SPEECH MANIFOLDS

A Arias

arias@irit.fr

1. INTRODUCTION

In many tasks we would like to discover the structure where high dimensional data «lives». For example, if we think in all possible color values that pixels in an image can take, we can imagine that only a subset of all possibilities build images with particular *sense*. When dealing with speech vectors, in spite of the huge quantity of different sounds that humans can produce, we notice that from all acoustic space, only certain sounds can effectively be produced by the human vocal tract. Hence we can say that there is an intrinsic dimensionality, or a small number of relevant directions, on the production space of some data. A manifold structure formalize this idea: it is a coordinate system embedded in a high dimensional space. A manifold is a mathematical space that looks locally like a Euclidean space of some fixed dimension.

2. PRINCIPAL COMPONENT ANALYSIS

If data lies in a high dimensional space, we might suppose it lies in a hyperplane. We can then approximate each data point by using the vectors that span the hyperplane. We are trying to choose a more appropriate low dimensional coordinate system that will approximately represent the data [Cal89].

$$x \approx y, \quad dim(y) \ll dim(x)$$

Optimal (in the sense of minimal squared reconstruction error) lower dimensional representation is giving by projecting the data onto the subspace spanned by the k eigenvectors (called principal axes E) of its covariance matrix with largest eigenvalues λ_k. The reconstruction error is dominated by these eigenvalues. If we look at the eigenvalue spectrum (figure 1), this would give an indication of the data intrinsic dimensionality. The directions corresponding to the small eigenvalues are sometimes interpreted as «noise», even if they can play a role in for example, a classification task. Each axe k of the new space has a variance λ_k.

$$y = E^T(x - \mu)$$

More generally, we would expect data to lie on low dimensional curved manifolds.

3. KERNEL MANIFOLDS

Kernel functions specify a nonlinear mapping from input space to feature space. We'll give here an insight to the manifolds where the data is mapped and how the Riemannian metric induced on the manifold is expressed in terms of the kernel.

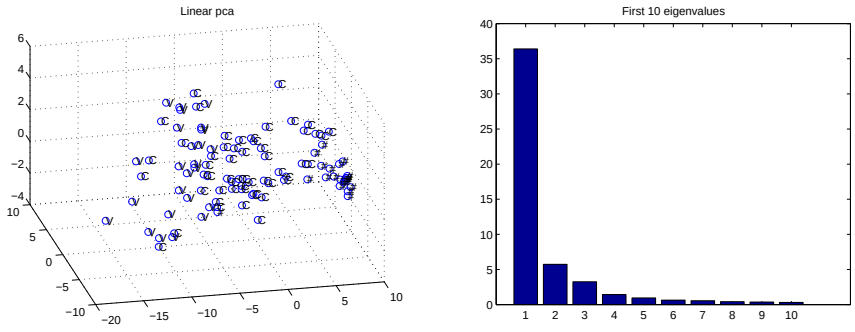

Fig. 1. PCA analysis of MFCC vectors from a sentence in english (reduction from 48 to 3 dimensions). Intrinsic dimensionality seems to be 5.

The use of kernel functions in support vector classification and regression is interpreted as a projection of original data from a d-dimensional space $\boldsymbol{L}$ to a «high dimensional» space $\boldsymbol{H}$. Suppose that your data are vectors in $\boldsymbol{R}^2$ and you choose $K(x_i \cdot x_j) = (x_i \cdot x_j)^2$. Then we find the mapping from $\boldsymbol{R}^2$ to $\boldsymbol{R}^3$ such that $(x_i \cdot x_j)^2 = \Phi(x_i) \cdot \Phi(x_j)$.

$$\Phi(x) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$$

Figure 2 shows how to think of this mapping: the image $\Phi(x)$ may live in a space of very high dimension, but is just a surface $\boldsymbol{S}$ whose intrinsic dimension is that of x. The mapping Φ and the space $\boldsymbol{H}$ are unique for a given kernel. $\boldsymbol{H}$ is a linear space, with an inner product defined and complete with respect to the corresponding norm (any Cauchy sequence of points converges to a point in the space). The properties of valid kernel functions are given by Mercer's conditions.

There are three measures of distance on $\boldsymbol{S}$. The first case considers the distance measured along the surface itself using the Riemmannian metric on $\boldsymbol{S}$. The second case is the Euclidean distance measured between the two points in $\boldsymbol{H}$. In this case the line joining two distance points will in general leave the surface $\boldsymbol{S}$. The third case considers projections from the surface along some hyperplane in $\boldsymbol{H}$, and it is the distance used in support vector classification.

A Riemannian metric defines a metric such that the distance between two points along a path lying in $\boldsymbol{S}$ and parameterized by $t \in [t_0, t_1]$ is given by a path integral.

For any positive kernel, the image $\boldsymbol{S}$ of the mapping will be a Riemannian manifold. All intrinsic geometrical properties of $\boldsymbol{S}$ can be derived once we know the Riemannian metric induced by the embedding of $\boldsymbol{S}$ in $\boldsymbol{H}$. The Riemannian metric can be defined by a symmetric metric tensor $g_{\mu\nu}$. The line element on $\boldsymbol{S}$ can be written (dx represent a small but finite displacement in $\boldsymbol{L}$):

$$ds^2 = \| \Phi(x + dx) - \Phi(x) \|^2$$

$$ds^2 = K(x + dx, x + dx) - 2K(x, x + dx) + K(x, x)$$

$$ds^2 = ((1/2)\partial_{x_\nu}\partial_{x_\mu}K(x, x) - \partial_{y_\nu}\partial_{y_\mu}K(x, y))_{y=x}dx^\mu dx^\nu$$

$$ds^2 = g_{\mu\nu}dx^\mu dx^\nu$$

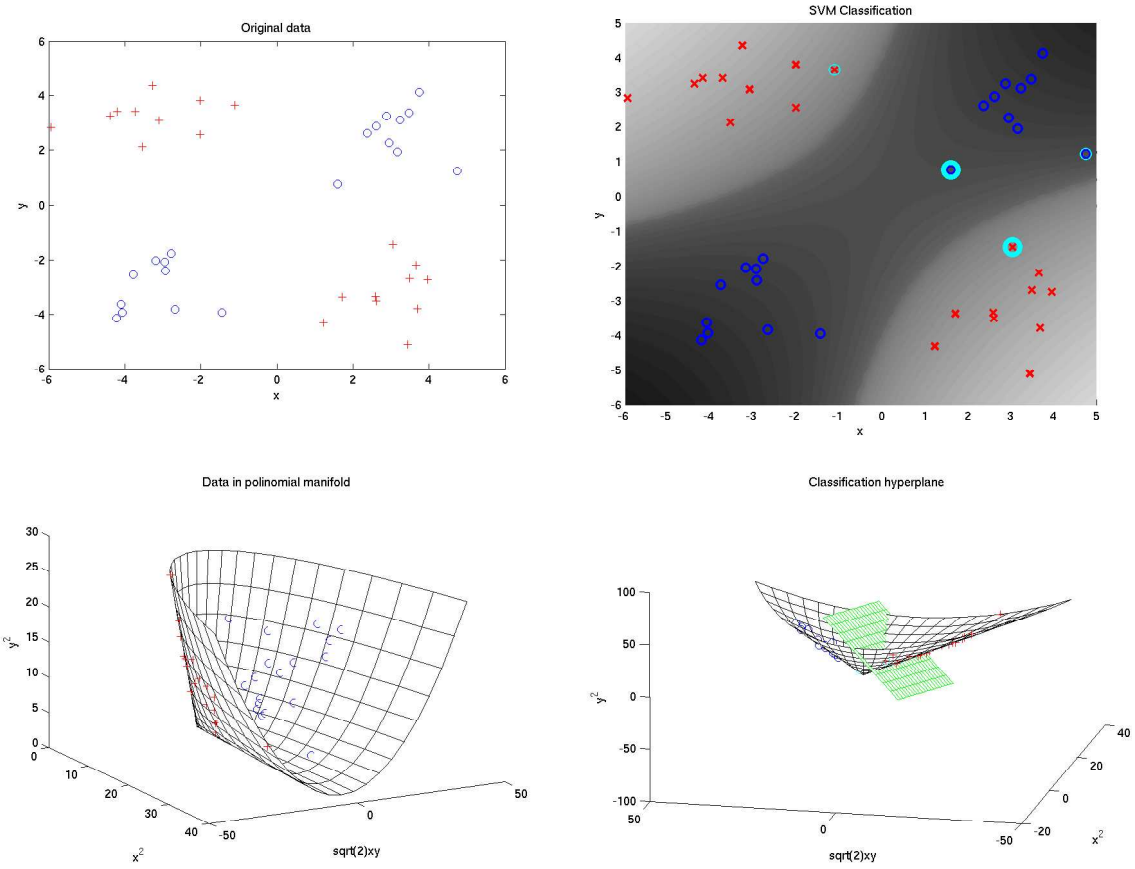

Fig. 2. 2D → 3D data projection for classification using a polynomial kernel function.

Example: Suppose Φ is the map from the line segment onto the circle of radius r: $\Phi : [0, 2\pi) \to \boldsymbol{S}^1$.

$$\Phi : \theta \to \begin{pmatrix} r\cos\theta \\ r\sin\theta \end{pmatrix}, \ r \text{ fixed}$$

Then

$$K(\theta, \theta') = r^2(\cos\theta\cos\theta' + \sin\theta\sin\theta') = r^2\cos(\theta - \theta')$$

and

$$ds^2 = \{(1/2)\partial_\theta^2 K(\theta,\theta) - \partial_{\theta'}\partial_{\theta'} K(\theta,\theta')\}_{\theta=\theta'} d\theta^2$$

$$ds^2 = r^2 d\theta^2$$

We do not need to know the explicit mapping Φ to construct $g_{\mu\nu}$; it can be written solely in terms of the kernel. For kernels that are functions only of dot products between points in $\boldsymbol{L}$ ($K(x, y) = K(x \cdot y)$), the general expression for the Riemmannian metric is [Bur98]:

$$g_{\mu\nu} = \delta_{\mu\nu} K'(\| x \|^2) + x_\mu x_\nu K''(\| x \|^2)$$

where the prime denotes derivative with respect to the argument $\| x \|^2$. For homogeneous polynomial kernels of type $K(x, y) = (x \cdot y)^p$ (for which Φ is a polynomial map), $g_{\mu\nu}$ is:

$$g_{\mu\nu} = \delta_{\mu\nu} p(\| x \|^2)^{p-1} + x_\mu x_\nu p(p-1)(\| x \|^2)^{p-2}$$

and the Riemann curvature:

$$\Gamma^p_{\mu\nu} = \frac{p-1}{\|x\|^2} \left(x_\mu \delta_{p\nu} + x_\nu \delta_{p\mu} - \frac{x_\mu x_\nu x_p}{\|x\|^2)} \right)$$

For $p = 2$ and $\dim(\boldsymbol{L}) = 2$ the surface generated is that of figure 2(c). For Gaussian Radial Basis Function kernels, $K = \exp^{-\|x-y\|^2/2\sigma^2}$, the metric tensor becomes $g_{\mu\nu} = \delta_{\mu\nu}/\sigma^2$. In this case the embedding space is infinite, and even if the surface $\boldsymbol{S}$ looks like a sphere since $\| \Phi(x) \|^2 = 1 \; \forall x \in \boldsymbol{L}$, and the manifold $\boldsymbol{S}$ is a $\dim(\boldsymbol{L})$ manifold, $\boldsymbol{S}$ is certainly not a $\dim(\boldsymbol{L})$-sphere.

3.1. Spherical normalisation of polynomial kernels

When using polynomial kernels, some values increase significantly, especially if the degree becomes large. A solution to alleviate this is to normalise the lengths of feature vectors with its Euclidean norm and constraint the dot products to the range $-1 \leq \overline{x} \cdot \overline{y} \leq +1$. However, this results in information loss and certain classification ambiguity. One alternative normalisation, with less information loss, can be achieved adding one dimension to the original space.

Consider the mapping from a 2D plane to a 3D unit sphere illustrated in figure 3. The new vectors representing the data are the unit vectors from the centre of the sphere to its surface. Mapping a plane onto a sphere's surface may be achieved by many different projections and can be generalised to arbitrary dimensions. The figure 3 shows an azimuthal projection, the modified stereographic projection [Wan03].

The mapping is achieved by augmenting the vectors with a constant d, and normalising the new vector by its Euclidean length (figure 4). Substituting the length normalised vectors into the polynomial kernel expression we get the spherically normalised polynomial kernel. The constant d is the shortest distance from the origin of the hemisphere to the input hyperplane. It determines how the data should be distributed on the sphere.

$$x' = \frac{1}{\sqrt{x^2+d^2}} \begin{bmatrix} x \\ d \end{bmatrix}$$

$$K_{sphnorm}(x', y') = \frac{1}{2^n} \left(\frac{x \cdot y + d^2}{\sqrt{(x \cdot x + d^2)(y \cdot y + d^2)}} + 1 \right)^n$$

The extra dimension may be added in input space or, if we are using an explicit polynomial expansion, in feature space.

4. KERNEL PCA

We saw that linear PCA attempts to provide us with a set of orthogonal axes along which we can project our data, capturing most of data's variance with just the first few axes of the new space. Kernel PCA relies on the utilisation of kernel functions to perform PCA in $\boldsymbol{H}$. We should remember two things [SS02]:

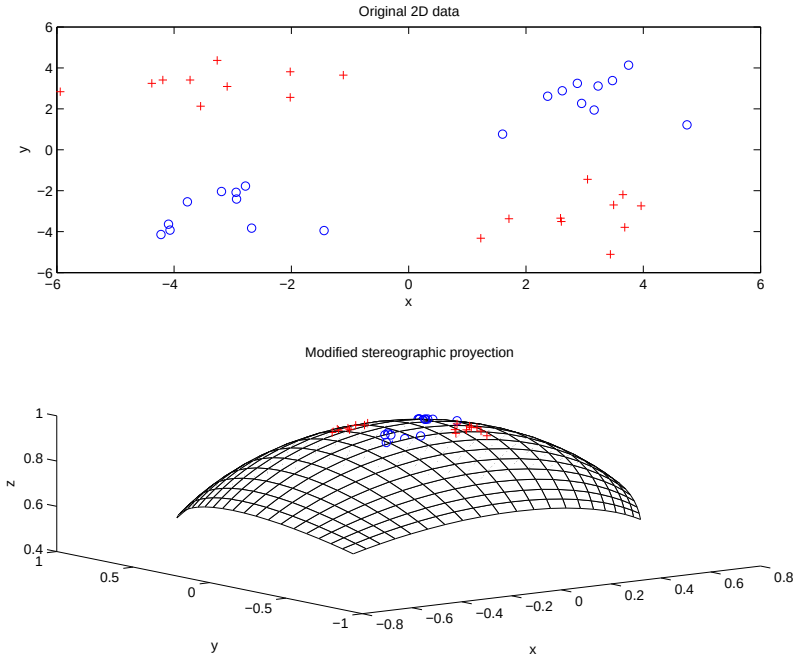

Fig. 3. 2D → 3D modified stereographic data projection.

1. VC theory shows that under some conditions, mappings Φ to high dimensional spaces provides great classification power (specially when the number of dimensions in the expansion is larger than the number of training vectors).

2. A kernel function $k(x_i, x_j)$ such that $x_i, x_j \in \boldsymbol{R}^d$, $k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ avoid us to compute this high dimensional mapping explicitly and is applicable when an algorithm depends only on dot products.

The covariance matrix of the mapped data in feature space is:

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

Where $\mu = \frac{1}{m} \sum_i \Phi(x_i)$. We are looking for the eigenvector solutions v.

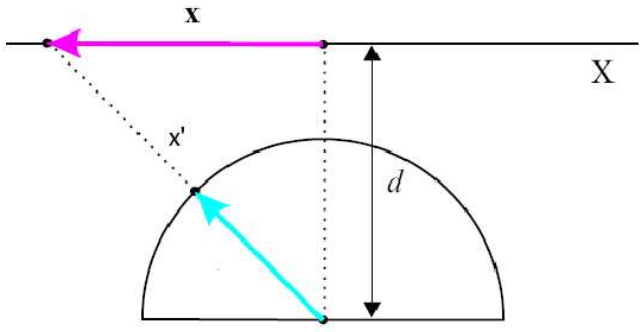

Fig. 4. Stereographic data projection.

$$Cv = \lambda v$$

There is a close relationship between the covariance matrix of «centered» data C and the kernel matrix K of X [STC04]. X is a $m \times N$ matrix of vectors in feature space (m =number of vectors, N =dimensions in feature space). If u is an eigenvector of mC and v is an eigenvector of K:

$$u = \lambda^{-1/2} X' v \text{ and } v = \lambda^{-1/2} X u. \ \lambda \text{ is an eigenvalue of } K \text{ and } mC.$$

We have then a representation for the *jth* eigenvector u_j of mC with the coefficients given by the corresponding eigenvector v_j of K scaled by $\lambda_j^{-1/2}$,

$$u_j = \lambda^{-1/2} \sum_i^m (v_j)_i \Phi(x_i) = \sum_i^m \alpha_i^j \Phi(x_i)$$

$\alpha^j = \lambda_j^{-1/2} v_j$. If we wish to compute the projection of a new data point $\Phi(x)$ onto the direction u_j,

$$P_{u_j} = u_j' \Phi(x) = \left\langle \sum_i^m \alpha_i^j \Phi(x_i), \Phi(x) \right\rangle = \sum_i^m \alpha_i^j k(x_i, x)$$

Pseudo-code of the Kernel PCA algorithm:

$$\begin{array}{l} \text{Calculate } K_{ij} = k(x_i, x_j),\ i, j = 1 \dots m \\ \text{Center } K_{ij} \\ [V, \Lambda] = eig(K) \\ \alpha^j = \frac{1}{\sqrt{\lambda_j}} v_j,\ j = 1 \dots k \\ \widetilde{x_i} = \left(\sum_{i=1}^m \alpha_i^j k(x_i, x) \right)_{j=1}^k \end{array}$$

Kernel PCA has the computational limitation of having to compute eigenvectors for matrices of size $m \times m$, but this can be addressed using th Nyström method for approximating the eigenvectors of a large Gram matrix (see appendix A).

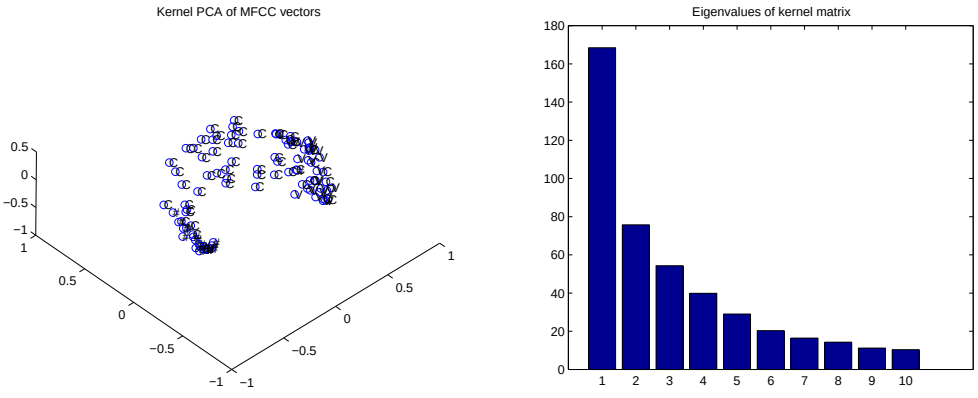

Fig. 5. Kernel PCA analysis of MFCC vectors from a sentence in english (non-linear reduction from 48 to 3 dimensions). Labelling of points shows the phonetic structure of the projection, with silences in one side in opposition to vowels. Consonant units are placed in the middle.

5. SPECTRAL CLUSTERING

Spectral methods for clustering have emerged recently. Their results are close to human experience because they create clusters that do not form convex regions in the space. They use eigenvectors of a matrix derived from the distance between points to help determine the partitions. Several algorithms have been proposed using eigenvectors in different ways. In [NJW01], a spectral relaxation is proposed to solve the k-way graph normalized cut problem. An affinity matrix A (nonnegative and symmetric) is built to represent the distance between points. If D is the diagonal matrix whose (i,i) entry is the sum of the entries of row i in matrix A, the top k eigenvectors of the Laplacian,

$$L = D^{-1/2} A D^{-1/2}$$

are clustered via K-means algorithm to compute a discrete partitioning of the points. These eigenvectors can be interpreted as a reduced dimension representation of the original samples.

The informal explanation of the algorithm is to consider the «ideal» case where 1) the vectors are ordered according to which cluster they are in and 2) the clusters are infinitely far apart from each other. In this case $\tilde{A}$ is block diagonal and each block A^{ii} represents the intra-cluster affinities for cluster i. For three clusters,

$$\tilde{A} = \begin{bmatrix} A^{11} & 0 & 0 \\ 0 & A^{22} & 0 \\ 0 & 0 & A^{33} \end{bmatrix} \quad ; \quad \tilde{L} = \begin{bmatrix} \tilde{L}^{11} & 0 & 0 \\ 0 & \tilde{L}^{22} & 0 \\ 0 & 0 & \tilde{L}^{33} \end{bmatrix}$$

Since $\tilde{L}$ is block diagonal, its eigenvalues and eigenvectors are the union of the eigenvalues and eigenvectors of its blocks padded appropriately with zeros. If we take the principal normalized eigenvector of block $\tilde{L}^{ii}$, we'll have the $n \times 3$ matrix Y.

$$\tilde{Y} = \begin{bmatrix} \tilde{Y}^1 \\ \tilde{Y}^2 \\ \tilde{Y}^3 \end{bmatrix} = \begin{bmatrix} v_1^{(1)} & 0 & 0 \\ 0 & v_1^{(2)} & 0 \\ 0 & 0 & v_1^{(3)} \end{bmatrix}$$

Each row Y_j will correspond to the true clustering of original data. In the general case there are not infinite distances between clusters, but we still hope to recover a stable cluster configuration. Matrix perturbation theory states that the choice of the first eigenvectors from the example will be a good approach if the difference between 3rd and 4th eigenvalue of $\tilde{L}$ is large. The difference between two eigenvalues is called the *eigengap*.

According to graph theory, we can consider the cloud of vectors as a graph $G = (V, E, A)$ where V is the set of vertices, E is the set of edges connecting vertices, and A is an edge affinity matrix, non negative and symmetric. Suppose two clusters $a \cup b = V$,

$$links(a,b) = \textstyle\sum_{i \in a, j \in b} A(i,j) \quad \text{is the sum of weights between } a \text{ and } b$$

if we normalize with respect to the «total weight» of each cluster,

$$normlinkratio(a,b) = \tfrac{links(a,b)}{links(a), links(b)}$$

We wish to find a and b such that $normlinkratio(a,b)$ is minimized. For a k-way partitioning of the vertices, the problem to solve is,

$$\min \tfrac{1}{k} \textstyle\sum_{j=1}^{k} normlinkratio(V_j, V \backslash V_j)$$

which is a NP-hard problem, solved approximately by the Laplacian presented before.

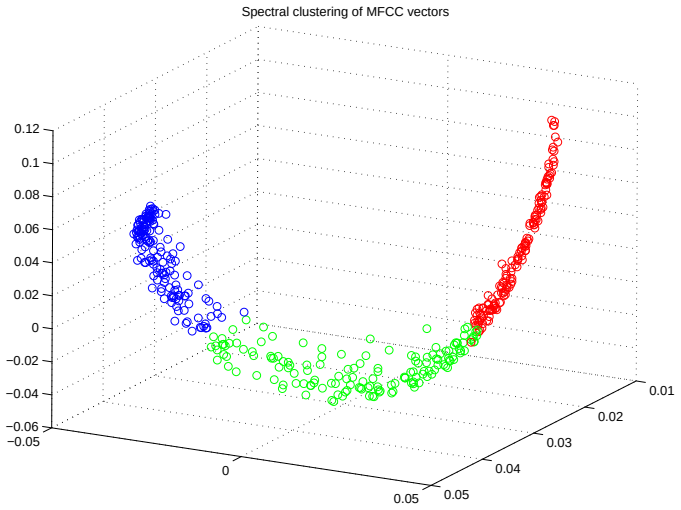

Fig. 6. Spectral clustering of MFCC vectors from a sentence in english (non-linear reduction from 48 to 3 dimensions). Color of points shows the phonetic structure of the projection (silences in one side in opposition to vowels). Consonant units are placed in the middle.

6. REFERENCES

[Bur98] C. Burges. Geometry and invariance in kernel based methods. In *Advances in Kernel Methods*. MIT Press, 1998.

[Cal89] Calliope. *La parole et son traitement automatique*. Masson, Paris, France, 1989.

[FBCM04] C Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nyström method. In *IEEE Transactions on pattern analysis and machine intelligence*, volume 26, February 2004.

[NJW01] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, 2001.

[SS02] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[STC04] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, UK, 2004.

[Wan03] V. Wan. *Speaker Verification using Support Vector Machines*. PhD thesis, University of Sheffield, 2003.

Appendix A. The Nyström Method

If $K \in M_n$ and the rank of K is $r \ll n$. Nyström method gives a way of approximating the eigenvectors and eigenvalues of K using those of a small submatrix A. If A has rank r, then the decomposition is exact.

The Nyström method is a technique for finding numerical approximations to eigenfunction problems of the form:

$$\int_a^b k(x,y)u(y)dy = \lambda u(x)$$

The integral is approximated using the quadrature rule:

$$\frac{1}{m}\sum_{i=1}^{m} k(x,x_i)u(x_i) \approx \lambda u(x)$$

A is a $r \times r$ matrix of affinities between r sample points with diagonalization $A = U\Lambda U'$. Let B represent the $r \times m$ matrix affinity between the x_r sample points and x_m remaining points. The matrix form of the Nyström extension for $u(x)$ is then $B^T U \Lambda^{-1}$, wherein B^T corresponds to the interpolation weights $k(x, x_i)$, U to $u(x_i)$ and Λ^{-1} to $1/\lambda_i$.

To understand the nature of Nyström extensions, we can see it as a matrix completion. Suppose that K_{mm} has rank $r < m$. If $n = m - r$, K can be written:

$$K_{mm} = \begin{bmatrix} A_{rr} & B_{rn} \\ B_{nr}^T & C_{nn} \end{bmatrix}$$

B contains the weights from the original samples to the rest, and C contains the weights between all remaining points. Since A is full rank, the r rows $[A_{rr}\ B_{rn}]$ are linearly independent, and since K is of rank r, the n rows $[B_{nr}^T\ C_{nn}]$ can be expanded in terms of them.

To approximate eigenvectors of K_{mm}, Nyström extension gives:

$$V = \begin{bmatrix} U \\ B^T U \Lambda^{-1} \end{bmatrix}$$

K can be written in the form:

$$K = V\Lambda V^T = \begin{bmatrix} U \\ B^T U \Lambda^{-1} \end{bmatrix} \Lambda \begin{bmatrix} U\ \Lambda^{-1} U^T B \end{bmatrix}$$

$$K = \begin{bmatrix} U\Lambda U^T & U\Lambda\Lambda^{-1}U^T B \\ B^T U U^T & B^T U \Lambda^{-1} U^T B \end{bmatrix} = \begin{bmatrix} A & B \\ B^T & B^T A^{-1} B \end{bmatrix}$$

We see that Nyström approximates C using $B^T A^{-1} B$. The quality of the approximation can then be qualified as the norm of the Schur complement $\| C - B^T A^{-1} B \|$. One detail is that the columns of V should be orthogonal.

If A is positive definite, V becomes:

$$V = \begin{bmatrix} A \\ B^T \end{bmatrix} A^{-1/2} U_S \Lambda_S^{-1/2}$$

So instead of diagonalize A we diagonalize S [FBCM04],

$$S = A + A^{-1/2} B B^T A^{-1/2}$$

$$S = U_S \Lambda_S U_S^T$$

So that $K = V\Lambda V^T$ and $VV^T = I_m$.

Example MATLAB code for finding the first k eigenvectors using Nyström approximation with positive definite affinities. The input is the matrix A_{rr} and B_{rn}.

```
% normalisation for NCut
  d1 = sum([A;B'],1);
  d2 = sum(B,1) + sum(B',1)*pinv(A)*B;
  dhat = sqrt(1./[d1 d2])';
  A = A.*(dhat(1:r)*dhat(1:r)');
  B = B.*(dhat(1:r)*dhat(r+(1:n))');
% Pseudoinverse of A preventing redundancy (linear dependence) in the random samples
  Asi = sqrtm(pinv(A));
% Matrix to diagonalize
  S = A + Asi*B*B'*Asi;
  [Us,Ls,T] = svd(S);
% Eigenvector approximation
  V = [A;B']*Asi*Us*pinv(sqrt(Ls));
% First k eigenvectors in E
  for i = 2:k+1,
  E(:,i-1) = V(:,i)./V(:,1);
  end
```

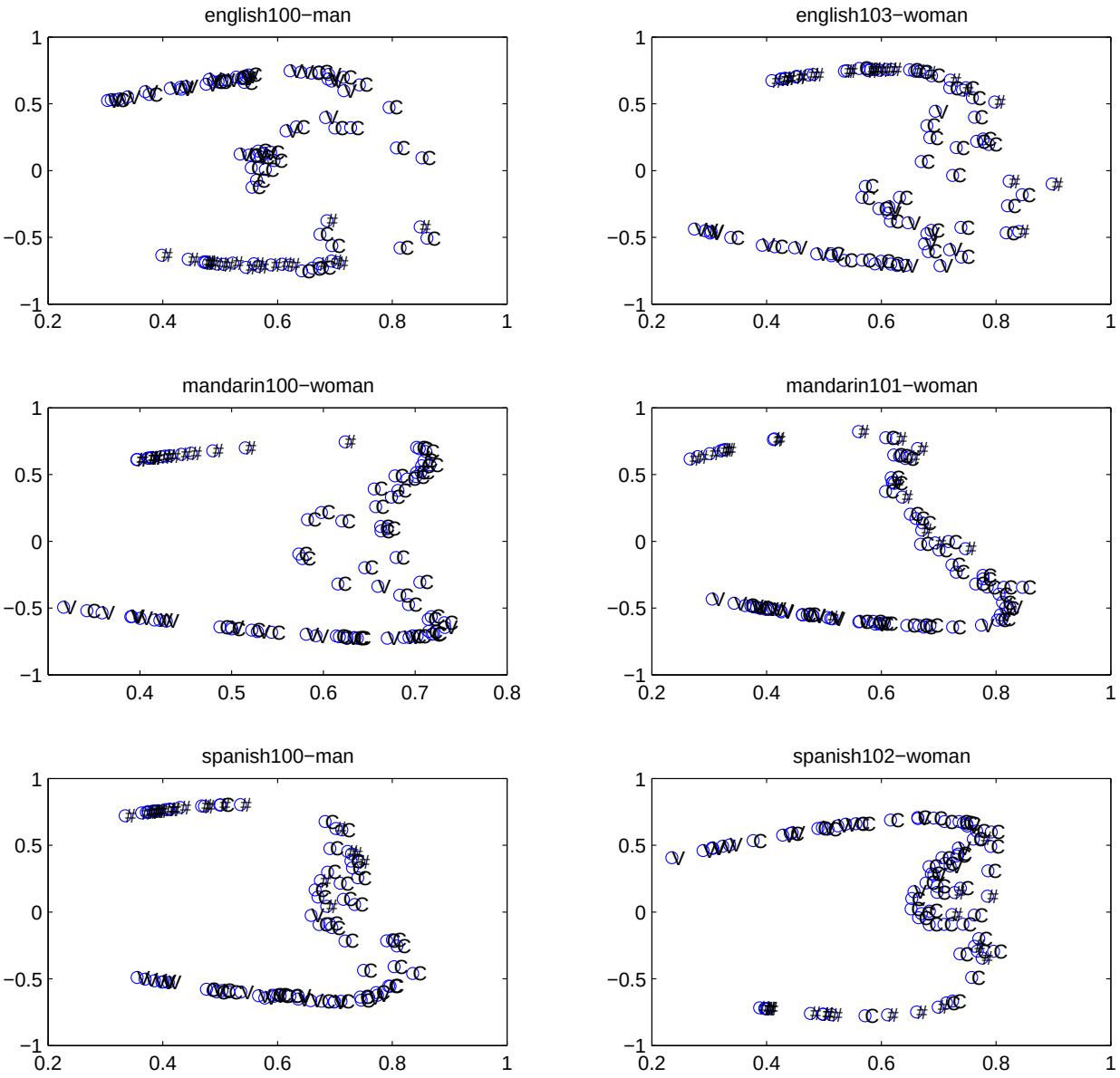

Fig. 7. Spectral clustering of MFCC vectors from sentences in different languages.