Dimension Reduction on Hyperspectral Images *

Meiching Fong
August 31, 2007
UCLA Department of Mathematics
Faculty Advisor: Prof. Todd Wittman

Introduction:

Although hyperspectral images contain much more information than regular RGB images, more than ninety percent of the variance can be explained by a small portion of the data. The goal of dimension reduction is to map high dimensional data into a lower dimension while preserving the main features of the original data. The dimension reduction codes are taken from Van Der Maaten's matlab dimension reduction toolbox [1], except for FastICA which comes from [2]. The Urban and Indian Pines datasets we are working with can be obtained from the *HyperCube* [3] and *MultiSpec* [4] website, respectively.

Dimension Reduction Methods:

We consider the following methods:

- PCA
- FastICA
- Laplacian Eigenmaps
- LLE
- LTSA
- LLTSA
- Diffusion Maps

Besides the traditional linear techniques of PCA and FastICA, we have tried Diffusion Maps, which is a global nonlinear technique that tries to preserve all pair wise distances. Laplacian Eigenmaps, LLE (Local Linear Embedding) and LTSA (Local Tangent Space Analysis) are local nonlinear techniques that preserve the properties of small neighborhoods around the data. LLTSA uses a linear technique to minimize the cost function in LTSA. One major drawback for most of the methods above is their incapability to handle images larger than 70 x 70. Thus, we divide the original hyperspectral images to about 50 x 50 square images then take the average of the results. There are other dimension reduction algorithms suggested by the toolbox, but they are unsuitable for hyperspectral images. LDA, GDA, LPP, LCC require user input more parameters. SPE, SNE, KernelPCA, CFA take more than ten minutes to run on a 50 x 50 image. The neighborhood of hyperspectral images fail to be connected to run FastMVU. HLLE destroys the patterns of hyperspectral images and the Isomap algorithm provided reshapes the original matrix, so we are unable to compare the result.

Performance Measurement:

To measure the performance of different dimension reduction algorithms, we consider four metrics. The classification rate, which is obtained by comparing the clustering of Kmeans on the dimension reduced data with the ground truth (Figure 1). The mean improvement is obtained by subtracting the classification rate of the original data from the classification rate of the dimension-reduced data. We also look at the standard deviation of the classification rate and the runtime.

* This project is supported by NGA NURI grant HM1582-06-BAA-004

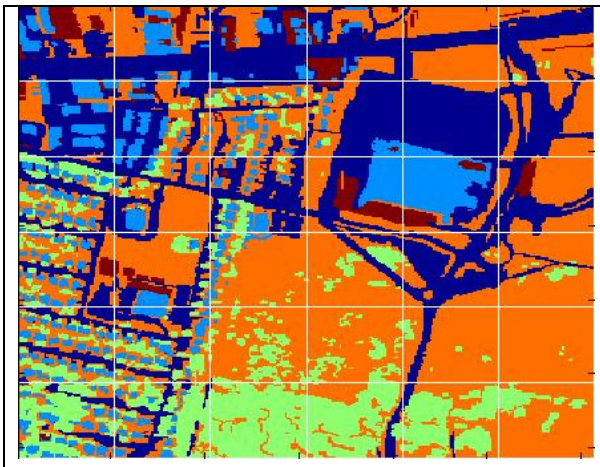
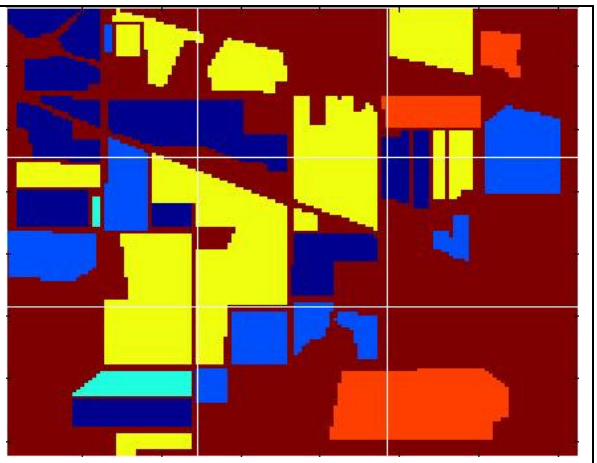


Fig.1a. Hand labeled ground truth of Urban



1b. Compressed ground truth of Indian Pines 220 bands, originally from Multispec©.

Intrinsic Dimension:

The original noisy Urban has 210 bands and the clean Urban has 162 bands. The original noisy Indian Pines has 220 bands and the clean Indian Pines has 179 bands. A natural question to ask is how many dimensions we should reduce to. We fed in the data into the intrinsic dimensionality estimation techniques suggested by Van Der Maaten's toolbox. As seen from table 1, they do not really agree, but we do get the majority of them suggesting below 10 dimensions. Because of the reflection of water vapor in the atmosphere, about 20 % of the bands are corrupted. Even when reducing the original noisy data to just six dimensions using PCA, two out of the six bands contain just pure noise. To get a better idea of the intrinsic dimension, we reduced the noisy datasets to various dimensions by PCA and look at the classification rate by comparing the result of Kmeans. Looking at the result of Figure 2, besides the outlier at dimension 2, the classification rate flattens out after six dimensions. Moreover, compare column A and column B from table 2 to 5, the mean improvement rates are larger for four dimensions.

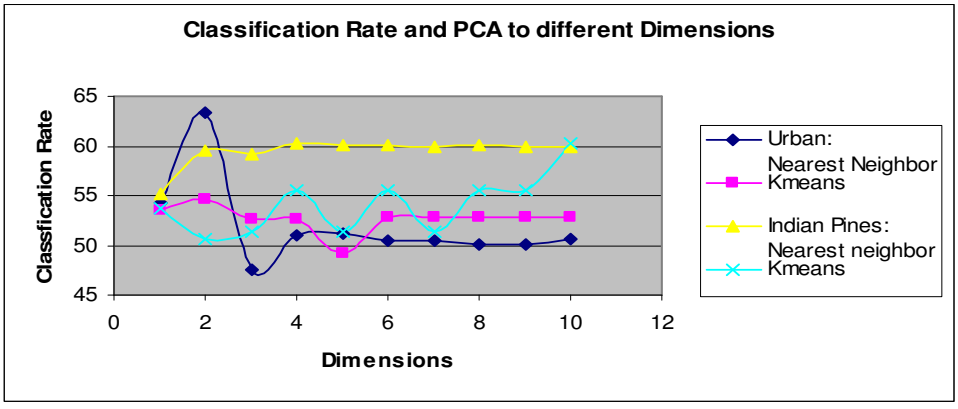


Fig. 2. Classification rate of different dimensions using PCA.

The Experiments and Results:

According to table 2 to 5, the clean data gives better classification rate than noisy data. In the experiments below, we only set the number of dimensions desired. Other parameters are the default values in the toolbox. We use k=12 for nearest neighbors for the nonlocal methods and $\sigma = 1.0$ for Diffusion maps. Without doing any variation on the data with Euclidean distance, Diffusion map gives the highest classification rate with a relatively fast runtime (Table 2-Table 5). However, it also has higher standard deviation, which implies it may not be a very stable method. LLE and LTSA are more stable with lower standard deviation (Figure 3 and 4).

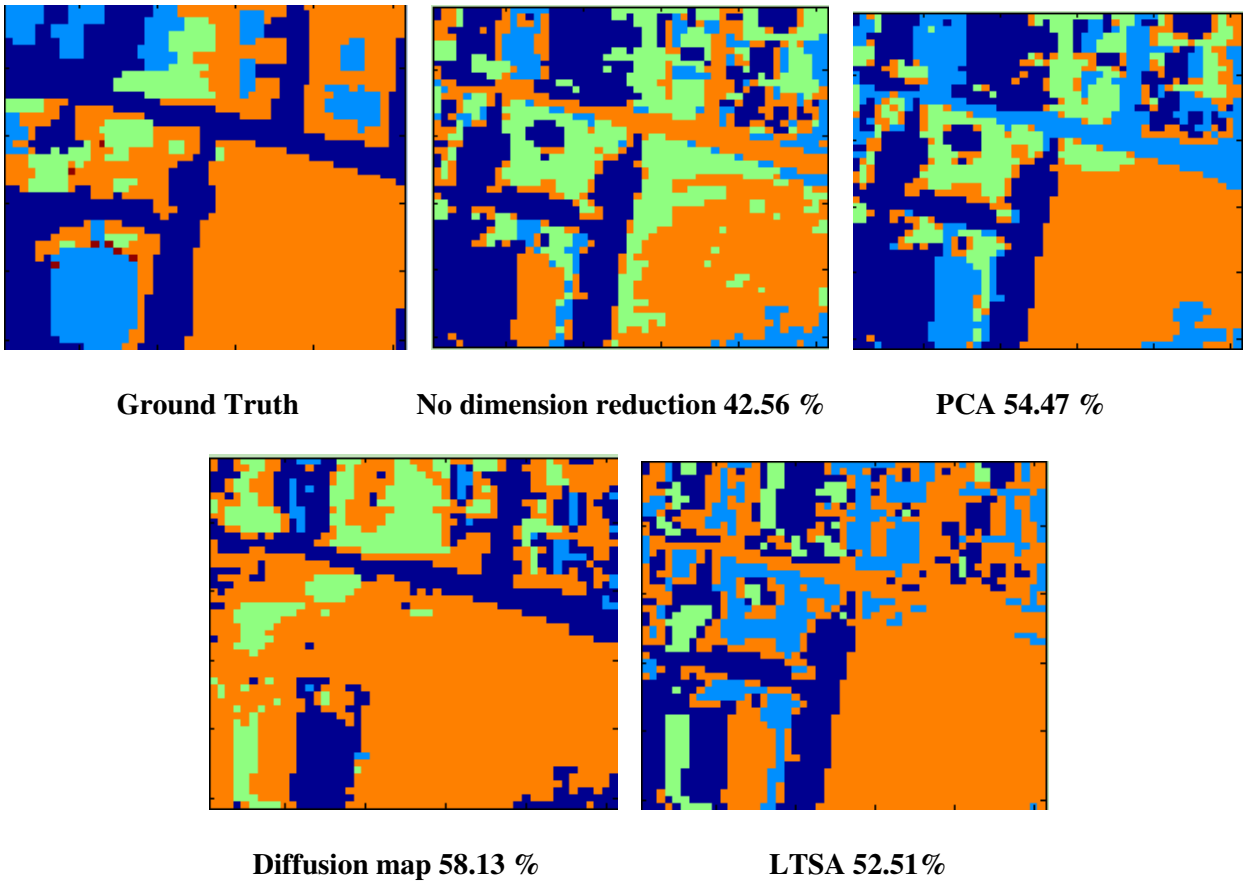
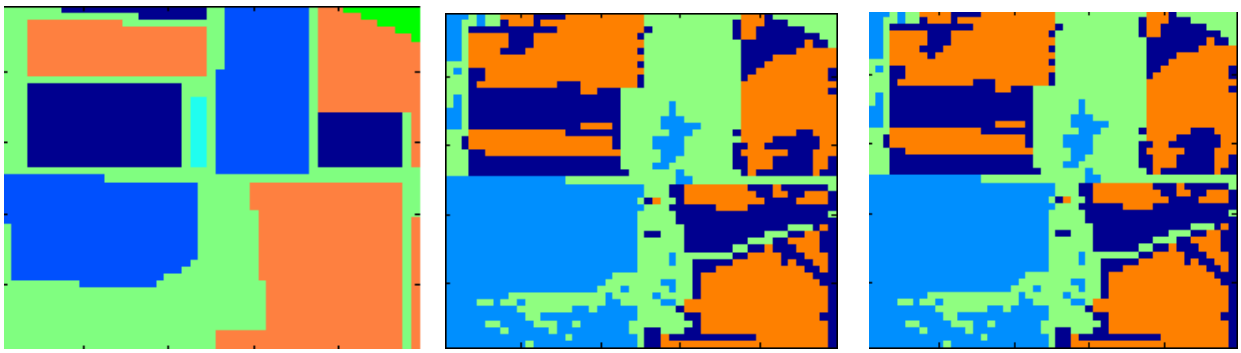
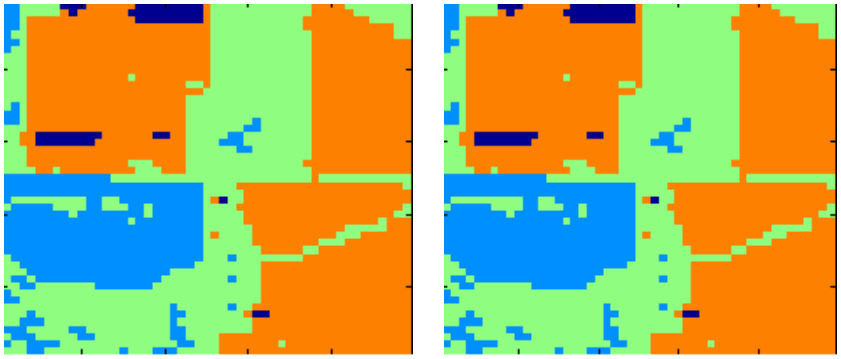


Fig. 3. Classification rate of a portion of clean Urban dataset with PCA, Diffusion map and LTSA



Ground Truth **No dimension reduction 62.23 %** **PCA 62.29 %**



Diffusion map 64.96 % **LTSA 69.89 %**

Fig. 4. Classification rate of a portion of clean Indian Pines dataset with PCA, Diffusion map and LTSA

Although Euclidean is the most popular built-in distance function in many toolbox, with a few steps of derivation, it can be shown easily that the cosine distance is similar to taking the Euclidean distance of (**x**/ |**x**| , **y**/|**y**|). To get the correlation distance, we may take the Euclidean distance of $((\mathbf{x}-\mu_x)/\sigma_x, (\mathbf{y}-\mu_y)/\sigma_y)$. Cosine or Correlation distance generally improves the classification rates (Table 5 and Table 6).Even though mean improvement rate seem to go down, it's because we are getting much higher classification rate for the original data as well. It is clear that PCA is the fastest method. To cut down the runtime of the routine, one may first perform PCA to reduce to half or a quarter of the original dimensions, then use different methods to further reduce the dimensions to less than ten bands. The result is presented at Table 7, comparing with direct dimension reduction (Table 5, Column B), we do not lose much information. The wavelet transform of the data doesn't seem to improve the classification rate (Table 8). The last experiment is on spatial coherence as suggested by Mohan, Sapiro and Bosch's paper [5]. Instead of considering a single pixel, we consider its neighborhood pixels within an n x n window as illustrate in Fig.5. It's hard to tell the effect of spatial coherence. It

seems to improve the classification of some methods but not all. Diffusion map gives the highest classification rate for Euclidean distance on original data using sigma range within 1.25 to 2.

Note that Kmeans clustering randomizes the initial points. It can produce different classification rates for every run. A more precise way of classifying the data and the ground truth is worth further investigation.

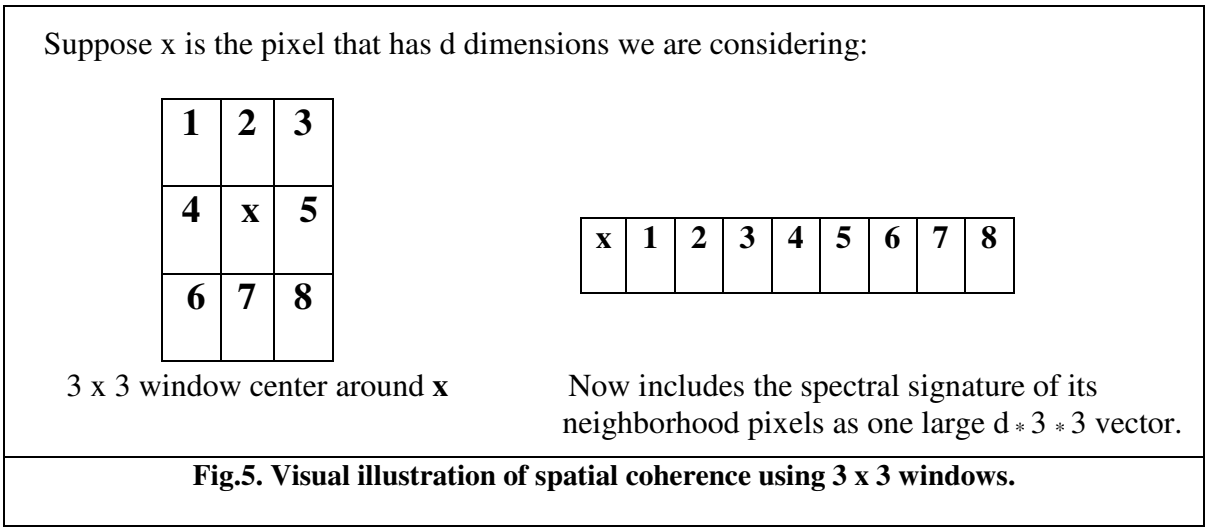


Fig.5. Visual illustration of spatial coherence using 3 x 3 windows.

Table 1: Intrinsic Dimension

Method	Noisy Urban	Clean Urban	Noisy Indian Pines	Clean Indian Pines
Eig Value	4	3	2	2
Packing Numbers	2.892	1.925	5.094	11.363
MLE	13.261	9.374	20.980	12.284
GMST	19.545	19.568	133.048	4.315
NearNbDim	4.981	4.974	4.319	3.627
CorrDim	3.981	2.373	8.583	2.032

Table 2: Noisy Urban 210 bands

	Column A				**Column B**			
	6 Dimensions				4 Dimensions			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify-Rate	Mean Improvement	Std.	Runtime
No DR	59.592	---------	10.537	---------	58.297	---------	8.969	---------
PCA	58.454	-1.138	8.412	0.139	59.886	1.589	8.766	0.135
Laplacian	56.986	-2.606	11.685	6.773	58.941	0.644	13.543	6.665
LLE	57.920	-1.673	13.463	42.166	56.902	-1.395	11.176	42.075
LTSA	59.034	-0.557	14.345	36.003	60.872	2.575	14.400	34.660
LLTSA	58.059	-1.533	11.129	34.618	56.583	-1.714	13.104	34.030
Diff-Maps	**67.064**	*7.473*	13.7	7.792	**66.250**	*7.953*	14.281	7.581

Table 3: Clean Urban 162 bands

	6 Dimensions				4 Dimensions			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	61.141	---------	10.723	---------	61.481	---------	12.007	---------
PCA	60.544	-0.597	10.682	0.091	61.109	-0.372	12.949	0.090
Laplacian	56.932	-4.209	10.964	5.729	57.775	-3.706	12.060	5.635
LLE	64.960	3.819	14.000	19.005	66.143	4.662	14.555	19.074
LTSA	**68.279**	*7.139*	13.656	17.445	**67.426**	*5.945*	13.725	17.344
LLTSA	65.636	4.496	13.557	17.690	64.699	3.219	11.529	17.743
Diff-Maps	67.736	6.595	13.403	6.962	66.203	4.723	12.519	6.750

Table 4: Noisy Indian Pines 220 bands

	6 Dimensions				4 Dimensions			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	69.821	---------	17.062	---------	66.810	--------	11.241	--------
PCA	72.220	2.399	15.345	0.170	71.143	4.333	7.812	0.165
Laplacian	77.808	7.987	15.231	5.501	72.470	5.660	15.082	5.399
LLE	73.479	3.658	13.788	34.681	69.848	3.038	15.605	34.577
LTSA	77.069	7.248	11.284	29.792	76.607	9.80	11.890	28.816
LLTSA	**78.099**	*8.278*	13.103	28.706	78.597	11.787	14.981	28.668
Diff-Maps	77.852	8.031	14.013	6.133	**79.515**	*12.705*	15.597	5.946
FastICA	72.470	2.649	18.495	7.274	72.470	5.660	18.495	7.2744

Table 5: Clean Indian Pines 179 bands

	6 Dimensions				4 Dimensions			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	75.884	--------	17.230	--------	72.793	---------	9.939	--------
PCA	73.006	-2.878	15.480	0.122	72.826	0.033	7.664	0.187
Laplacian	73.249	-2.635	14.511	4.886	72.290	-0.503	13.685	4.899
LLE	79.886	4.002	15.943	13.899	81.139	8.346	15.999	14.024
LTSA	80.912	5.028	10.374	13.101	70.124	-2.669	10.091	13.356
LLTSA	80.937	5.053	12.965	13.440	78.901	6.108	13.529	13.294
Diff-Maps	**86.531**	*10.647*	16.541	5.635	**83.612**	*10.818*	13.602	5.430
FastICA	73.787	-2.097	14.969	6.259	73.063	0.270	15.389	4.179

Experiments on clean Indian Pines 179 bands with dimension reduction to four dimensions: May Compare with Table 4, Column B. (highlighted)

Table 6: Cosine or Correlation Distance

	4 Dimensions Cosine Distance				4 Dimensions Correlation Distance			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	77.925	--------	12.327	--------	80.386	--------	13.782	--------
PCA	74.615	-3.310	9.265	0.116	79.617	-0.769	12.474	0.136
Laplacian	76.661	-1.263	12.562	4.838	72.888	-7.498	12.190	5.240
LLE	79.209	1.285	12.696	16.363	75.823	-4.563	16.707	18.995
LTSA	**82.221**	*4.296*	13.869	15.124	**82.914**	*2.528*	12.054	17.074
LLTSA	77.972	0.048	11.692	14.784	LLTSA and Diff-Maps produced errors in original code when running with correlation distance.			
Diff-Maps	68.736	-9.189	18.085	2.836				
FastICA	74.438	-3.4872	13.233	28.414	74.465	-5.921	13.430	24.231

Table 7: First perform PCA to reduce to 100 dimensions then apply dimension reduction

	First do PCA then do Dimension Reduction			
Method	Classify-Rate	Mean Improvement	Std.	Runtime
No DR	72.793	--------	9.939	--------
PCA	72.826	0.033	7.664	0.044
Laplacian	72.021	-0.772	13.966	4.131
LLE	**82.683**	*9.890*	13.459	12.893
LTSA	67.401	-5.392	13.229	11.931
LLTSA	79.952	7.158	17.005	11.573
Diff-Maps	74.705	1.912	21.806	2.718
FastICA	71.044	-1.749	13.105	2.880

Table 8: Wavelet transforms the data then applies dimension reduction

	Do wavelet transforms before Dimension Reduction			
Method	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	72.847	--------	16.196	--------
PCA	70.257	-2.590	7.722	0.131
Laplacian	70.329	-2.519	13.047	4.895
LLE	69.984	-2.864	10.160	15.788
LTSA	**78.54**	*5.693*	12.075	14.773
LLTSA	70.560	-2.287	16.581	16.360
Diff-Maps	76.127	3.279	19.206	3.687
FastICA	76.160	3.313	17.995	3.968

Table 9: Spatial Coherent, comparing each pixels based on their local surroundings

	Spatial Coherent with 3 x 3 windows				First apply PCA then apply 3 x3 spatial coherent			
Method	Classify-Rate	Mean Improvement	Std.	Runtime	Classify -Rate	Mean Improvement	Std.	Runtime
No DR	74.802	--------	14.958	--------	68.302	--------	8.919	--------
PCA	71.354	-3.447	8.550	41.6	69.574	1.272	14.956	0.046
Laplacian	**77.915**	*3.113*	14.576	21.3	78.005	9.704	14.695	4.466
LLE	74.114	-0.688	14.878	43.9	**80.682**	*12.38*	12.028	19.376
LTSA	73.794	-1.007	12.492	51	78.414	10.112	15.431	16.450
LLTSA	72.115	-2.687	12.403	59.8	75.094	6.792	15.547	15.606
Diff-Maps	71.891	-2.91	11.725	24.1	68.614	0.312	13.704	11.171

Reference:

[1] Laurens van der Maaten, "Matlab Toolbox for Dimensionality Reduction v0.3b". < http://www.cs.unimaas.nl/l.vandermaaten>

[2] Laboratory of Computer and Information Science (CIS). Helsinki University of Technology, "FastICA". <http://www.cis.hut.fi/projects/ica/fastica/>

[3] US Army Topographic Engineering Center, "HyperCube." <http://www.tec.army.mil/Hypercube/>

[4] Purdue Research Foundation, "MultiSpec©." <http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/>

[5] A. Mohan, G. Sapiro and E. Bosch, "Spatially coherent Nonlinear Dimensionality Reduction and Segmentation of Hyperspectral Images," *IEEE Geosic. Remote Sens.*, vol. 4, no.2, pp.206-210, April 2007.