Connection of Local Linear Embedding, ISOMAP, and Kernel Principal Component Analysis

Alvina Goh

Vision Reading Group

13 October 2005

Kernel Principal Component Analysis

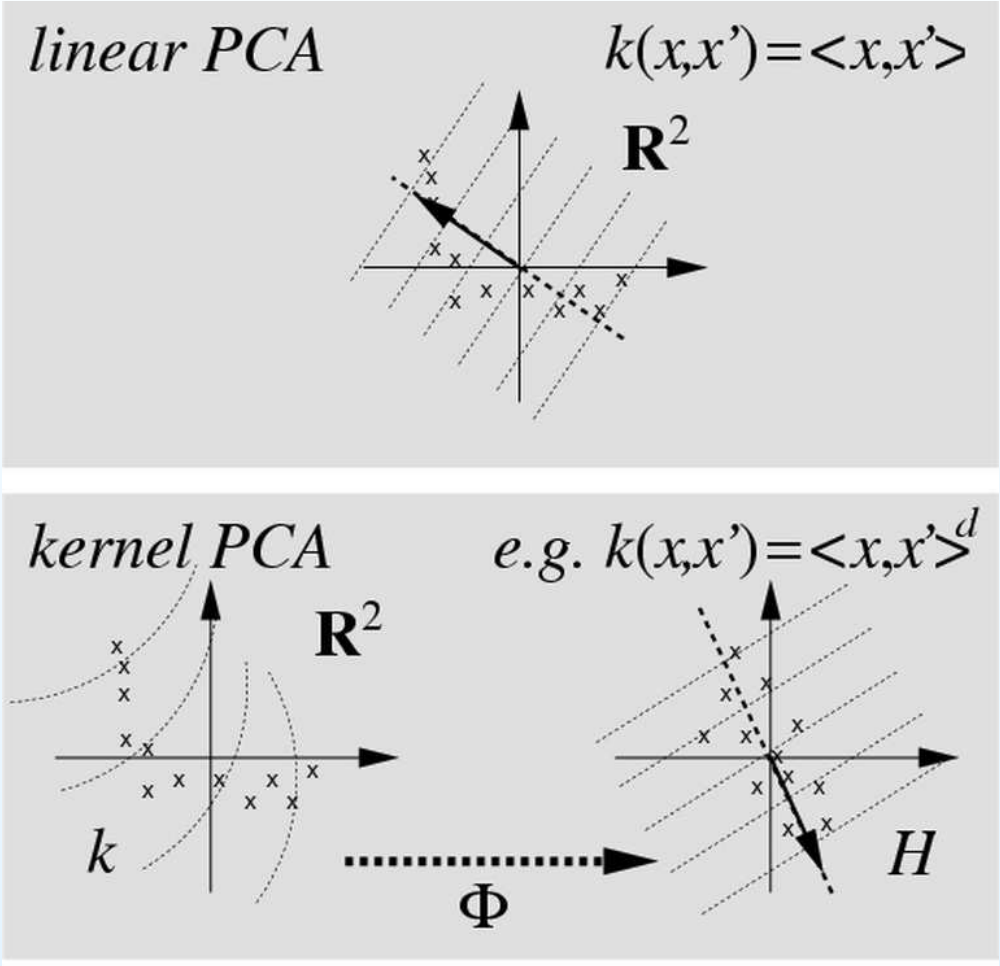


Figure 1: Basic idea of kPCA

Kernel Principal Component Analysis

- **Positive definite kernel** k
  Given a nonempty set $\mathcal{X}$ and a positive definite kernel k, there exists a map $\Phi : \mathcal{X} \mapsto \mathcal{H}$ into a dot product space $\mathcal{H}$ such that for all $x, x' \in \mathcal{X}$, $\langle \Phi(x), \Phi(x') \rangle = k(x, x')$.
  Thus k is a nonlinear similarity measure.
- Given centered data $x_1, ..., x_m \in \mathcal{X}$, kPCA computes the principal components of the points $\Phi(x_1), ..., \Phi(x_m) \in \mathcal{H}$ (in feature space).
- Consider the covariance matrix in $\mathcal{H}$,

$$\mathbf{C} = \frac{1}{m} \sum_{i=1}^{m} \Phi(x_i) \Phi(x_i)^T \qquad (1)$$

Kernel Principal Component Analysis

- We now have to find eigenvalues $\lambda \geq 0$ and nonzero eigenvectors $\mathbf{v} \in \mathcal{H} \backslash \{0\}$ satisfying

$$\lambda \mathbf{v} = \mathbf{C}\mathbf{v} \tag{2}$$

  By substituting $\mathbf{C}$, we observe that all solutions $\mathbf{v}$ with $\lambda \neq 0$ lie in the span of $\Phi(\cdot)$.

- First observation: We may consider the set of equations

$$\lambda \langle \Phi(x_n), \mathbf{v} \rangle = \langle \Phi(x_n), \mathbf{C}\mathbf{v} \rangle \tag{3}$$

  for all $n = 1, ..., m$.

- Second observation: There exist coefficients $\alpha_i, i = 1, ..., m$ such that

$$\mathbf{v} = \sum_{i=1}^{m} \alpha_i \Phi(x_i) \tag{4}$$

Kernel Principal Component Analysis

- Combining the two observations, we get

$$\lambda \sum_{i=1}^{m} \alpha_i \langle \Phi(x_n), \Phi(x_i) \rangle = \frac{1}{m} \sum_{i=1}^{m} \alpha_i \left\langle \Phi(x_n), \sum_{j=1}^{m} \Phi(x_j) \langle \Phi(x_j), \Phi(x_i) \rangle \right\rangle \tag{5}$$

  for all $n = 1, .., m$.

- With the $m \times m$ Gram matrix is given as $K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle$, we rewrite the above equation as

$$m\lambda K \boldsymbol{\alpha} = K^2 \boldsymbol{\alpha} \tag{6}$$

$$\Rightarrow m\lambda \boldsymbol{\alpha} = K \boldsymbol{\alpha} \tag{7}$$

  where $\boldsymbol{\alpha}$ is the column vector with entries $\alpha_1, ..., \alpha_m$.

Kernel Principal Component Analysis

- As analogous to PCA, we consider the largest p eigenvectors, and $\lambda_1, ...\lambda_p$ are all non-zero eigenvalues.
- Normalization of $\boldsymbol{\alpha}$ by imposing for all $n = 1, ..., p$,

$$1 = \langle \mathbf{v}^n, \mathbf{v}^n \rangle = \sum_{i,j=1}^{m} \alpha_i^n \alpha_j^n \langle \Phi(x_i), \Phi(x_j) \rangle = \sum_{i,j=1}^{m} \alpha_i^n \alpha_j^n K_{ij}$$

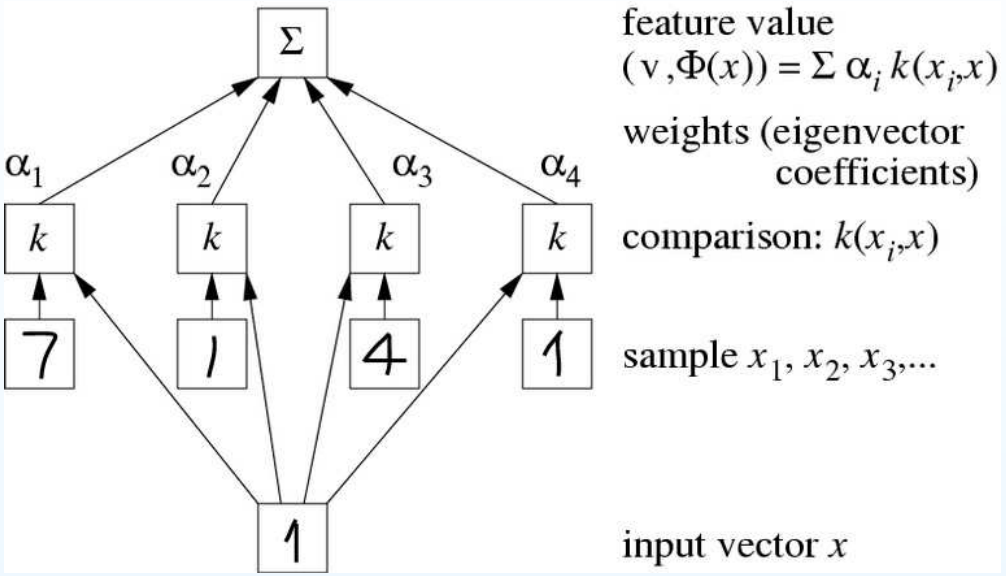
$$= \langle \boldsymbol{\alpha}^n, K\boldsymbol{\alpha}^n \rangle = \lambda_n \langle \boldsymbol{\alpha}^n, \boldsymbol{\alpha}^n \rangle \tag{8}$$

We want to compute projections onto the eigenvectors $\mathbf{v}^n$ in $\mathcal{H}$ ($n = 1, ..., p$). Let x be a test point with an image $\Phi(x)$ in $\mathcal{H}$. Then

$$\langle \mathbf{v}^n, \Phi(x) \rangle = \sum_{i=1}^{m} \alpha_i^n \langle \Phi(x_i), \Phi(x) \rangle \tag{9}$$

are the nonlinear principal components corresponding to Φ.

Summary of kPCA algorithm



1. Compute the Gram matrix $K_{ij} = k(x_i, x_j)_{ij}$.
2. Diagonalize K, and normalize eigenvector expansion coefficients α^n by requiring $\lambda_n \langle \boldsymbol{\alpha}^n, \boldsymbol{\alpha}^n \rangle = 1$
3. Compute projections onto the eigenvectors.

Final equation:

$$\langle \mathbf{v}^n, \Phi(x) \rangle = \frac{1}{\sqrt{\lambda^n}} \sum_{i=1}^{m} \alpha_i^n k(x_i, x)$$

Modification to account for zero mean

- The previous construction of kPCA assumes that the data in feature space has zero mean.
- It is not true for arbitrary data, and hence there is a need to subtract the mean

$$\frac{1}{m}\sum_i \Phi(x_i)$$

  from all the points $\Phi(\cdot)$
- This leads to a different eigenvalue problem with the following Gram matrix

$$K' = (I - ee^T)K(I - ee^T)$$

  where

$$e = \frac{1}{\sqrt{m}}(1, 1, ..., 1)^T$$

ISOMAP

kPCA and MDS

On a Connection between Kernel PCA and Metric Multidimensional Scaling

Christopher K. I. Williams
Division of Informatics
The University of Edinburgh
5 Forrest Hill, Edinburgh EH1 2QL, UK
c.k.i.williams@ed.ac.uk
http://anc.ed.ac.uk

Abstract

In this paper we show that the kernel PCA algorithm of Schölkopf *et al* (1998) can be interpreted as a form of metric multidimensional scaling (MDS) when the kernel function $k(\mathbf{x}, \mathbf{y})$ is isotropic, i.e. it depends only on $||\mathbf{x} - \mathbf{y}||$. This leads to a metric MDS algorithm where the desired configuration of points is found via the solution of an eigenproblem rather than through the iterative optimization of the stress objective function. The question of kernel choice is also discussed.

ISOMAP

Recall ISOMAP= MDS on geodesic distances. Thus, one can write

$$K_{ISOMAP} = -\frac{1}{2}(I - ee^T)S(I - ee^T) \tag{10}$$

where S is the squared distance matrix. There is no theoretical guarantee that K_{ISOMAP} is positive definite.

There is, however, a theorem by Grimes and Donoho (Hessian LLE) that apply in the *continuum limit for a smooth manifold* which says the geodesic distance between points on the manifold will be proportional to Euclidean distance in the low-dimensional parameter space of the manifold. In the *continuum limit*, $(-S)$ will be conditional positive definite and so will K_{ISOMAP}. Hence, ISOMAP is a form of kPCA.

LLE

Recall in the last step of LLE, we find the smallest eigenvectors of the matrix $M = (I - W)(I - W^T)$.

Denote the maximum eigenvalue of M by λ_{max}, we define the matrix

$$K = \lambda_{max} I - M$$

NOTE

- $M = M^T$. Hence M is real symmetric.
- Positive definite matrix $\equiv$ All eigenvalues are positive
- For a eigenvector $\mathbf{v}$ of M, $M\mathbf{v} = \lambda\mathbf{v}$.
- Consider $K\mathbf{v} = (\lambda_{max} I - M)\mathbf{v} = (\lambda_{max} - \lambda)\mathbf{v}$.

By construction, K is positive definite.

LLE

$$K = \lambda_{max} I - M$$

The leading eigenvector of K is e and the coordinates of the eigenvectors $2, ..., d+1$ provide the LLE embedding.

If we project out e and use the eigenvectors $1, ..., d$ of the resulting matrix

$$(I - ee^T)K(I - ee^T)$$

we get the embedding vectors.

If we project out e and use the eigenvectors $1, ..., d$ of the resulting matrix

$$(I - ee^T)K(I - ee^T)$$

we get the embedding vectors.

LLE

Remember that LLE is invariant under scaling, translation and rotation. LLE embedding is equivalent to the kPCA projections up to a multiplication with $\sqrt{\lambda^n}$.

This corresponds to the whitening step which is performed in LLE in order to fix the scaling, but not normally in kPCA, where the scaling is determined by the variance of the data along the respective directions in $\mathcal{H}$.

Finally, there need not be an analytic form of a kernel k which gives rise to the LLE kernel matrix K. Hence, there need not be a feature map on the whole input domain.

But one can give a feature map defined on the training points by writing the SVD of K.

So far

- MDS, ISOMAP, LLE, and Laplacian eigenmaps can be interpreted as kPCA with special kernels.
- Note that the kernel matrix is only defined on the training data, unlike kPCA where one starts with the analytic form of the kernel and hence defining over entire space.
- Recap that MDS, ISOMAP, LLE, and Laplacian eigenmaps are manifold learning algorithms.

Question:

In classification algorithm such as Support Vector Machines, there is a set of kernels commonly used. These includes polynomial, Gaussian, sigmod (tanh), and B-spline kernels and they are used in "normal" kPCA.

In particular, consider the Gaussian kernel

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} = e^{-\beta\|x-x'\|^2}$$

and we see that it preserves the mapping property of "nearby inputs into nearby features".

Can we use the Gaussian kernel to do manifold learning?

Question:

In classification algorithm such as Support Vector Machines, there is a set of kernels commonly used. These includes polynomial, Gaussian, sigmod (tanh), and B-spline kernels and they are used in "normal" kPCA.

In particular, consider the Gaussian kernel

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} = e^{-\beta\|x-x'\|^2}$$

and we see that it preserves the mapping property of "nearby inputs into nearby features".

Can we use the Gaussian kernel to do manifold learning?

NO!

Answer:

Consider the Gaussian kernel

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} = e^{-\beta\|x-x'\|^2}$$

Not forgetting that $k(x, x')$ is the dot product of the input vectors in feature space, we see that if the two feature vectors are far away, $k(x, x') = 0$.

Physically, we are mapping distant patches of the manifold into orthogonal feature spaces. Thus, we can no longer unroll the swissroll.

Answer:

Consider the Gaussian kernel

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}} = e^{-\beta\|x-x'\|^2}$$

In addition, for every patch of radius $\beta^{-\frac{1}{2}}$, the Gaussian kernel create an additional dimension. Hence, impossible to do dimension reduction.

Physically, the Gaussian kernel is really doing dimension EXPANSION. This follows logically from the SVM school of thought of mapping inputs into higher dimensions in order to find a linear classifier in feature space. By creating orthogonal spaces for different patches, one can easily find a linear classifier. This is why the Gaussian kernel is commonly used in SVM whose purpose is to do clustering/classification.

Summary

What we covered so far

1. Principal Components Analysis PCA
2. Multi Dimensional Scaling MDS
3. ISOMAP
4. k-means
5. Locally Linear Embedding
6. Laplacian LLE
7. Hessian LLE
8. Kernel Principal Components Analysis

Topics not covered

Dimension Reduction: Semi Definite Programming

- Maximum variance unfolding
- Conformal eigenmaps

Dimension Reduction: Spectral Clustering

- Laplacian matrix used in Laplacian eigenmaps and normalized cuts.
- Manifold learning and clustering are linked because the clusters that spectral clustering manages to capture can be arbitrary curved manifolds as long as there is enough data to locally capture the curvature of the manifold.

Feature Extraction: Probabilistic PCA

- Probabilistic PCA
- Oriented PCA

References

- Ham, J., D.D. Lee, S. Mika and B. Schölkopf. *A kernel view of the dimensionality reduction of manifolds*. Proceedings of the Twenty-First International Conference on Machine Learning, 369-376.
- B. Scholkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- and many many more.