Robust classification of hyperspectral images

Anne Schistad Solberg Asbjørn Berge and Are F. C. Jensen

Department of Informatics, University of Oslo, P.O. Box 1080 Blindern, 0314 Oslo, Norway

ABSTRACT

This paper discusses robust classification of hyperspectral images. Both methods for dimensionality reduction and robust estimation of classifier parameters in full dimension are presented. A new approach to dimensionality reduction that uses piecewise constant function approximation of the spectral curve is compared to conventional dimensionality reduction methods like principal components, feature selection, and decision boundary feature extraction. Computing robust estimates of the decision boundary in full dimension is an alternative to dimensionality reduction. Two recently proposed techniques for covariance estimation based on the eigenvector decomposition and the Cholesky decomposition are compared to Support Vector Machine classifiers, simple regularized estimates, and regular quadratic classifiers. The experimental results on four different hyperspectral data sets demonstrate the importance of using simple, sparse models. The sparse model using Cholesky decomposition in full dimension performed slightly better than dimensionality reduction. However, if speed is an issue, the piecewise constant function approximation method for dimensionality reduction could be used.

Keywords: Classification, dimensionality reduction, covariance matrix estimation

1. INTRODUCTION

Hyperspectral imaging involves detailed sampling of the reflected sunlight in wavelengths typically from 0.4 to 3 μm. Hyperspectral sensors record the sunlight in 50 to several hundred narrow spectral bands. The number of bands produced by available sensors range from 30 to several hundreds, however there is no universally agreed upon minimum number of bands or bandwidth dimension required for a dataset to be considered hyperspectral.

Classification of pixels in hyperspectral images is a complex problem. We usually have few samples available for training the classifier, and the ratio between available ground truth samples and dimensionality of the input vector is low. When using parametric methods, such as the Gaussian maximum-likelihood approach, the parameter estimates (in particular the estimated covariance matrix) will become increasingly unstable when the number of training samples is low.

The conventional approaches to dealing with problems of dimensionality when training data is sparse, are dimensionality reduction, improving the ratio of training data to dimensionality, and/or some sort of stabilization of parameter estimates. Stabilization of parameter estimates can be done directly by assuming some simpler parameter structure or biasing the estimates toward simpler and more stable estimates, or indirectly by increasing the training sample size.

Consider a classification problem with K classes, assuming class conditional distributions to be Gaussian with mean μ_k and class-wise covariance matrices Σ_k. It is well known that this reduces to comparing the K quadratic discriminant functions $g_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k) + \log\pi_k$, where $k \in \{1,...,K\}$ and π_k is the a priori probability for class k. The parameters of these distributions are usually calculated from the maximum likelihood estimates and plugged into the above rule. This decision rule is commonly referred to as a Gaussian Maximum Likelihood (GML) classifier. The common problem with the GML classifier is that the number of parameters to estimate grows quadratically with the dimensionality d of the feature space, since Σ_k has $d(d+1)/2$ parameters.

The number of free parameters of the classifier must be restriced to produce a robust classifier. There are several ways of doing this. The most common approach is to apply dimensionality reduction in terms of either feature selection or transformation. An alternative is to use robust methods for parameter sparsing in full dimensionality, allowing mean differences between spectral bands to be retained. In this paper, we apply both strategies using state-of-the art algorithms for dimensionality reduction and sparse estimation in full dimension. We compare the performance of the different approaches on several different hyperspectral images.

E-mail of Anne Solberg: anne@ifi.uio.no

2. DIMENSIONALITY REDUCTION

Both classical feature reduction techniques from the pattern recognition literature and techniques developed with hyperspectral imagery in mind can be applied to reduce the dimensionality of hyperspectral images. The classical techniques include sequential forward selection, forward-backwards selection, floating search algorithms, and the recent approach for fast feature reduction proposed in.[1,2] A drawback of using feature selection is that class mean differences in the features not selected are ignored. For the transform methods, all the original spectral bands are used, but they often require a scatter or covariance estimate in full dimension.

Linear transformations like principal component analysis and Fisher's linear discriminant are well known. The decision boundary feature extraction approach (DBFE)[3] is commonly used in the literature on hyperspectral data. The general idea behind DBFE is to find a set of normal vectors to the intersection points between between the decision boundary and samples from different classes, and then eigenanalyze the outer product matrix of these vectors. The principal components of these points are argued to describe the most important vectors in the decision boundary. Another popular feature extraction method developed for hyperspectral images is the nonparametric weighted feature extraction (NWFE) proposed in[4] , which can be viewed as a nonparametric extension of Fisher's linear discriminant. In Fisher's linear discriminant, the between-class scatter is of deficit rank. In NWFE this is overcome by redefining this scatter to represent the scatter between all samples and a distance weighted mean.

2.1 Piecewise constant function approximation

We recently proposed[5] a method for fast piecewise constant function approximation that we will describe in the following. Spectral curves are divided into contiguous regions by piecewise constant function approximations. The extracted constants are then used as new features. The number of segments, and their corresponding length, are found by dynamic programming.

The basic idea behind this approach is to parition the hyperspectral signatures into a fixed number of contiguous intervals with constant intensities by minimizing the mean square representation error.

Let S_{ij} be the jth feature in the ith pixel of a dataset with a total of N pixels with M features. $S = \{S_{ij} | 1 \leq i \leq N, 1 \leq j \leq M\}$ is thus the collection of all spectral curves (the entire training dataset) available regardless of class belonging. We seek a set of K breakpoints, $P = p_1, p_2, .., p_K$, which define the contiguous intervals, $I_k = [p_k, p_{k+1})$. Note that the K breakpoints are equal for all the different classes. Each interval, for each pixel i, is represented by a constant, $\mu_{ik} \in \mathcal{R}$. The square representation error of the model is thus

$$H = \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{j \in I_k} (S_{ij} - \mu_{ik})^2. \tag{1}$$

If the breakpoints, P, are given, one ends up with a simple square function w.r.t. the constants, μ_{ik}, so the minimizer of (1) w.r.t. μ_{ik} is given by, letting $|I_k|$ denote the number of elements in the k-th interval,

$$\mu_{ik} = \frac{1}{|I_k|} \sum_{j \in I_k} S_{ij}, \tag{2}$$

i.e., the mean value of each pixel's interval between breakpoints. What is left to be determined to minimize (1) is thus the locations of the breakpoints. Note that when the breakpoints are given the segment lenghts are also given. These breakpoints can be found using dynamic programming. The detalied algorithm for this can be found in[5] .

After finding the breakpoints, the constants, μ_{ik}, are applied as features in the classification process. The optimal number of breakpoints and their location is estimated using cross validation. An example of the function approximation technique is shown in Figure 1.

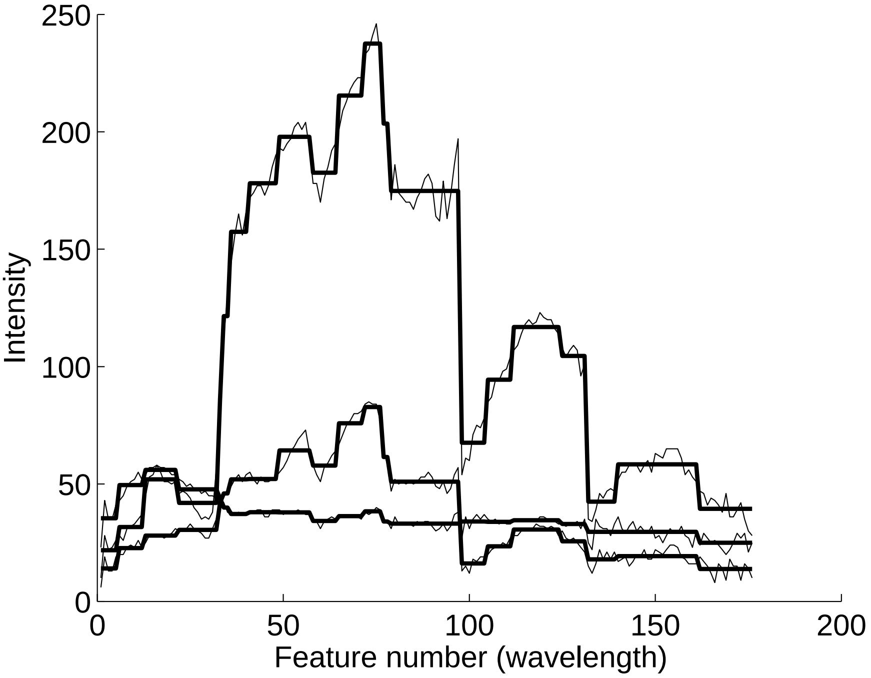

Figure 1. Example of constant function approximation for the KSC data set. The number of segments (K+1) was 21.

3. CLASSIFICATION IN FULL DIMENSION

Computing robust estimates of the decision boundary in full dimension is an alternative to dimensionality reduction. Consider a Gaussian classifier with mean vector μ_k and covariance matrix Σ_k. The simplest approach to reduce the number of parameters is to assume that the features are uncorrelated. Classical methods for robust covariance matrix estimation are e.g. regularized discriminant analysis[6] where the covariance matrix is given as

$$\Sigma_{kRDA} = (1-\alpha)\hat{\Sigma} + \alpha\hat{\Sigma_k},$$

where $\hat{\Sigma}$ is the regular sample covariance matrix, $\hat{\Sigma_k}$ is the classwise sample covariance matrix, and the parameter α is determined using cross validation. In standard ridge regression, a small penalty term is added to the diagonal of the covariance matrix:

$$\Sigma_{Ridge} = \hat{\Sigma} + RI,$$

where R is a penalization parameter.

Support Vector Machine (SVM) classifiers have recently been reported to perform well in classifying hyperspectral images[7–9] . Some studies claim that SMVs are insensitive to dimesionality issues and overtraining, while others argue that this is not true[10] . An advantage with SVMs is that they can be used to design arbitrary complex decision boundaries. The SVM is a kernel method, which means that it measures sample distance in some space implicitly defined by a weighting function. A common kernel is the Gaussian radial basis function (RBF). To avoid overfitting to the data, the SVM formulation has a tuning parameter acting as a regularizer on the decision boundary, which needs to be carefully adjusted to ensure good generalization performance. This regularization is measured as the cost of misclassifying a training sample. In addition, the width of the Gaussian kernel must be determined.

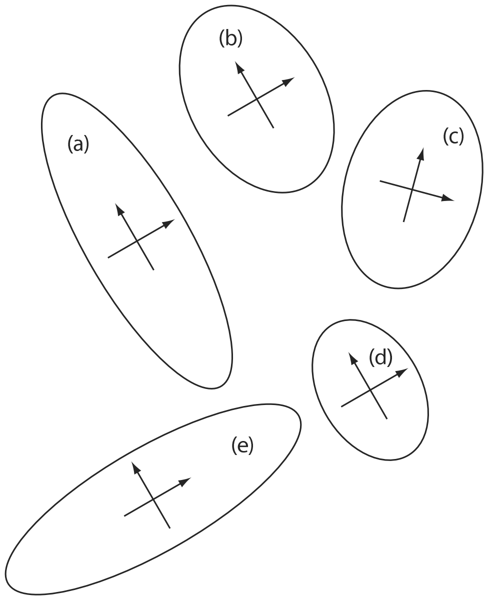

Figure 2. Example of different structures for 2D covariance matrices.

3.1 Structured eigenvector decomposition of the covariance matrix

In[11] we presented a model where eigenvector decomposition is used to decompose the covariance matrix. We use the modified eigenvalue decomposition of the covariance matrix estimate

$$\Sigma_{SMOG} = \lambda_k D_k A_k D'_k$$

where λ_k measures volume or scale, A_k shape, and D_k orientation. Common structures in the covariance matrices are obtained by forcing some or all of these to be equal for a subset of the components. $\lambda_k = |\Sigma_{SMOG}|^{1/d}$ is a proportional scaling parameter for the eigenvalues A_k, and D_k is the principal directions of the hyperellipsoid corresponding to the covariance matrix. How sharing of some of these structures affects the shape of the hyperellipsoids representing the covariance matrices is illustrated in Figure 2. The ellipsoids for (a), (b), (d) and (e) share orientation (note that the eigenvectors are common). (b), (c) and (d) have common shape, and (b) and (d) also share orientation but have different scale. The pairs {(a),(e)} and {(b),(c)} share scale.

The possible non-Gaussianity of the data is modelled using mixtures of Gaussians. Each class is modelled as a mixture of M Gaussian components. This decomposition of the covariance matrix allows a wealth of possible models for sharing of parameters. To build a classifier model, a model structure for deciding which parameters (λ_k, A_k and D_k) that should be shared between different classes (or subclasses in the Gaussian mixtures) must be determined. The primary goal is to use as few parameters as possible, thus the algorithm starts with a structure where all classes or components share all parameters. This corresponds to starting with a common covariance matrix. Parameters are then released one at a time. Since the number of parameters to estimate for each λ_k, A_k and D_k is 1, d and $d(d-1)/2$ respectively, they are released in that order.

A search for finding the best structure is performed, starting with a structure in which all components share both cluster shape, volume and scale using the following algorithm:

1. Initialization - Assume that all parameters D_k, A_k, λ_k are shared between all M components and find the 10-CV error when using this model.

2. Search: For each of the parameters λ_k, A_k and D_k do the following model search:

   (a) Initialize the table with value $i = 1$.

$$L = \begin{bmatrix} 1 & & & & \\ -\alpha_{2,1} & 1 & & & \\ -\alpha_{3,1} & -\alpha_{3,2} & 1 & & \\ \vdots & & & \ddots & \\ -\alpha_{p,1} & & -\alpha_{p,2} & -\alpha_{p,p-1} & 1 \end{bmatrix}$$

Figure 3. Illustration of a matrix of correlations, L, for the inverse covariance matrix . The matrix is lower triangular, with ones on the diagonal. Sparsity in the covariance estimate is obtained by only estimating the matrix elements in *some* off-diagonal vectors.

(b) From all possible components with table value i, find the component k that gives the largest decrease in 10-CV error when the parameter corresponding to this component is estimated freely.

(c) Allow component k to be estimated freely and repeat from b) until 10-CV error stops decreasing.

(d) Define a new set $i = i + 1$ of shared parameters over all (zero) marked components in the table and repeat from b) until 10-CV error stops decreasing.

The number of mixture components, M, is also determined using crossvalidation, see.[11] Parameter estimates of λ_k, A_k and D_k are found in.[11] Adding a small value to the diagonal of the scatter matrix, similar to ridge penalization in regression can be used to stabilize singular or near singular scatter matrices before eigenvector decomposition.

3.2 Cholesky decomposition of the inverse covariance matrix

In[12] we presented an approach where Cholesky decomposition of the inverse covariance matrix is utilized. Well known results from the time series literature[13, 14] relates the estimation of the inverse covariance matrix to a series of regressions by using the Cholesky decomposition. Consider the discriminant functions for a Gaussian classification problem with K classes

$$g_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)'\Sigma_k^{-1}(x-\mu_k) + \log\pi_k$$

where π_k is the a priori probability for class k. Noting that $-\log|\Sigma_k| = log|\Sigma_k^{-1}|$, it is clear that there is no need for matrix inversion when classifying data if we have a method for estimating the inverse covariance matrices directly.

The Cholesky decomposition of the inverse covariance matrix is given by

$$\Sigma^{-1} = LDL^T,$$

where L is a lower triangular matrix with ones on the diagonal (see Figure 3) and D a diagonal matrix.

If we were to consider the features of each sample as a time-series, the elements in L can be seen row-wise as parameters in autoregressive processes of the same order as the row number minus one. We use this to transform the task of approximating covariance matrices into a sequence of regressions. For each row, r, one could then predict the next feature x_r based on the $r-1$ preceding features, $\{x_1, ..., x_{r-1}\}$. In keeping with the earlier notation, and assuming zero mean for readability, this can be expressed as minimizing the squared residual in:

$$x_r = \sum_{j=1}^{r-1} \alpha_{r,j} x_j + \epsilon_r.$$

As long as the diagonal elements of D are positive, any choice of α will produce a positive definite covariance matrix.

Sparsity in the inverse covariance matrix can thus be obtained by fixing some α to be zero. Setting elements in the L-matrix to zero is thus a way of sparsing the inverse covariance matrix. A full search of testing all possible elements of L individually is not computationally feasible, so a heuristic must be defined. We use a diagonal pattern where diagonal vectors are added if this increases the crossvalidation performance. One diagonal vector corresponds to a certain lag (using time series terminology) in the time series expression above, indicating that when predicting feature x_r, correlations with certain features x_{r-t} are added. A search over the best lags, t, is then performed, and a given lag is added if it improved the cross-validation performance.

The general idea is to start by approximating the covariance matrices with the simplest possible model, i.e., diagonal matrices, and add parameters to the approximation until the classification performance of the model no longer improves. With regard to our proposed heuristic, we search for the off-diagonal vectors in the class-wise covariance matrices that need to be estimated in order to improve classification performance on the training data. The search, guided by cross-validation (10-CV) as a performance measure, can be described by the following steps:

1. Initialization - Approximate class-wise covariance matrices by diagonal matrices, and find 10-CV performance.

2. Search - Select off-diagonal vectors in L_k to be nonzero in a sequential forward manner:

   (a) For all zero off diagonal vectors in L_k, evaluate the 10-CV performance gain when allowing each to have nonzero elements.

   (b) Add the one off-diagonal vector that gives the largest improvement in 10-CV to the set of off-diagonal vectors to be nonzero in L_k.

   (c) Loop from a) until 10-CV performance does not improve further.

Maximum likelihood estimates for estimating all the parameters involved can be found in.[12] All classes share a common set of alphas fixed to zero.

4. EXPERIMENTAL RESULTS

4.1 Data sets

The classification approaches are tested on four different data sets. Not all of the methods were tested on all data sets as the experiments were run in a sequential order. The four hyperspectral datasets contain widely different types of data, and with dimensions ranging from 81 to 176 bands. The first dataset, ROSIS, is from an airborne sensor, contains forest type data, is divided into three classes, has 81 spectral bands and has a pixel size of 5.6m. The second data set, Pavia, is also from an airborne sensor (DAIS) depicting urban landcover pixels over Pavia, Italy. The dataset has 71 bands, 9 different landcover classes, and 2.6m pixels. The third set, KSC [12], is from an airborne sensor (AVIRIS), contains vegetation type data, divided into 13 classes, has 176 spectral bands and has 18m pixels. The last dataset, BOTSWANA [12], is from a scene taken by the Hyperion sensor aboard the EO-1 satellite, contains vegetation type data, is divided into 14 classes, has 145 bands, and has a 30m pixel size. The average number of training pixels per class is 700, 100, 196 and 115 for the datasets in sequential order. 10-CV was used for parameter estimation in all models. SVM parameters C_{err} (misclassification penalty) and the width of the Gaussian kernel were found by grid search.

4.2 Dimensionality reduction experiments

The performance of PCA, sequential forward selection using the sum of the estimated Mahalanobis distances as feature evaluation function, decision boundary feature extraction (DBFE), and the piecewise constant function approximation (PieceConst) is compared. PCA was selected for comparison because it is fast and simple, while DBFE is more elaborate since it includes class separability in the optimization process. Only the simplest feature selection algorithm (sequential forward search (SFS)) was used, the more complex feature selection algorithms were omitted due to their excessive execution times.

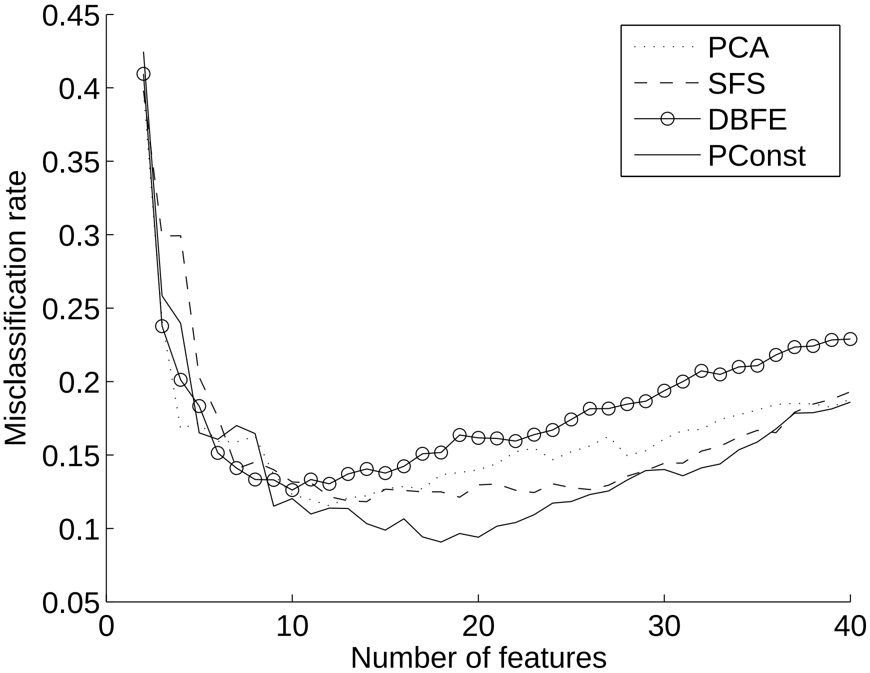

Figure 4. Misclassification rate for different numbers of features on the KSC dataset. Crossvalidation estimated 12, 14, 16 and 20 features for PieceConst, PCA, SFS and DBFE.

The number of features for each approach, that is, the number of principal components for PCA, the number of features for SFS, and the number of contiguous intervals in PieceConst, were chosen using 10-CV on the training sets.

Table 1 shows the overall classification accuracy for the datasets ROSIS, KSC and PAVIA when the optimal number of features is chosed using cross-valiation. Overall, PieceConst performs equal to or better than the other dimensionality reduction metods. Figures 4-5 shows the performance as a function of dimensionality. For the BOTSWANA data set the optimal number of features was 12, 14, 16 and 20 for PieceConst, PCA, SFS and DBFE respectively. For the KSC data set 21, 14, 18 and 11 features were selected for PieceConst, PCA, SFS and DBFE. We note that most of the dimensionality reduction algorithms have fairly similar performance over a range of dimensions. The plots only show performance up to 40 features, after that the classification error increases further.

Table 1. Classification accuracy for dimensionality reduction algorithms.

	ROSIS	KSC	BOTSWANA
PCA	85.21	87.78	95.55
SFS	85.07	87.51	94.72
DBFE	83.56	86.67	93.8
PieceConst	85.82	89.84	96.26

An advantage with the Piecewise Constant Function Approximation approach is that it allows direct interpretation of the selected features (SFS also gives interpretable results), while PCA and DBFE results in linear combinations of the original features and can not be directly interpreted. The PieceConst method is fast compared to the other methods (except PCA).

4.3 Classification in full dimension

The following experiments are done on the original, full-dimensional feature vectors without dimensionality reduction. We compare the performance of regular quadratic classifiers (QDA) with and without penalized covariance

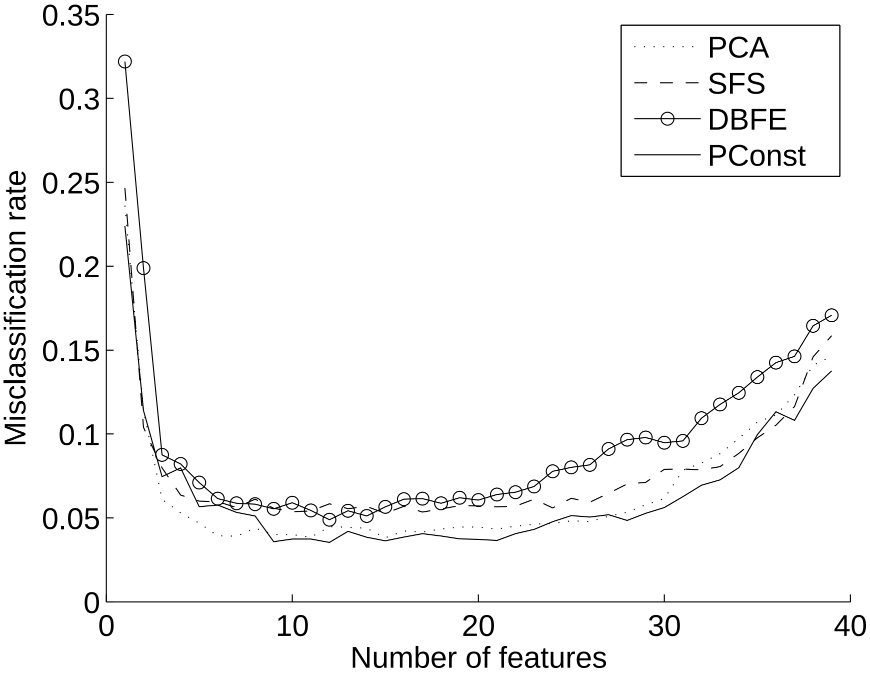

Figure 5. Misclassification rate for different numbers of features on the Botswana dataset. Crossvalidation estimated 12, 14, 16 and 20 features for PConst, PCA, SFS and DBFE.

matrix estimation to Support Vector Machine (SVM) classifiers, the eigenvector decomposition approach (called SMOG) and the Cholesky decomposition of the inverse covariance matrix (called STIC). A SVM with radial basis kernels was tuned using 10-CV. The classification accuracies for the various approaches are shown in Table 2. Regularization significantly improves the performance on all data sets. On the KSC and BOTSWANA data sets (with the highest dimensionality), ordinary QDA broke down due to sample sparsity. For SMOG, we note that using mixture of Gaussians did not improve the classification accuracy significantly (more results can also be found in[11]). The standard regularization approaches, LOOC and ridge regression, were outperformed by SMOG, SVM and STIC. On the ROSIS data set, SVM was slightly better than STIC, but on the other data sets STIC performed the best.

We can also compare the fraction of the parameters estimated in relation to a full QDA. For the ROSIS data set, STIC used 23%, SMOG 42%, and SMOG+MIX 41% of the parameters involved in regular QDA. For the Pavia data set, the corresponding fractions were 17% for STIC, 27% for SMOG, and 33% for SMOG+MoG. STIC typically performed better than SMOG, and using sparser matrices.

Table 2. Classification accuracies for classification in full dimension using different methods. SMOG was not run on all datasets, and missing values are denoted with '-'.

	ROSIS	Pavia	KSC	BOTSWANA
QDA	81.2	84.3	13	14.3
QDA ridge	84.2	87.1	-	-
QDA LOOC	84.9	90.8	88.5	96.0
SMOG	84.7	92.0	-	-
SMOG + MoG	85.2	91.7	-	-
SVM	87.10	92.5	89.8	96.0
STIC	85.7	93.8	91.1	97.1

4.4 Dimensionality reduction vs. robust classification in full dimension

The performance of the best dimensionality reduction method, PieceConst, is compared to the performance of the best methods for classification in full dimension in Table 3. We note that overall, STIC has a slight edge on SVM and PieceConst. However, the difference is quite small.

Table 3. Classification accuracies for the best dimensionality reduction method compared to the best classification in full dimension.

	ROSIS	KSC	BOTSWANA
PiecewiseConst	85.8	89.8	96.3
STIC	85.7	91.1	97.1
SVM	87.1	89.8	96.0

The execution times for the PieceConstant method were between 19 and 239 seconds, while the corresponding numbers for STIC were 11-127 minutes and 218-276 minutes for SVM. Thus dimensionality reduction was much faster than robust classification in full dimension.

The performance of STIC seems reasonably robust over a range of increasing number of parameters (off-diagonal vectors) (see Figure 6). These figures illustrate the performance as a function of the number of parameters estimated compared to a full QDA model.

5. DISCUSSION AND CONCLUSIONS

In this paper, we have presented and evaluated different methods for dimensionality reduction and robust, sparse estimation in full dimension for the task of classifying hyperspectral images. For dimensionality reduction, a new technique for fast piecewise constant function approximation (PieceConst) performed equal to or better than principal component analysis, sequential forward feature selection and decision boundary feature extraction. This was compared to classification in full dimension using simple, sparse models for parameter estimates. A new method for sparse covariance estimation based on Cholesky decomposition of the inverse covariance matrix (STIC) performed slightly better than SVM and eigenvector decomposition of the covariance matrix (SMOG).

When comparing dimensionality reduction to sparse estimation in full dimension, STIC had a slight edge on the piecwise constant method for dimensionality reduction. STIC has an advantage compared to the other methods for covariance estimation in that no estimate in full dimension (with all parameters non-zero) is needed. It is unclear if this difference is significant. If speed is an issue, the piecewise constant model for dimensionality reduction could be used. All experiments showed that simple, sparse models should be used when the classification problem is ill-posed.

Both PieceConst and STIC have an advantage compared to SVM in that the resulting sparse estimates are interpretable. Future work can include a study of how these methods can be used for visualization in exploratory data analysis.

REFERENCES

1. S. Serpico and L. Bruzzone, “A new search algorithm for feature selection in hyperspectral remote sensing images,” *IEEE Transactions on Geoscience and Remote Sensing* **39**, pp. 1360–1367, July 2001.
2. S. B. Serpico, M. D’Inca, F. Melgani, and G. Moser, “Comparison of feature reduction techniques for classification of hyperspectral remote sensing data,” in *Proc. SPIE, Image and Signal Processing for Remote Sensing VIII*, S. B. Serpico, ed., **4885**, pp. 347–358, 2003.
3. C. Lee and D. Landgrebe, “Feature extraction based on desicion boundaries,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, pp. 388–400, April 1993.
4. B.-C. Kuo and D. Landgrebe, “A robust classification procedure based on mixture classifiers and nonparametric weighted feature extraction,” *IEEE Transactions on Geoscience and Remote Sensing* **40**, pp. 2486–2494, November 2002.
5. A. C. Jensen and A. S. Solberg, “Fast hyperspectral feature reduction using piecewise constant function approximations,” *IEEE Geoscience and Remote Sensing Letters* **4**(4), pp. 547–551, 2007.

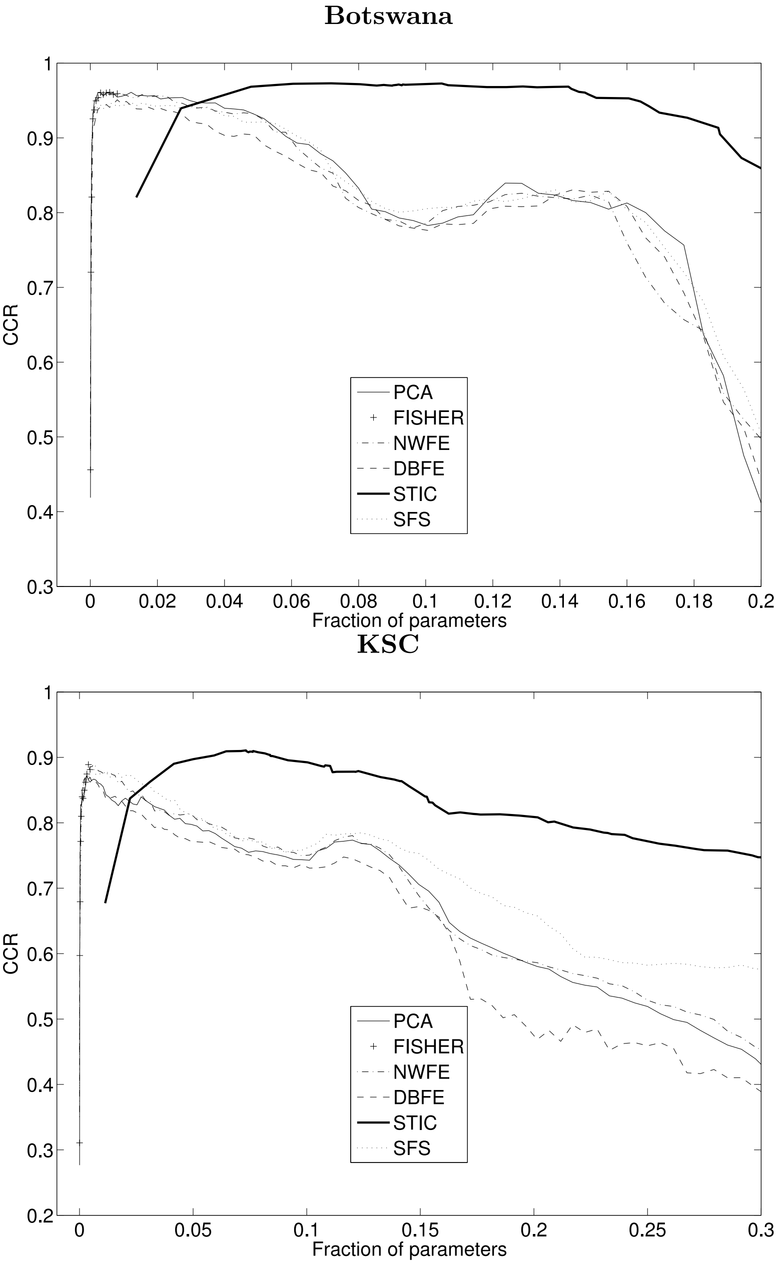

Figure 6. Correct classification rates on test data compared to the fraction of covariance parameters for a full model. All methods decay quite rapidly for cases using more than 30% of the parameters of a full model.

6. J. Friedman, "Regularized discriminant analysis," *J. Amer. Stat. Assoc.* **84**, pp. 165–175, March 1989.
7. C. Huang, L. Davis, and J. Townsend, "An assessment of support vector machines for landcover classification," *Int. J. Remote Sens.* **23**(4), pp. 221–232, 2002.
8. F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Transactions on Geoscience and Remote Sensing* **42**, pp. 1778–1790, August 2004.
9. G. Camps-Valls, L. Gomez-Chova, J. Calpe, E. Soria, J. Martin, L. Alonso, and J. Moreno, "Robust support vector method for hyperspectral data classification and knowledge discovery," **47**, pp. 1530–1542, July 2004.
10. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2001.
11. A. Berge and A. S. Solberg, "Structured gaussian components for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing* , pp. 3386–3396, November 2006.
12. A. Berge, A. C. Jensen, and A. S. Solberg, "Sparse inverse covariance estimates for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing* **45**, pp. 1399–1407, May 2007.
13. M. Pouhramadi, *Foundations of Time Series Analysis and Prediction Theory*, Wiley, 2001.
14. J. Whittaker, *Graphical Models in Applied Multivariate Statistics*, Wiley, 1990.