Kernel PCA and de-noising in feature spaces by S. Mika et al.

Federico Baldassarre, Zacharie Brodard, Alfredo Fanghella, Lucas Rodés i Guirao

KTH Royal Institute of Technology

{*fedbal, zacharie, ajfv, lucasrg*} *@kth.se*

16 January, 2017

Outline

Introduction

- **Kernel PCA** for data reconstruction and de-noising
- Approximate **pre-images** using **Gaussian Kernels**.

Kernel PCA

Consider the data set $\mathcal{X} = \{x_1, \ldots, x_\ell\}$ with $x_i \in \mathbb{R}^N$.

- Kernel PCA maps $\mathcal{X}$ into a higher dimensional feature space $\mathbf{F}$, i.e.

$$\mathbf{\Phi} : \mathbb{R}^N \to \mathbf{F}.$$

  $\mathbf{\Phi}(\boldsymbol{x}_i)$ is known as the $\mathbf{\Phi}$-image of $\boldsymbol{x}_i$.

- It then performs linear PCA in $\mathbf{F}$ using a projection operator onto the first n principal components in $\mathbf{F}$ as $\mathrm{P_n}$.

Reconstruction

Given the test point $\boldsymbol{t} \in \mathbb{R}^N$

- We reconstruct its $\boldsymbol{\Phi}$-image as $\mathrm{P_n}\boldsymbol{\Phi}(\boldsymbol{t})$.
- We then find its **pre-image** $\boldsymbol{z} \in \mathbb{R}^N$, such that $\boldsymbol{\Phi}(\boldsymbol{z}) \approx \mathrm{P_n}\boldsymbol{\Phi}(\boldsymbol{t})$ by minimizing

$$||\boldsymbol{\Phi}(\boldsymbol{z}) - \mathrm{P_n}\boldsymbol{\Phi}(\boldsymbol{t})||^2.$$

The Kernel Trick

- Using the **Kernel Trick** we can avoid working in high dimensional spaces
- Using **Gaussian kernels** it is possible to devise an iterative scheme to obtain an approximate pre-image.

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\Phi}(\boldsymbol{x}_i) \cdot \boldsymbol{\Phi}(\boldsymbol{x}_j). = \exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/c)$$

Toy example: 11 Gaussians

- 11 spherical gaussians with random centers in $[-1, 1]^{10}$
- Sample 100 points from each for training data
- Sample 33 points from each for test data
- Denoise the test data using kernel and linear PCA trained on training data

Results

σ	$n = 1$	2	3	4	5	6	7	8	9
0.05	2058.42	1238.36	846.14	565.41	309.64	170.36	125.97	104.40	92.23
0.1	10.22	31.32	21.51	29.24	27.66	23.53	29.64	40.07	63.41
0.2	0.99	1.12	1.18	1.50	2.11	2.73	3.72	5.09	6.32
0.4	1.07	1.26	1.44	1.64	1.91	2.08	2.22	2.34	2.47
0.8	1.23	1.39	1.54	1.70	1.80	1.96	2.10	2.25	2.39

Table: Their results

σ	$n = 1$	2	3	4	5	6	7	8	9
0.05	496.24	1614.73	1816.32	1598.64	972.89	623.70	364.37	153.05	90.50
0.1	178.33	752.52	803.60	586.85	396.99	259.77	152.16	104.50	89.89
0.2	37.65	63.62	18.41	5.16	3.35	2.82	2.73	2.30	3.43
0.4	6.03	4.06	1.77	1.78	1.87	2.08	5.49	9.24	9.54
0.8	0.62	0.98	1.25	1.54	1.83	1.80	1.75	1.57	1.53

Table: Our results

Greater than 1: Kernel PCA performs better

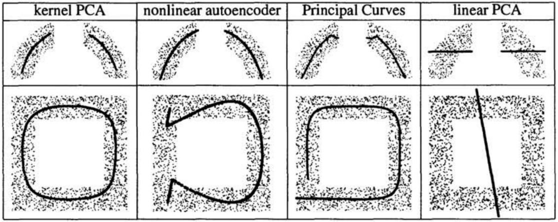
Toy example: De-noising

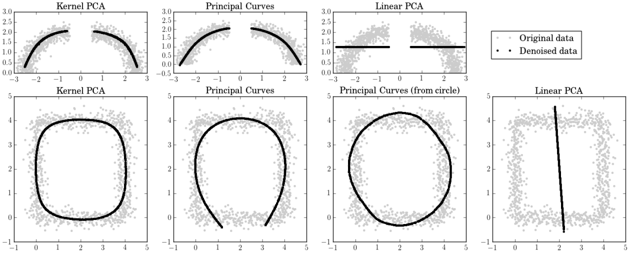
Figure: Noisy shapes in two dimensional plane

The intention is to compare de-noising results using Kernel PCA and other algorithms.

- Synthetic data is generated starting from different shapes in the plane and adding noise.
- Using two-dimensional data allows for a simple subjective analysis of the results by visual inspection.

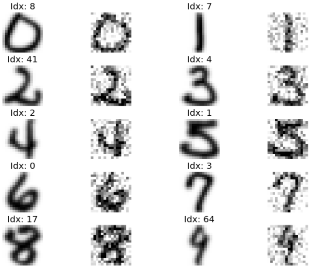
Results


kernel PCA
nonlinear autoencoder
Principal Curves
linear PCA


Kernel PCA
Principal Curves
Linear PCA
Original data
Denoised data
Kernel PCA
Principal Curves
Principal Curves (from circle)
Linear PCA

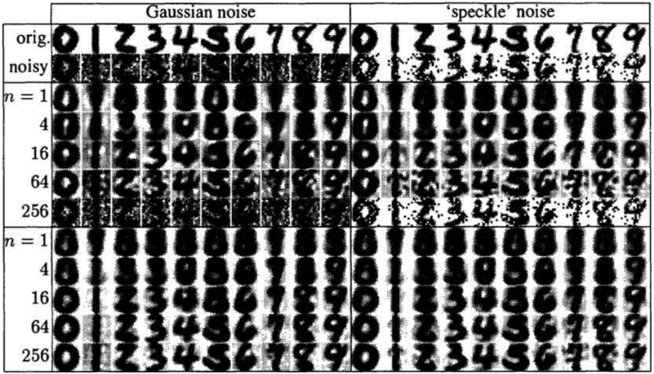
USPS Data set

- 16x16 labeled images of handwritten digits
- Scanned by the U.S. Postal Service
- A total of 9298 observations:
  - 7291 for training
  - 2007 for testing
- Deslanted and size-normalized
- Downloaded through `scikit-learn`
- Manually added noise:
  - Gaussian noise ($\mu = 0, \sigma = 0.5$)
  - Speckle noise ($p = 0.4$)



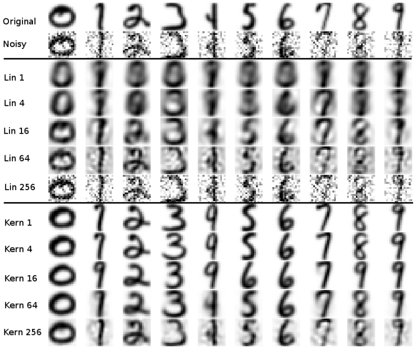
The first occurrence of each digit and its noisy version

Denoising: Linear vs Kernel PCA (their results)

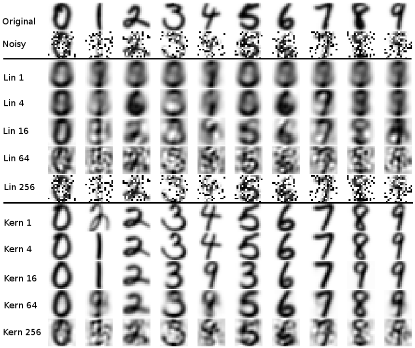


- For each digit select 300 "clean" samples
- Run Linear PCA and Kernel PCA on the noisy test classes with a variable number components for reconstruction

Denoising: Linear vs Kernel PCA (our results)

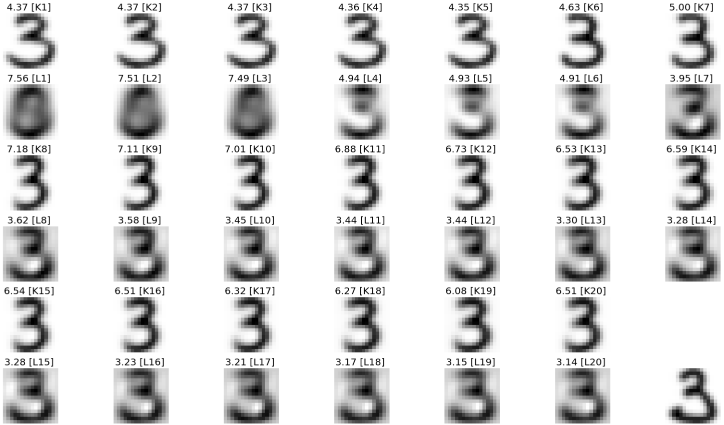


Linear and Kernel based denoising of Gaussian noise



Linear and Kernel based denoising of Speckle noise

Reconstruction: Linear vs Kernel PCA



- Number of features in the range [1,20]
- Euclidean distance from the target (bottom right)

Conclusion

- The results obtained by Mika *et al.* were reproduced with minor divergences
- Kernel PCA works well for extracting non-linear features
- Overhead when compared to Linear PCA:
  - Time complexity: needs to compute $k(t, x_i)\ \forall i$
  - Spatial complexity: need to store all training data

Thank you

References

Person, K (1901)
On Lines and Planes of Closest Fit to System of Points in Space.
Philiosophical Magazine, 2, 559-572

Hotelling, Harold (1993)
Analysis of a complex of statistical variables into principal components.
Journal of educational psychology, 24 5, 417.

Schölkopf, Bernhard and Smola, Alexander and Müller, Klaus-Robert (1997)
Kernel principal component analysis
International Conference on Artificial Neural Networks, 583–588

Mika, Sebastian and Schölkopf, Bernhard and Smola, Alexander J and Müller, Klaus-Robert and Scholz, Matthias and Rätsch, Gunnar (1998)
Kernel PCA and De-Noising in Feature Spaces.
NIPS, 11, 536–542.

References

Tipping, Michael E and Bishop, Christopher M (1999)
Probabilistic principal component analysis
Journal of the Royal Statistical Society: Series B (Statistical Methodology), 61 3, 611–622

Schölkopf, Bernhard and Smola, Alexander and Müller, Klaus-Robert (1998)
Nonlinear component analysis as a kernel eigenvalue problem
Neural computation, 10 5, 1299–1319

Zou, Hui and Hastie, Trevor and Tibshirani, Robert (2006)
Sparse principal component analysis
Journal of computational and graphical statistics, 15 2, 265–286

Evangelista, Paul F and Embrechts, Mark J and Szymanski, Boleslaw K (2007)
Some properties of the Gaussian kernel for one class learning
International Conference on Artificial Neural Networks, 269–278