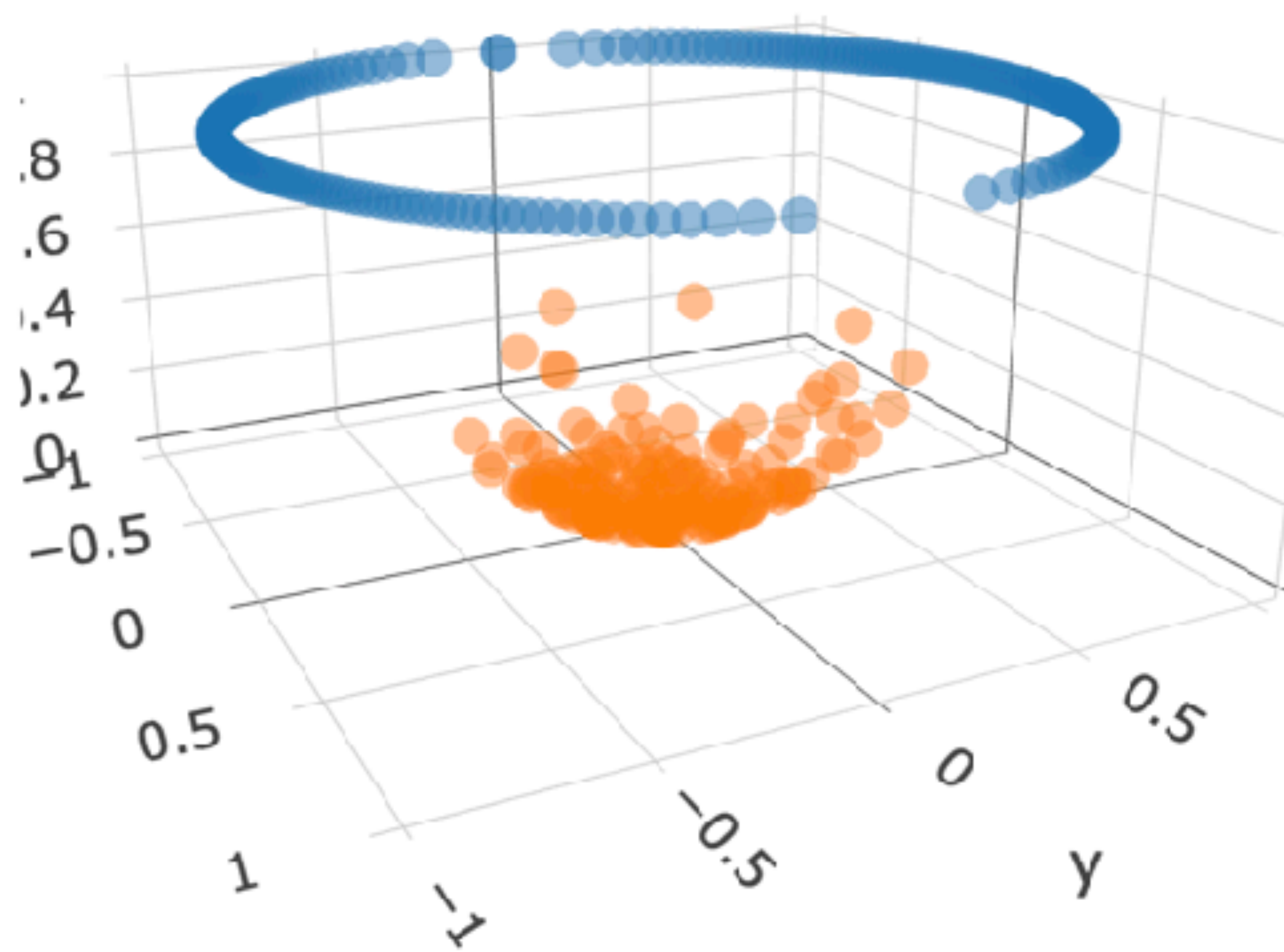
Demo

EXAMPLE



Original Data in 2D

(x,y)



Data Lifted to 3D

(x, y, x^2 + y^2)

A First Cut

- Given $\mathbf{x}_t \in \mathbb{R}^d$, the feature space vector is given by mapping

$$\Phi(\mathbf{x}_t) = (\mathbf{x}_t[1], \ldots, \mathbf{x}_t[d], \mathbf{x}_t[1] \cdot \mathbf{x}_t[1], \mathbf{x}_t[1] \cdot \mathbf{x}_t[2], \ldots, \mathbf{x}_t[d] \cdot \mathbf{x}_t[d], \ldots)^\top$$

- Enumerating products up to order K (ie. products of at most K coordinates) we can get degree K polynomials.
- However dimension blows up as d^K
- Is there a way to do this without enumerating Φ?

Kernel Trick

- Essence of Kernel trick:
  - **If** we can write down an algorithm only in terms of $\Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ for data points $\mathbf{x}_t$ and $\mathbf{x}_s$

Kernel Trick

- Essence of Kernel trick:
  - **If** we can write down an algorithm only in terms of $\Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ for data points $\mathbf{x}_t$ and $\mathbf{x}_s$
  - **Then** we don't need to explicitly enumerate $\Phi(\mathbf{x}_t)$'s but instead, compute $k(\mathbf{x}_t, \mathbf{x}_s) = \Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ (even if Φ maps to infinite dimensional space)

Kernel Trick

- Essence of Kernel trick:
  - **If** we can write down an algorithm only in terms of $\Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ for data points $\mathbf{x}_t$ and $\mathbf{x}_s$
  - **Then** we don't need to explicitly enumerate $\Phi(\mathbf{x}_t)$'s but instead, compute $k(\mathbf{x}_t, \mathbf{x}_s) = \Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ (even if Φ maps to infinite dimensional space)
- Example: RBF kernel $k(\mathbf{x}_t, \mathbf{x}_s) = \exp(-\sigma\|\mathbf{x}_t - \mathbf{x}_s\|_2^2)$, polynomial kernel $k(\mathbf{x}_t, \mathbf{x}_s) = \left(\mathbf{x}_t^\top \mathbf{y}_t\right)^p$

Kernel Trick

- Essence of Kernel trick:
  - **If** we can write down an algorithm only in terms of $\Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ for data points $\mathbf{x}_t$ and $\mathbf{x}_s$
  - **Then** we don't need to explicitly enumerate $\Phi(\mathbf{x}_t)$'s but instead, compute $k(\mathbf{x}_t, \mathbf{x}_s) = \Phi(\mathbf{x}_t)^\top \Phi(\mathbf{x}_s)$ (even if Φ maps to infinite dimensional space)
- Example: RBF kernel $k(\mathbf{x}_t, \mathbf{x}_s) = \exp(-\sigma \|\mathbf{x}_t - \mathbf{x}_s\|_2^2)$, polynomial kernel $k(\mathbf{x}_t, \mathbf{x}_s) = \left(\mathbf{x}_t^\top \mathbf{y}_t\right)^p$
- Kernel function measures similarity between points.

Kernel Trick

$(\mathbf{x}_t^\top \mathbf{y}_t)^p$

Kernel Trick

$$(\mathbf{x}_t^\top \mathbf{y}_t)^p = \sum_{k_1+k_2+\ldots+k_d=p} \binom{p}{k_1, k_2, \ldots, k_d} \prod_{j=1}^{d} (x_t[j] y_t[j])^{k_j}$$

Kernel Trick

$$
\begin{aligned}
(\mathbf{x}_t^\top \mathbf{y}_t)^p &= \sum_{k_1+k_2+\ldots+k_d=p} \binom{p}{k_1, k_2, \ldots, k_d} \prod_{j=1}^{d} (x_t[j] y_t[j])^{k_j} \\
&= \sum_{k_1+k_2+\ldots+k_d=p} \left(\sqrt{\binom{p}{k_1, k_2, \ldots, k_d}} \prod_{j=1}^{d} x_t[j]^{k_t} \right) \cdot \left(\sqrt{\binom{p}{k_1, k_2, \ldots, k_d}} \prod_{j=1}^{d} y_t[j]^{k_j} \right)
\end{aligned}
$$

Kernel Trick

$$(\mathbf{x}_t^\top \mathbf{y}_t)^p = \sum_{k_1+k_2+\ldots+k_d=p} \binom{p}{k_1, k_2, \ldots, k_d} \prod_{j=1}^{d} (x_t[j] y_t[j])^{k_j}$$

$$= \sum_{k_1+k_2+\ldots+k_d=p} \left(\sqrt{\binom{p}{k_1, k_2, \ldots, k_d}} \prod_{j=1}^{d} x_t[j]^{k_t} \right) \cdot \left(\sqrt{\binom{p}{k_1, k_2, \ldots, k_d}} \prod_{j=1}^{d} y_t[j]^{k_j} \right)$$

$$\Phi(\mathbf{x})^\top = \left(\ldots, \sqrt{\binom{p}{k_1, k_2, \ldots, k_d}} \prod_{j=1}^{d} x_t[j]^{k_t}, \ldots \right)_{k_1+k_2+\ldots+k_d=p}$$

Key Idea:

If an algorithm only depends on inner products, we can simply replace inner product in x space by inner product in $\phi(x)$ space

Can we write PCA so it only depends on inner products?

Lets Rewrite PCA

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{x}_t^\top W_k\right)\mathbf{x}_t$$

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n}\left(\mathbf{x}_t^\top W_k\right)\mathbf{x}_t$$

But $\mathbf{x}_t^\top W_k = \mathbf{y}_t[k]$

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n} \left(\mathbf{x}_t^\top W_k\right) \mathbf{x}_t$$

But $\mathbf{x}_t^\top W_k = \mathbf{y}_t[k]$

$$\lambda_k W_k = \frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

$$= \frac{1}{\lambda_k}\left(\frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t\right)^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

$$= \frac{1}{\lambda_k}\left(\frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t\right)^\top \mathbf{x}_s$$

$$= \frac{1}{n\lambda_k}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$
\begin{aligned}
\mathbf{y}_s[k] &= W_k^\top \mathbf{x}_s \\
&= \frac{1}{\lambda_k}\left(\frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t\right)^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n} \mathbf{y}_t[k]\tilde{K}_{s,t}
\end{aligned}
$$

Lets Rewrite PCA

$$\begin{aligned}
\mathbf{y}_s[k] &= W_k^\top \mathbf{x}_s \\
&= \frac{1}{\lambda_k}\left(\frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t\right)^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n} \mathbf{y}_t[k]\tilde{K}_{s,t}
\end{aligned}$$

Where $\tilde{K}_{s,t} = \mathbf{x}_t^\top \mathbf{x}_s$ is the kernel matrix for centered data

LETS REWRITE PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top \left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

Also we have, $1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top \left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top = \frac{1}{n\lambda_k}\|\mathbf{y}[k]\|^2$$

LETS REWRITE PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top = \frac{1}{n\lambda_k}\|\mathbf{y}[k]\|^2$$

Hence $P_k = \mathbf{y}[k]/\sqrt{n\lambda_k}$ is an eigenvector of $\tilde{K}$ with eigen value $\gamma_k = n\lambda_k$

Rewritting PCA

- We assumed centered data, what if its not,

REWRITTING PCA

- We assumed centered data, what if its not,

$$\tilde{K}_{s,t} = \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u)\right)^\top \left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)$$

Rewritting PCA

- We assumed centered data, what if its not,

$$\tilde{K}_{s,t} = \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u)\right)^\top \left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)$$

$$= \mathbf{x}_t^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_t$$

$$+ \frac{1}{n^2}\left(\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \left(\sum_{v=1}^{n}\mathbf{x}_v\right)$$

Rewritting PCA

- We assumed centered data, what if its not,

$$
\begin{aligned}
\tilde{K}_{s,t} &= \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u)\right)^\top \left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right) \\
&= \mathbf{x}_t^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_t \\
&\quad + \frac{1}{n^2}\left(\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \left(\sum_{v=1}^{n}\mathbf{x}_v\right) \\
&= \mathbf{x}_t^\top \mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u^\top \mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u^\top \mathbf{x}_t + \frac{1}{n^2}\sum_{u=1}^{n}\sum_{v=1}^{n}\mathbf{x}_u^\top \mathbf{x}_v
\end{aligned}
$$

Rewriting PCA

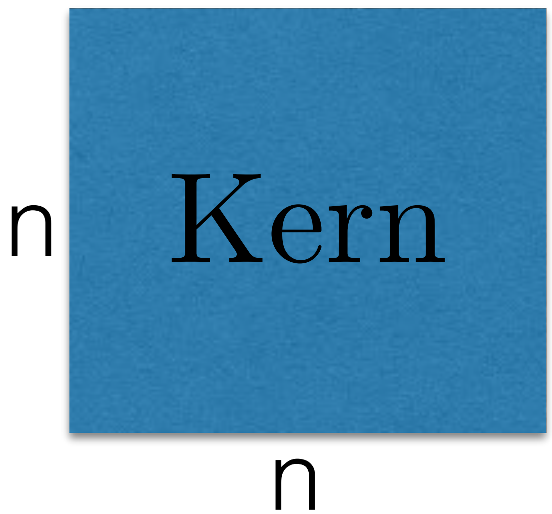
- Equivalently, if Kern is the matrix ($\text{Kern}_{t,s} = x_t^\top x_s$),

$$\tilde{K} = \text{Kern} - \frac{(\mathbf{1}_{n\times n} \times \text{Kern})}{n} - \frac{(\text{Kern} \times \mathbf{1}_{n\times n})}{n} + \frac{(\mathbf{1}_{n\times n} \times \text{Kern} \times \mathbf{1}_{n\times n})}{n^2}$$

Kernel PCA

KERNEL PCA

1.



$$= \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \dots & k(x_2, x_n) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ k(x_{n-1}, x_1) & k(x_{n-1}, x_2) & \dots & k(x_{n-1}, x_n) \\ k(x_n, x_1) & k(x_n, x_2) & \dots & k(x_n, x_n) \end{bmatrix}$$

Kernel PCA

1\.

$$\underset{n \times n}{\text{Kern}} = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \dots & k(x_2, x_n) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ k(x_{n-1}, x_1) & k(x_{n-1}, x_2) & \dots & k(x_{n-1}, x_n) \\ k(x_n, x_1) & k(x_n, x_2) & \dots & k(x_n, x_n) \end{bmatrix}$$

2\.

$$\underset{n \times n}{\tilde{K}} = \text{Kern} - \frac{1}{n}\left(\mathbf{1}\ \text{Kern} + \text{Kern}\ \mathbf{1}\right) + \frac{1}{n^2}\mathbf{1}\ \text{Kern}\ \mathbf{1}$$

Kernel PCA

Kernel PCA

3. $$\left[{}^{n}\underset{K}{P}, \gamma \right] = \mathrm{eigs}\left(\tilde{K}, K \right)$$

Kernel PCA

3. $\left[\underset{K}{{}^{n}P}, \gamma \right] = \mathrm{eigs}\left(\tilde{K}, K \right)$

4. $\underset{K}{{}^{n}Y} = \underset{K}{\left[P_1\sqrt{\gamma_1} \;\cdots\; P_K\sqrt{\gamma_K} \right]}$

Demo