An Approximate Version of Kernel PCA

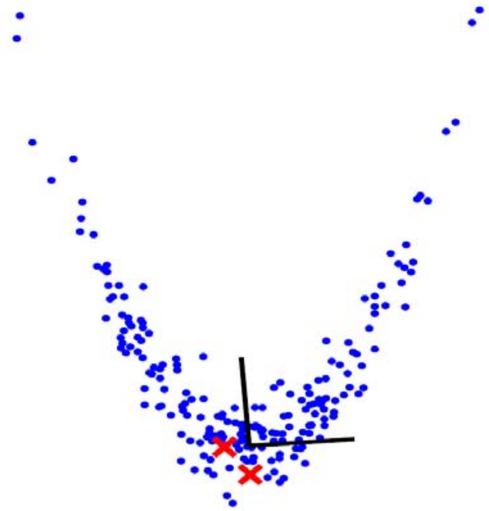
Shawn Martin

Sandia National Laboratories
Albuquerque, NM

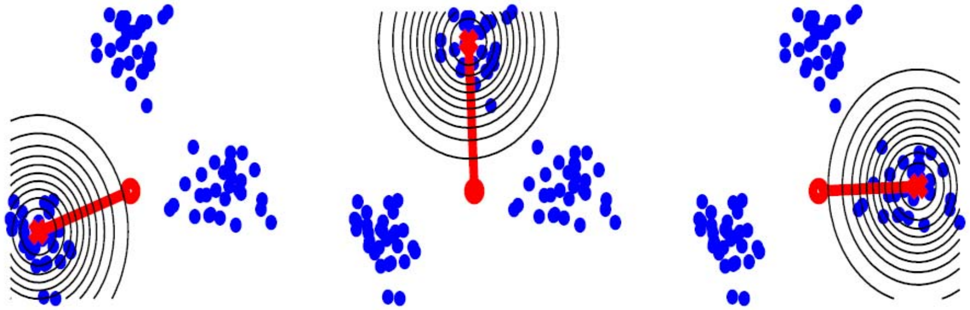
12/13/2006

Outline of Talk

- Background
  - Principal Component Analysis (PCA)
  - Gram-Schmidt
  - Support Vector Machine Kernels



- Algorithms
  - Approximate PCA
  - Approximate kernel PCA
- Examples
  - Parabola
  - Clusters
  - Taylor-Couette Flow
  - Microarray Data

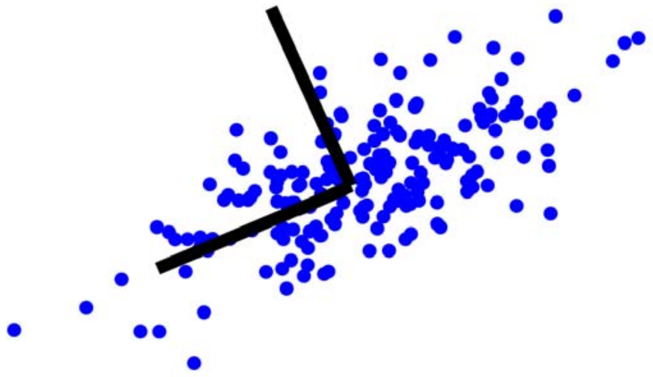


Principal Component Analysis (PCA)

PCA is a procedure for successively capturing the maximal variance in a dataset $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$:

$$\begin{aligned}
\mathbf{u}_1 &= \arg\max_{\mathbf{u}} \textstyle\sum_{i=1}^{n} (\mathbf{u}, \mathbf{x}_i)^2 \\
\mathbf{u}_2 &= \arg\max_{\mathbf{u}} \textstyle\sum_{i=1}^{n} (\mathbf{u}, \mathbf{x}_i - (\mathbf{x}_i, \mathbf{u}_1)\mathbf{u}_1)^2 \\
&\vdots \\
\mathbf{u}_m &= \arg\max_{\mathbf{u}} \textstyle\sum_{i=1}^{n} (\mathbf{u}, \mathbf{x}_i - \sum_{j=1}^{m-1} (\mathbf{x}_i, \mathbf{u}_j)\mathbf{u}_j)^2,
\end{aligned}$$

where $(\mathbf{x}, \mathbf{y})$ is the inner product between $\mathbf{x}$ and $\mathbf{y}$ and $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m\}$ are orthonormal.



Gram-Schmidt

Gram-Schmidt orthonormalization is a procedure for transforming linearly independent {$\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_m$} into an orthonormal basis {$\mathbf{u}_1$, $\mathbf{u}_2$, …, $\mathbf{u}_m$}:

$$\mathbf{u}_1 = \frac{\mathbf{x}_1}{\|\mathbf{x}_1\|},\ \mathbf{p}_1 = (\mathbf{x}_2, \mathbf{u}_1)\mathbf{u}_1$$

$$\mathbf{u}_2 = \frac{\mathbf{x}_2 - \mathbf{p}_1}{\|\mathbf{x}_2 - \mathbf{p}_1\|},\ \mathbf{p}_2 = (\mathbf{x}_3, \mathbf{u}_2)\mathbf{u}_2 + (\mathbf{x}_3, \mathbf{u}_1)\mathbf{u}_1$$

$$\vdots$$

$$\mathbf{u}_m = \frac{\mathbf{x}_m - \mathbf{p}_{m-1}}{\|\mathbf{x}_m - \mathbf{p}_{m-1}\|}.$$

Inner Product Gram-Schmidt

Gram-Schmidt can be re-written in terms of inner products:

$$\begin{aligned}
(\mathbf{x}_1, \mathbf{u}_1) &= \|\mathbf{x}_1\| = \tfrac{(\mathbf{x}_1, \mathbf{x}_1)}{(\mathbf{x}_1, \mathbf{u}_1)} \\
(\mathbf{x}_2, \mathbf{u}_1) &= \tfrac{(\mathbf{x}_2, \mathbf{x}_1)}{\|\mathbf{x}_1\|} = \tfrac{(\mathbf{x}_2, \mathbf{x}_1)}{(\mathbf{x}_1, \mathbf{u}_1)} \\
(\mathbf{x}_2, \mathbf{u}_2)^2 &= \|\mathbf{x}_2 - \mathbf{p}_1\|^2 = \|\mathbf{x}_2\|^2 - \|\mathbf{p}_1\|^2 \\
(\mathbf{x}_2, \mathbf{u}_2) &= \tfrac{1}{(\mathbf{x}_2, \mathbf{u}_2)} \left[(\mathbf{x}_2, \mathbf{x}_2) - (\mathbf{x}_2, \mathbf{u}_1)^2 \right] \\
(\mathbf{x}_3, \mathbf{u}_1) &= \tfrac{(\mathbf{x}_3, \mathbf{x}_1)}{(\mathbf{x}_1, \mathbf{u}_1)} \\
(\mathbf{x}_3, \mathbf{u}_2) &= \tfrac{1}{(\mathbf{x}_2, \mathbf{u}_2)} \left[(\mathbf{x}_3, \mathbf{x}_2) - (\mathbf{x}_2, \mathbf{u}_1)(\mathbf{x}_3, \mathbf{u}_1) \right]
\end{aligned}$$

$$\vdots$$

$$(\mathbf{x}_i, \mathbf{u}_j) = \frac{1}{(\mathbf{x}_j, \mathbf{u}_j)} \left[(\mathbf{x}_i, \mathbf{x}_j) - \sum_{k=1}^{j-1} (\mathbf{x}_j, \mathbf{u}_k)(\mathbf{x}_i, \mathbf{u}_k) \right].$$

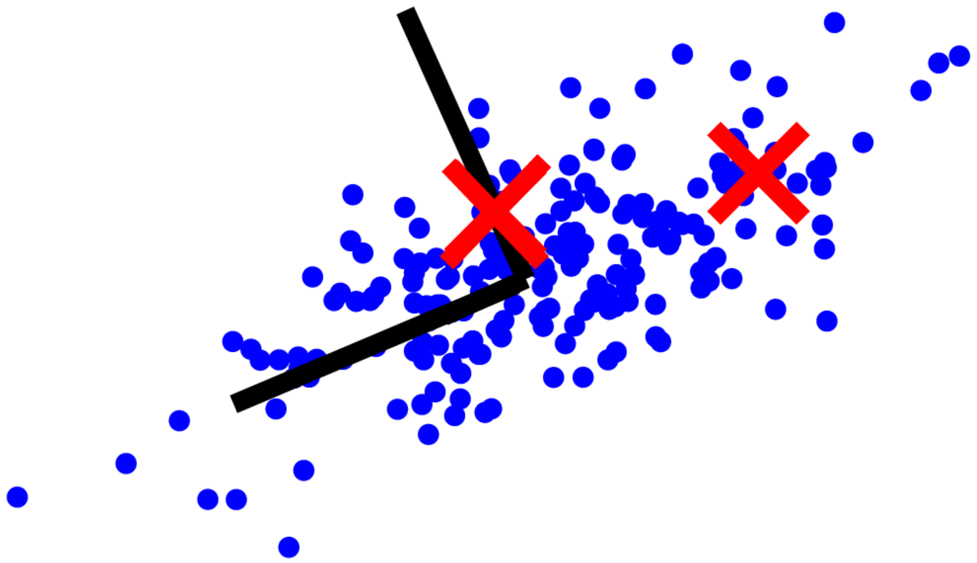
Approximate Version of PCA (APCA)

By combining PCA with the inner product version of Gram-Schmidt we obtain Approximate PCA (APCA):

$$\begin{aligned}
\mathbf{x}_{i_1} &= \arg\max_{\mathbf{x}_i} \tfrac{1}{\|\mathbf{x}_i\|^2} \textstyle\sum_{j=1}^{n} (\mathbf{x}_i, \mathbf{x}_j)^2 \\
\mathbf{x}_{i_2} &= \arg\max_{\mathbf{x}_i} \\
&\quad \tfrac{1}{\|\mathbf{x}_i\|^2} \textstyle\sum_{j=1}^{n} \left[(\mathbf{x}_i, \mathbf{x}_j) - (\mathbf{x}_i, \mathbf{u}_1)(\mathbf{x}_j, \mathbf{u}_1)\right]^2 \\
&\vdots \\
\mathbf{x}_{i_m} &= \arg\max_{\mathbf{x}_i} \\
&\quad \tfrac{1}{\|\mathbf{x}_i\|^2} \sum_{j=1}^{n} \left[(\mathbf{x}_i, \mathbf{x}_j) - \sum_{l=1}^{m-1} (\mathbf{x}_i, \mathbf{u}_l)(\mathbf{x}_j, \mathbf{u}_l)\right]^2 .
\end{aligned}$$

APCA

The primary advantage of APCA over PCA is that APCA is easier to interpret than PCA, due to the fact that APCA basis vectors are instances of the original dataset.



Support Vector Machine (SVM) Kernel Functions

A Support Vector Machine (SVM) kernel is a function $k : \circ^{d} \times \circ^{d} \rightarrow \circ$ with an associated map $\Phi : \circ^{d} \rightarrow F$ such that

$$k(\mathbf{x}, \mathbf{y}) = (\Phi(\mathbf{x}), \Phi(\mathbf{y})),$$

where F is an inner product space. Kernels act to introduce nonlinear interactions in the original space $\circ^{d}$ while maintaining linear relations in the higher dimensional space F.

Examples of kernel functions include:

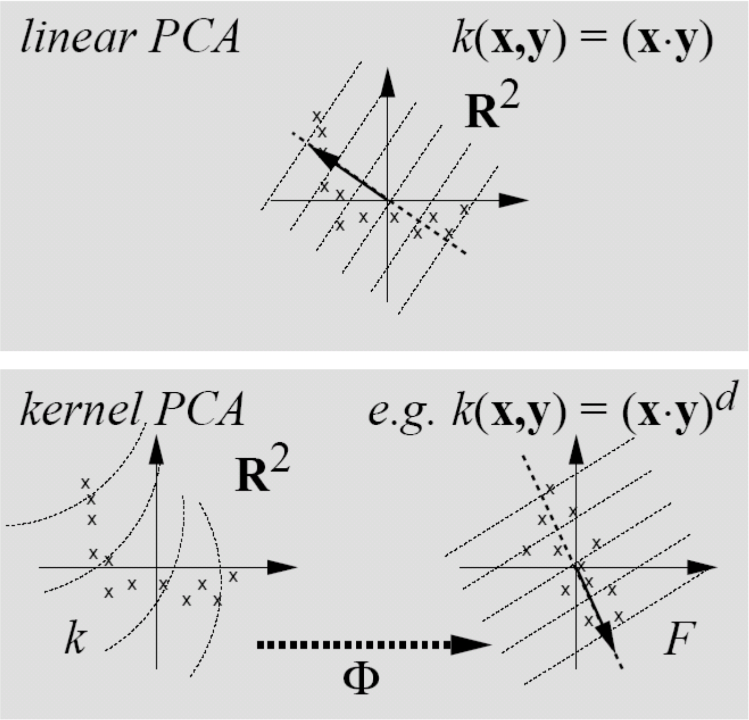
$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}, \mathbf{y}) \qquad \text{(linear)}$$

$$k(\mathbf{x}, \mathbf{y}) = ((\mathbf{x}, \mathbf{y}) + a)^{b}, b \in \square \qquad \text{(polynomial)}$$

$$k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\| / 2\sigma^{2}), \sigma \neq 0 \qquad \text{(Gaussian).}$$

Kernel PCA (kPCA)

Kernel PCA (kPCA) (Scholkopf, 1998) is obtained by using kernels with PCA:



linear PCA
$k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})$
$\mathbf{R}^2$
kernel PCA
e.g. $k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d$
$\mathbf{R}^2$
k
Φ
F

Kernel PCA can discover nonlinear relationships in a dataset.

Approximate Kernel PCA (AKPCA)

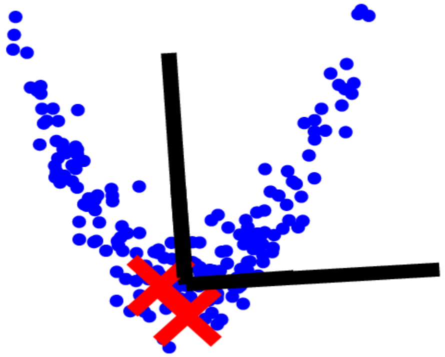
Kernel PCA can be difficult to interpret because basis vectors are members of the high dimensional space *F*.

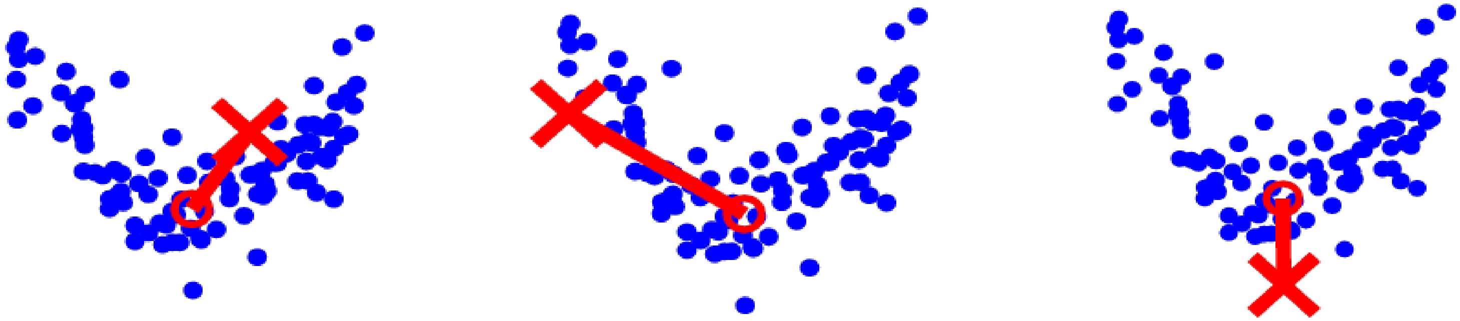
One possible solution is to use APCA with kernels (AKPCA). Then:

- AKPCA can discover nonlinear relationships in a dataset.
- AKPCA is interpretable because basis vectors are members of original dataset.
- AKPCA can be computed on larger datasets than kPCA because it is iterative and parallelizable.

AKPCA Applied to Parabola

Neither PCA or APCA can discover nonlinear structure in the parabola.

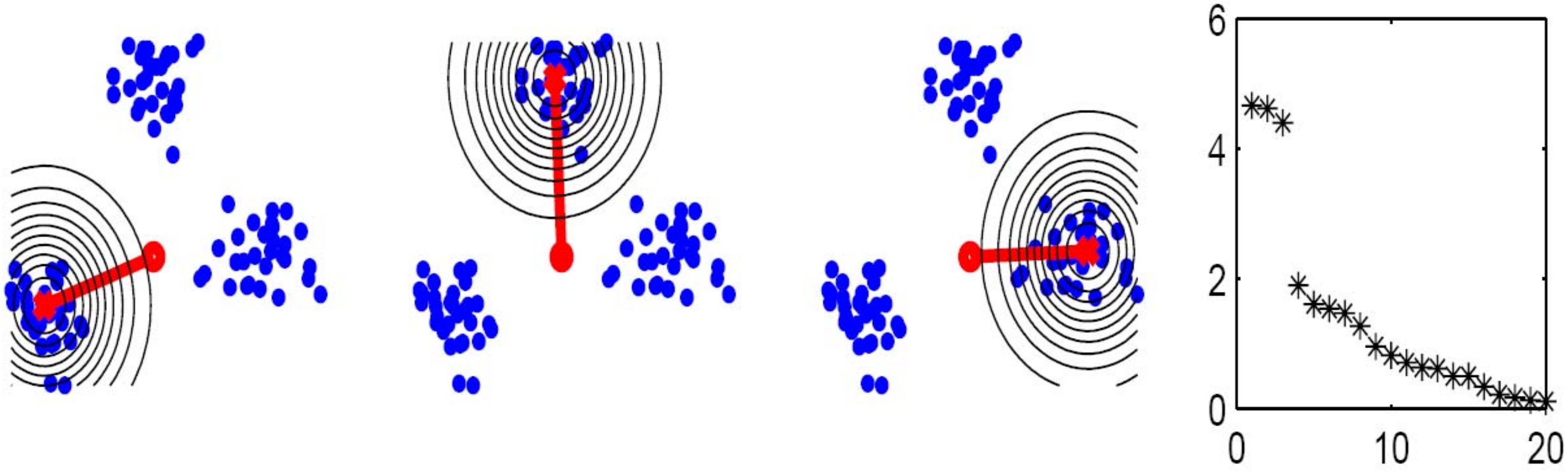




AKPCA discovers parabolic structure and provides explanation in terms of basis vectors. First two basis vectors are arms of parabola. Third basis vector is noise.

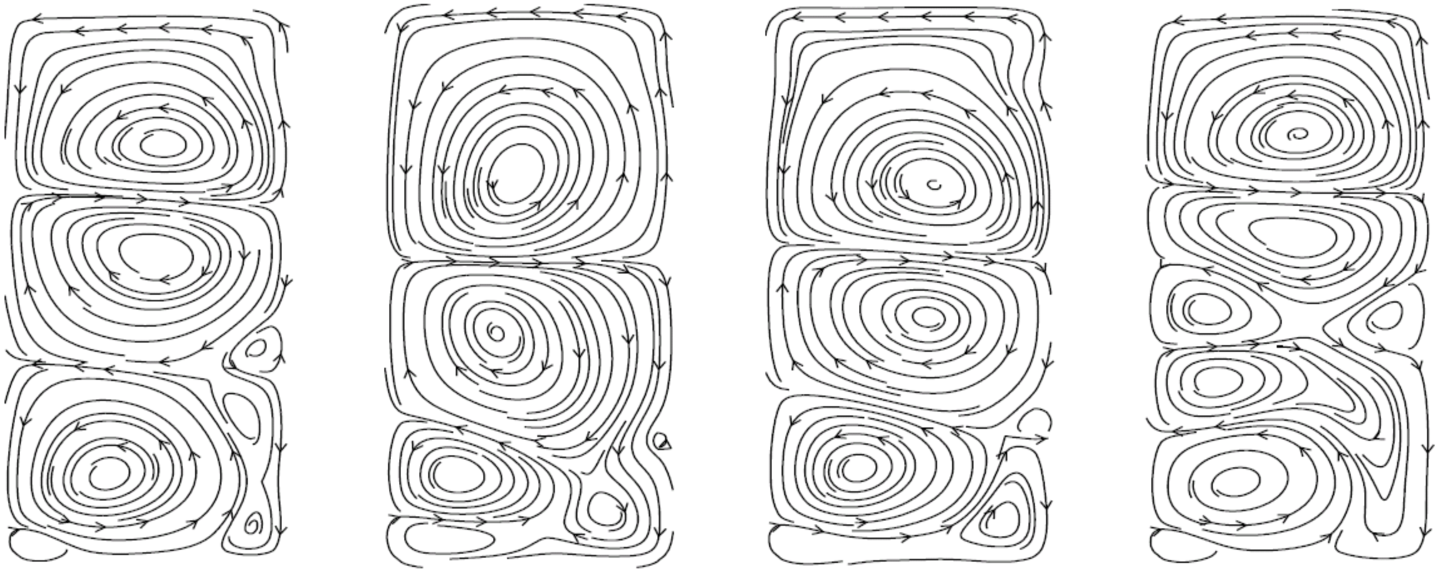
AKPCA Applied to Clusters

AKPCA basis vectors locate the cluster centers of three clusters. AKPCA “singular values” show these are the most important features of the dataset.



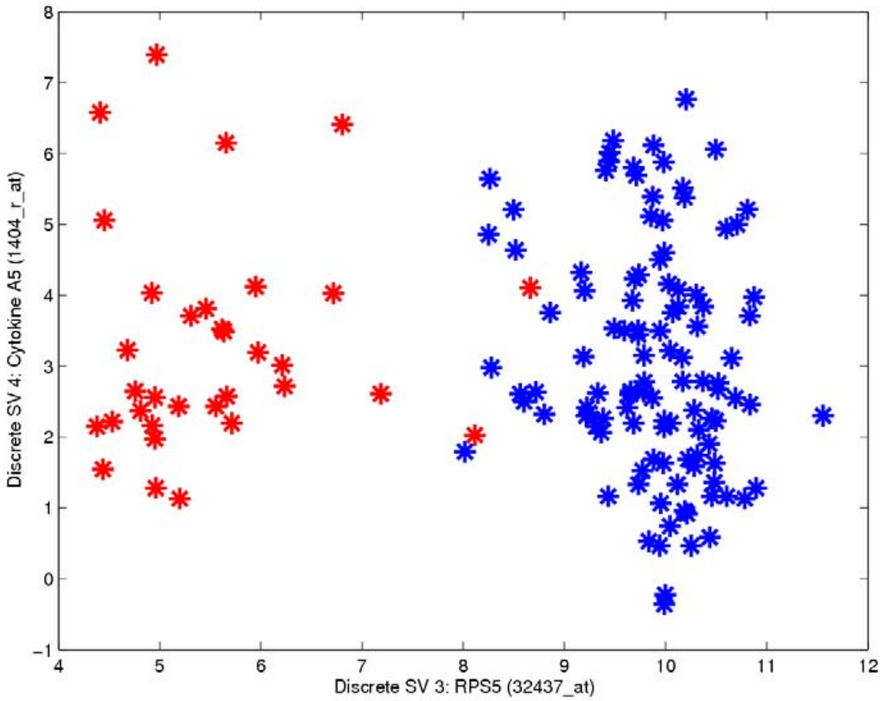
APCA Applied to Taylor-Couette Flow

Taylor-Couette flow occurs in fluid trapped between two concentric rotating cylinders. APCA was used to find the first four most typical flow patterns.



APCA Applied to Microarray Data

APCA was used with a Leukemia microarray dataset to identify a group of anomalous patients, shown here in red. APCA also identified the gene most relevant to distinguishing the cluster (*x*-axis, or RPS5).



Conclusions

- We developed an inner product (kernel) version of Gram-Schmidt.
- We applied this version of Gram-Schmidt to obtain an approximation to PCA (APCA) which is easier to interpret than standard PCA.
- We used kernels to provide an approximate version of kernel PCA (AKPCA).
- AKPCA provides nonlinear capabilities for data analysis and is also easier to interpret than kernel PCA.
- AKPCA can be used on very large datasets because it is iterative and can be made parallel.