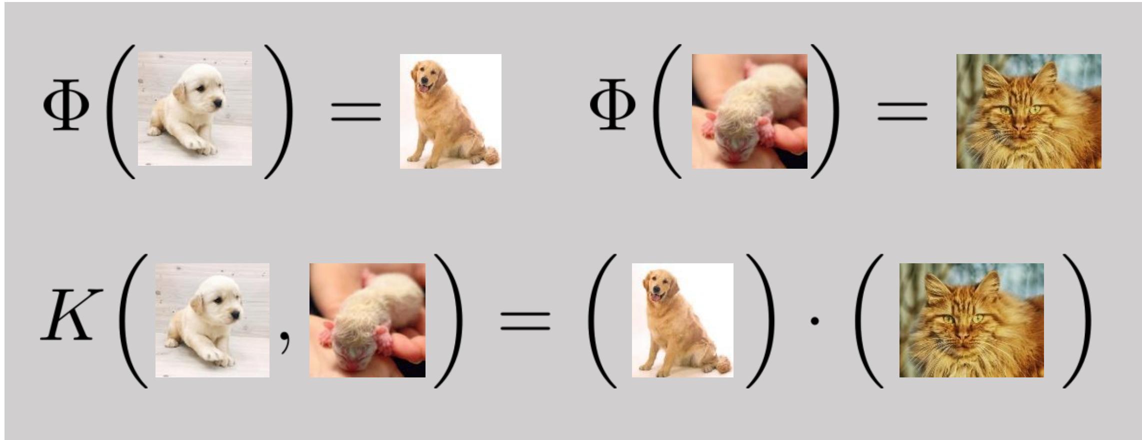


Kernel-based Learning: Theoretical Foundation

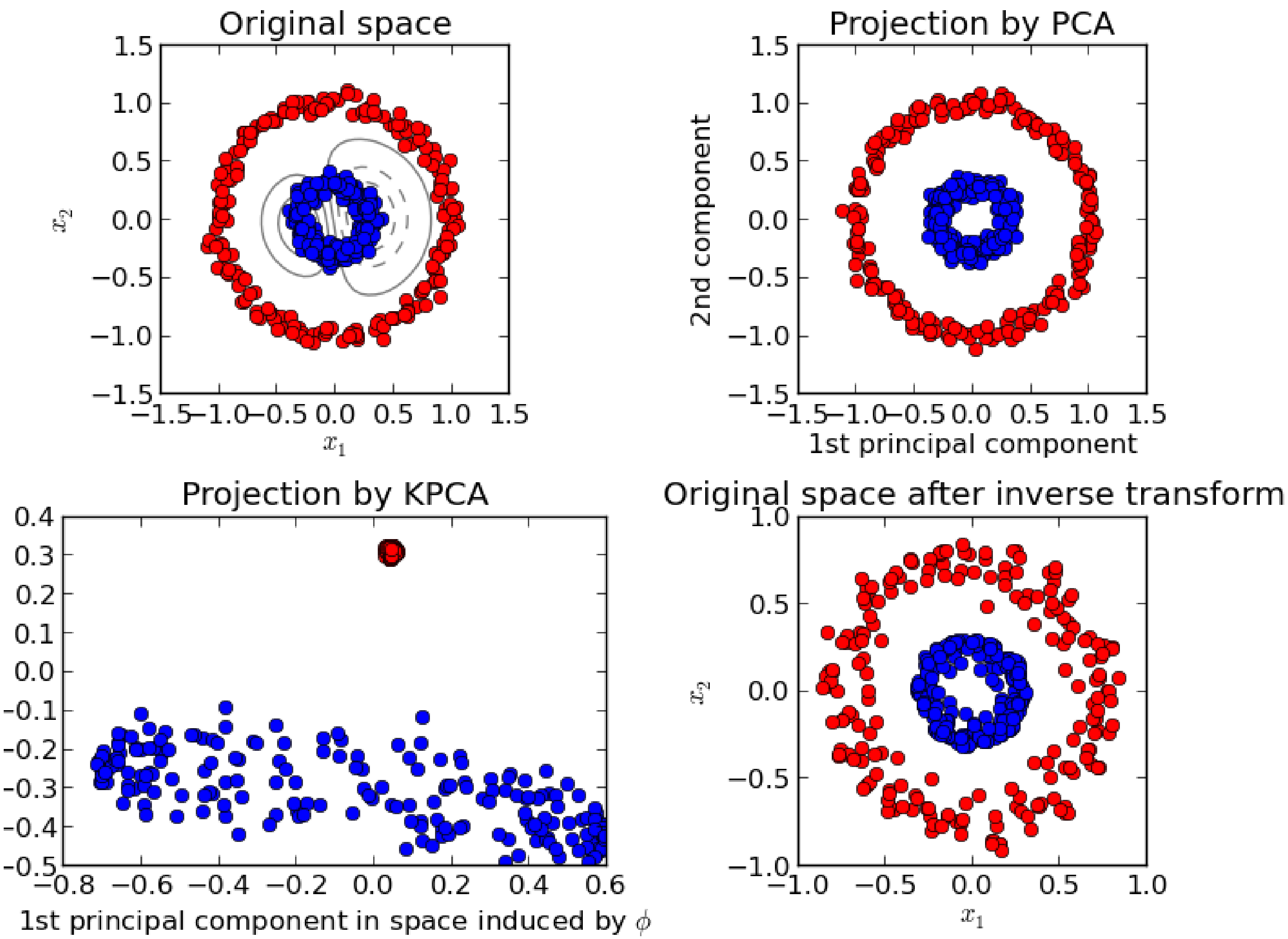
Pilsung Kang

School of Industrial Management Engineering

Korea University

Kernel PCA

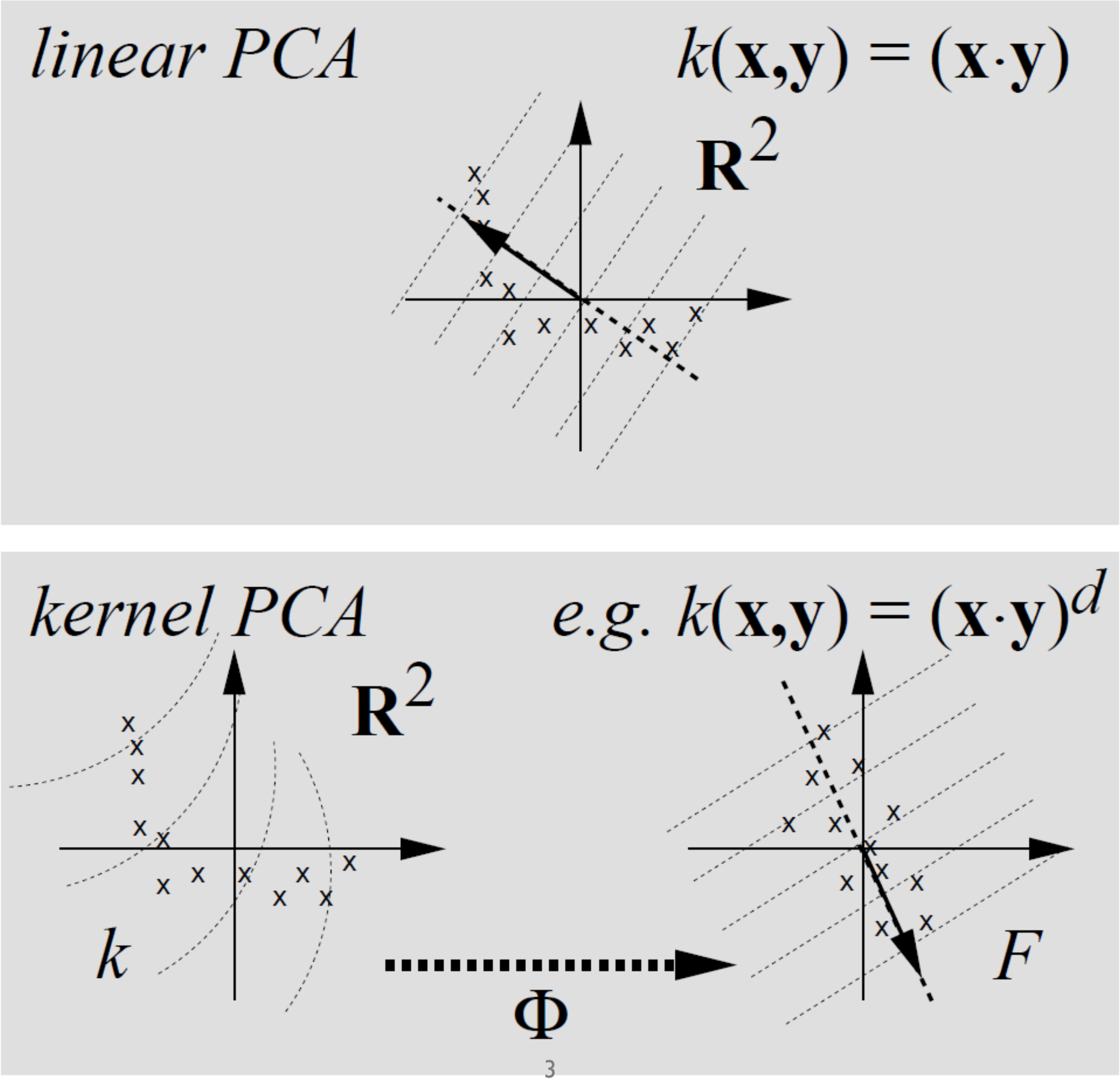
- What if the embedding is not linear?



Kernel PCA

Schölkopf et al. (1998)

- Motivation



Kernel PCA

- Kernel PCA Procedure
  - ✓ Assumption: the projected new features have zero mean

$$\mathbf{m}^{\Phi} = \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i) = \mathbf{0}$$

  - ✓ The covariance matrix of the projected features is M by M, calculated by

$$\mathbf{C}^{\Phi} = \frac{1}{N}\sum_{i=1}^{N}(\Phi(\mathbf{x}_i) - \mathbf{m}^{\Phi})(\Phi(\mathbf{x}_i) - \mathbf{m}^{\Phi})^T = \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^T$$

  - ✓ Its eigenvalues and eigenvectors are given by

$$\mathbf{C}^{\Phi}\mathbf{v}_k = \lambda_k \mathbf{v}_k$$

    - where k = 1. 2. …, M.

Kernel PCA

- Kernel PCA Procedure
  - ✓ From the previous two equations, we have

$$\frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)\big(\Phi(\mathbf{x}_i)^T\mathbf{v}_k\big) = \lambda_k\mathbf{v}_k$$

  - ✓ which can be rewritten as

$$\mathbf{v}_k = \frac{1}{N}\sum_{i=1}^{N}\alpha_{ki}\Phi(\mathbf{x}_i)$$

  - ✓ By substituting v_k above, we have

$$\frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^T\sum_{j=1}^{N}\alpha_{kj}\Phi(\mathbf{x}_j) = \lambda_k\sum_{i=1}^{N}\alpha_{kj}\Phi(\mathbf{x}_i)$$

Kernel PCA

✓ If we define the kernel function

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$$

✓ And multiply both side of the last equation in the previous page by $\Phi(\mathbf{x}_l)$

$$\frac{1}{N}\sum_{i=1}^{N} \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_i) \sum_{j=1}^{N} \alpha_{kj} \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j) = \lambda_k \sum_{i=1}^{N} \alpha_{kj} \Phi(\mathbf{x}_l)^T \Phi(\mathbf{x}_i)$$

$$\frac{1}{N}\sum_{i=1}^{N} \mathbf{K}(\mathbf{x}_l, \mathbf{x}_i) \sum_{j=1}^{N} \alpha_{kj} \mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \lambda_k \sum_{i=1}^{N} \alpha_{kj} \mathbf{K}(\mathbf{x}_l, \mathbf{x}_i)$$

✓ Then, we can use the matrix notation

$$\mathbf{K}^2 \boldsymbol{\alpha}_k = \lambda_k N \mathbf{K} \boldsymbol{\alpha}_k$$

- Where $\boldsymbol{\alpha}_k$ is the N-dimensional column vector $\boldsymbol{\alpha}_k = (\alpha_{k1}, \alpha_{k2}, \cdots, \alpha_{kN})^T$

Kernel PCA

✓ The eigenvector problem becomes

$$\mathbf{K}\boldsymbol{\alpha}_k = \lambda_k N \boldsymbol{\alpha}_k$$

✓ And the resulting kernel PCA can be calculated using

$$y_k(\mathbf{x}) = \Phi(\mathbf{x})^T \mathbf{v}_k = \sum_{i=1}^{N} \alpha_{ki} \mathbf{K}(\mathbf{x}, \mathbf{x}_i)$$

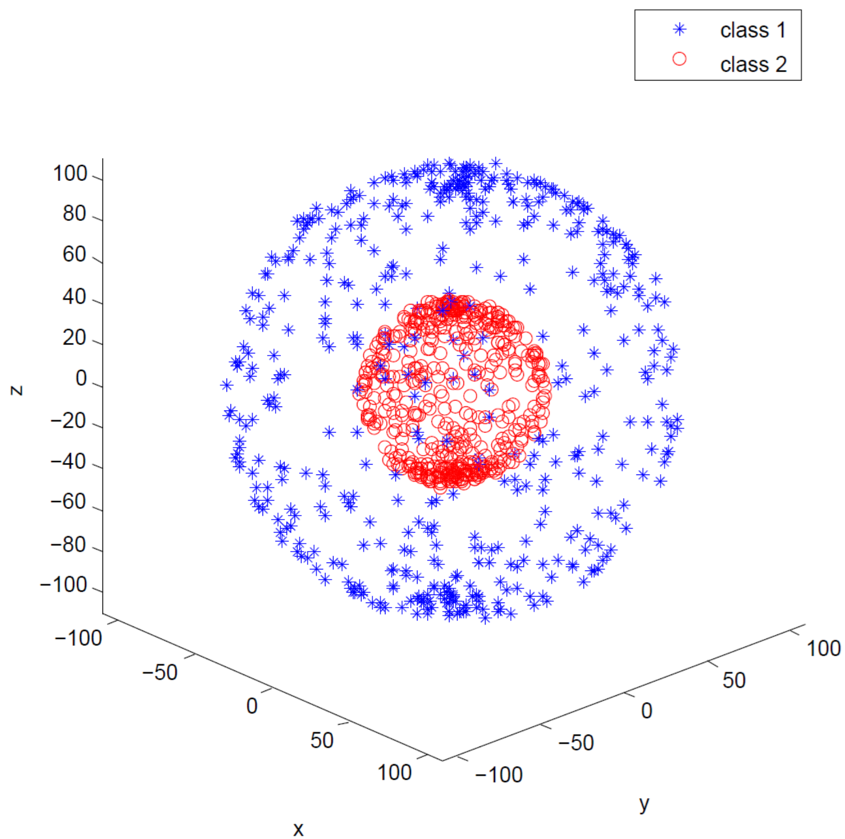
✓ If the projected dataset does not have zero mean, use the Gram matrix

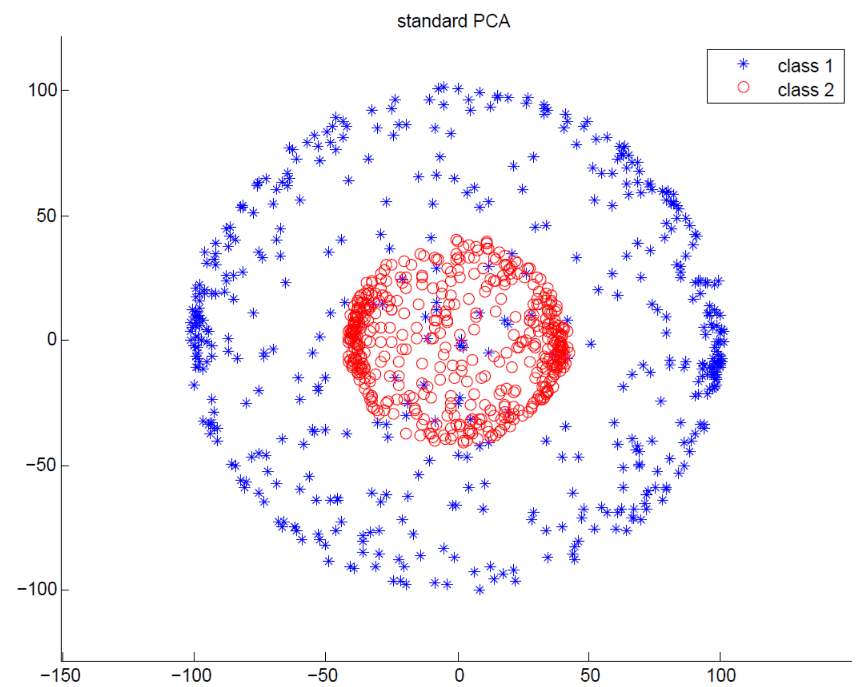
$$\begin{aligned}\widetilde{\mathbf{K}} &= (\mathbf{I} - \mathbf{1}_N)\mathbf{K}(\mathbf{I} - \mathbf{1}_N) \\ &= \mathbf{K} - \mathbf{1}_N\mathbf{K} - \mathbf{K}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}\mathbf{1}_N\end{aligned}$$

- Where $\mathbf{1}_N$ is the N by N matrix with all elements equal to 1/N

Kernel PCA

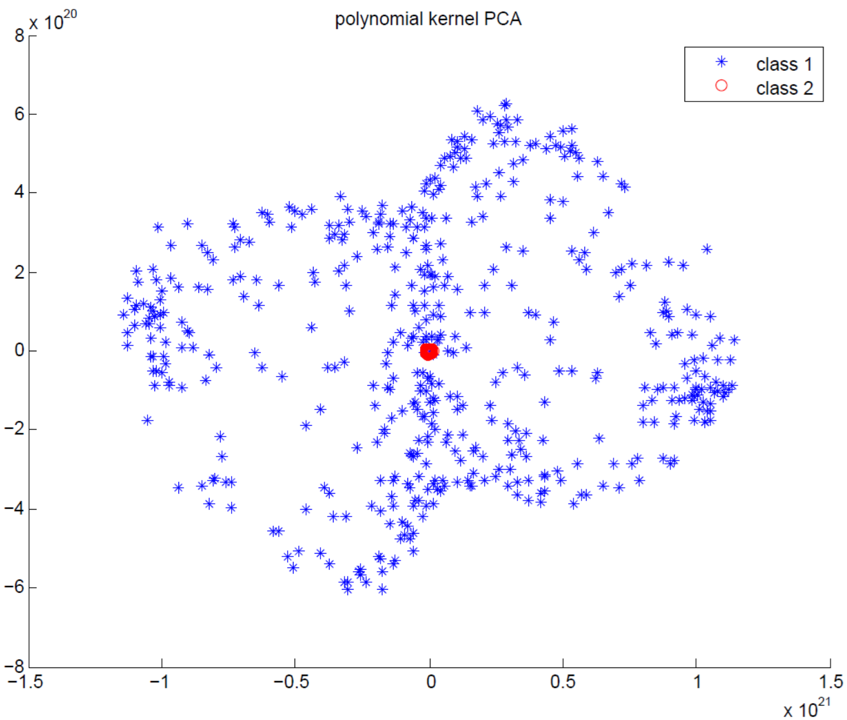
- PCA for two sphere sets

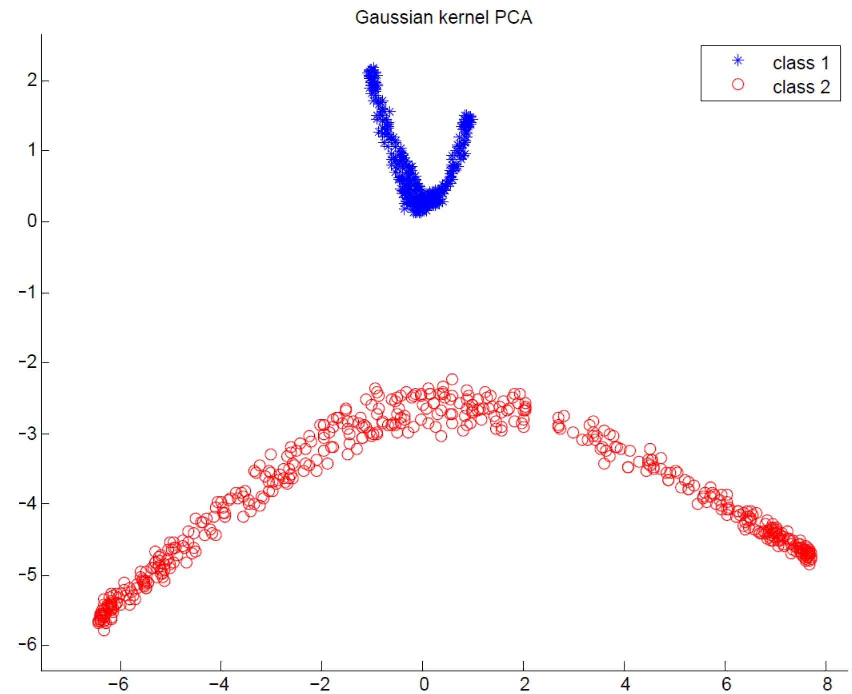




Kernel PCA

- KPCA for two sphere sets



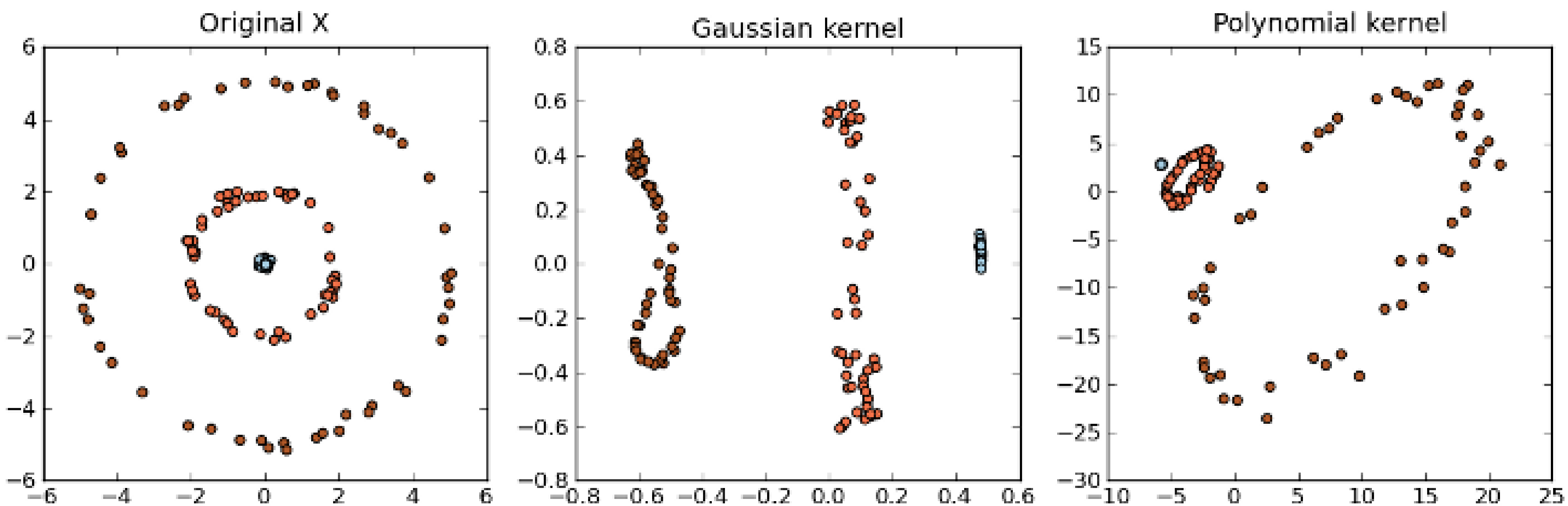


고려대학교
KOREA UNIVERSITY

DSBA
Data Science & Business Analytics

Kernel PCA

- Projection result according to different kernel types

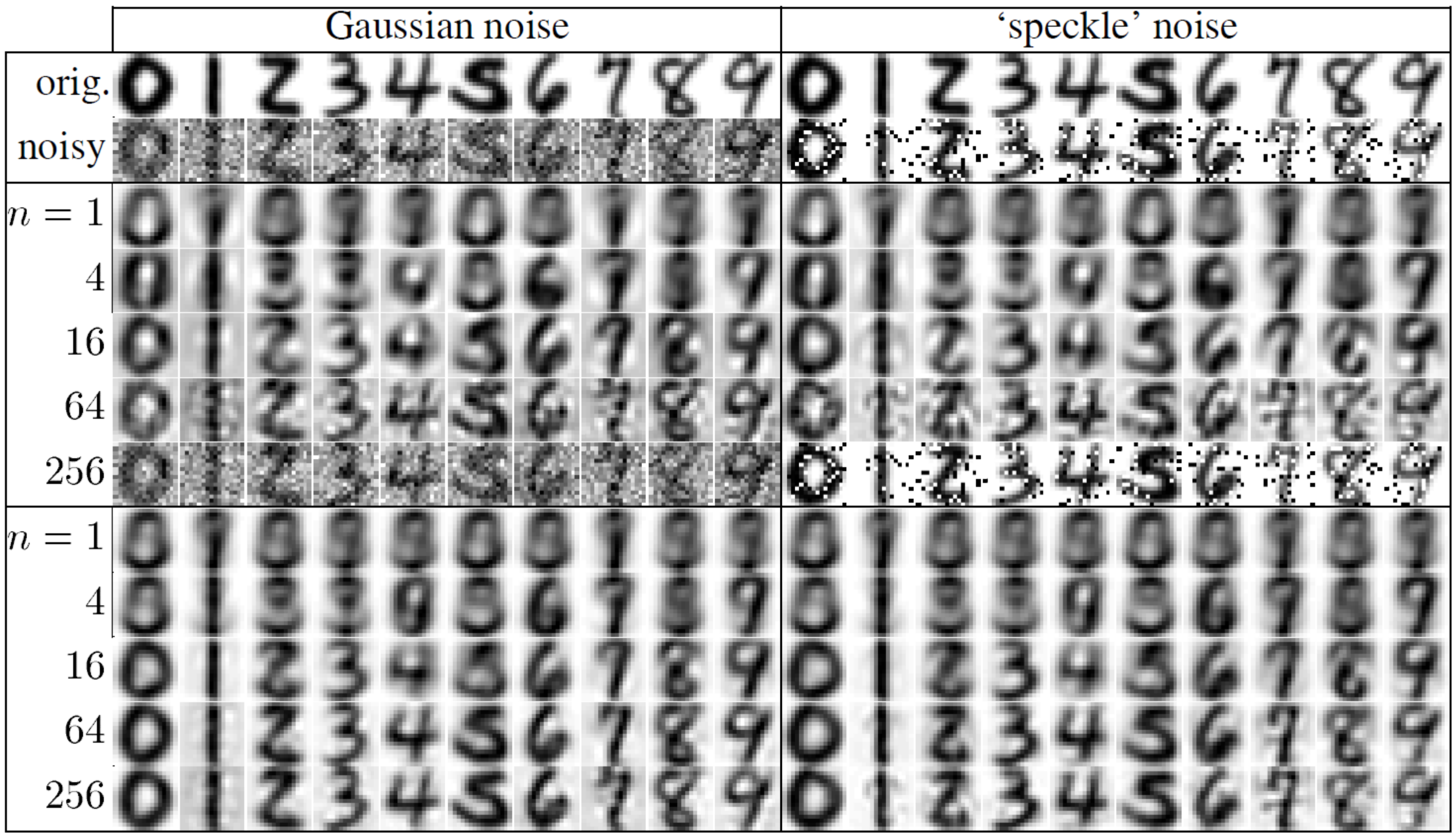


Kernel PCA: Application

- De-noising images

PCA

KPCA




Q & A

References

Research Papers

- Mika, S., Schölkopf, B., Smola, A. J., Müller, K. R., Scholz, M., and Rätsch, G. (1998). Kernel PCA and de-noising in feature spaces. In Proceedings of Neural Information Processing Systems (NIPS'98).
- Müller, K., Mika, S., Rätsch, G., Tsuda, K., and Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. IEEE Transactions on Neural Networks 12(2): 181-201.
- Schölkopf, B., Smola, A., and Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. Neural computation 10(5): 1299-1319.