Kernel Multivariate Analysis Framework for Supervised Subspace Learning: A Tutorial on Linear and Kernel Multivariate Methods

Jerónimo Arenas-García, Senior Member, IEEE, Kaare Brandt Petersen, Gustavo Camps-Valls, Senior Member, IEEE, and Lars Kai Hansen

Abstract: Feature extraction and dimensionality reduction are important tasks in many fields of science dealing with signal processing and analysis. The relevance of these techniques is increasing as current sensory devices are developed with ever higher resolution, and problems involving multimodal data sources become more common. A plethora of feature extraction methods are available in the literature collectively grouped under the field of Multivariate Analysis (MVA). This paper provides a uniform treatment of several methods: Principal Component Analysis (PCA), Partial Least Squares (PLS), Canonical Correlation Analysis (CCA) and Orthonormalized PLS (OPLS), as well as their non-linear extensions derived by means of the theory of reproducing kernel Hilbert spaces. We also review their connections to other methods for classification and statistical dependence estimation, and introduce some recent developments to deal with the extreme cases of large-scale and low-sized problems. To illustrate the wide applicability of these methods in both classification and regression problems, we analyze their performance in a benchmark of publicly available data sets, and pay special attention to specific real applications involving audio processing for music genre prediction and hyperspectral satellite images for Earth and climate monitoring.

I. INTRODUCTION

As sensory devices develop with ever higher resolution and the combination of diverse data sources is more common, feature extraction and dimensionality reduction become increasingly important in automatic learning. This is especially true in fields dealing with intrinsically high-dimensional signals, such as those acquired for image analysis, spectroscopy, neuroimaging, and remote sensing, but also for situations when many heterogeneous features are computed from a signal and stacked together for classification, clustering, or prediction.

Multivariate analysis (MVA) constitutes a family of methods for dimensionality reduction successfully used in several scientific areas [1]. The goal of MVA algorithms is to exploit correlations among the variables to find a reduced set of features that are relevant for the learning task. Among the most well-known MVA methods are Principal Component Analysis (PCA), Partial Least Squares (PLS), and Canonical Correlation Analysis (CCA). PCA disregards the target data and exploits correlations between the input variables to maximize the variance of the projections, while PLS and CCA look for projections that maximize, respectively, the covariance and the correlation between the features and the target data. Therefore, they should in principle be preferred to PCA for regression or classification problems. In this paper, we consider also a fourth MVA method known as Orthonormalized PLS (OPLS) that is also well-suited to supervised problems, with certain optimality in least-squares (LS) multiregression. A common advantage of these methods is that they can be formulated using standard linear algebra, and can be implemented as standard or generalized eigenvalue problems. Furthermore, implementations and variants of these methods have been proposed that operate either in a blockwise or iterative manner to improve speed or numerical stability.

No matter how refined the various methods of MVA are, they are still constrained to account for linear input-output relations. Hence, they can be severely challenged when features exhibit non-linear relations. In order to address these problems, non-linear versions of MVA methods have been developed and these can be classified into two fundamentally different approaches [2]: 1) The modified methods in which the linear relations among the latent variables are substituted by non-linear relations [3], [4]; and 2) Variants in which the algorithms are reformulated to fit a kernel-based approach [5]–[7]. In this paper, we will review the second approach, in which the input data is mapped by a non-linear function into a high-dimensional space where the ordinary MVA problems are stated. A central property of this kernel approach is the exploitation of the so-called kernel trick, by which the inner products in the transformed space are replaced with a kernel function working solely with input space data so the explicit non-linear mapping is not explicitly necessary.

An appealing property of the resulting kernel algorithms is that they obtain the flexibility of non-linear expressions using straightforward methods from linear algebra. However, supervised kernel MVA methods are hampered in applications involving large datasets or a small number of labeled samples. Sparse and incremental versions have been presented to deal with the former problem, while the field of semisupervised learning has recently emerged for the latter. Remedies to these problems involve a particular kind of regularization, guided either by selection of a reduced number of basis functions or by considering the information about the manifold conveyed by the unlabeled samples. Both approaches will be also reviewed in this paper. Concretely, we aim:

1) To review linear and kernel MVA algorithms, providing their theoretical characterization and comparing their main properties under a common framework.
2) To present relations between kernel MVA and other
TABLE I
ACRONYMS AND NOTATION USED IN THE PAPER.

| Acronym | Meaning |
|---------|---------|
| AR | Autoregressive |
| ECMWF | European Centre for Medium-Range Weather Forecasts |
| GMM | Gaussian Mixture Model |
| HSIC | Hilbert-Schmidt Independence Criterion |
| IASI | Infrared Atmospheric Sounding Interferometer |
| (k)CCA | (Kernel) Canonical Correlation Analysis |
| kFD | Kernel Fisher Discriminant |
| (k)MVA | (Kernel) Multivariate Analysis |
| (k)OPLS | (Kernel) Orthonormalized Partial Least Squares |
| (k)PCA | (Kernel) Principal Component Analysis |
| (k)PLS | (Kernel) Partial Least Squares |
| MSE | Mean-square error |
| LDA | Linear Discriminant Analysis |
| LS | Least-squares |
| OA | Overall Accuracy |
| RBF | Radial Basis Function |
| rkCCA | reduced complexity kCCA |
| rkOPLS | reduced complexity kOPLS |
| rkPCA | reduced complexity kPCA |
| RMSE | Root-mean-square Error |
| RTM | Radiative transfer models |
| skPLS | Sparse kPLS |
| UCI | University of California, Irvine |
| l | Number of labelled dataset |
| u | Number of unlabeled data |
| n | Total number of training samples |
| d | Dimension of input space |
| m | Dimension of output space |
| X | Centered input data (size \( l \times d \)) |
| Y | Centered output data (size \( l \times m \)) |
| \( C_x \), \( C_y \) | Input, output data sample covariance matrices |
| \( C_{xy} \) | Input-output data sample cross-covariance matrix |
| \( X^\dagger \) | Moore-Penrose pseudoinverse of matrix \( X \) |
| W | Regression coefficients matrix |
| \( \Phi \) | Identity matrix |
| \( \|A\|_F \) | Frobenius norm of matrix \( A \) |
| \( n_f \) | Number of extracted features |
| \( u_i, v_i \) | \( i \)th projection vector for the input, output data |
| \( U, V \) | \( [u_1, \ldots, u_{n_f}], [v_1, \ldots, v_{n_f}] \). Projection matrices |
| \( X', Y' \) | Extracted features for the input, output data |
| \( \mathcal{F} \) | Reproducing kernel Hilbert Space |
| \( \phi(x) \) | Mapping of \( x \) in feature space |
| \( k(x_i, x_j) \) | \( \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{F}} \). Kernel function |
| \( \Phi = \Phi_{\Phi^\top} \) | Gram Matrix |
| \( A = [\alpha_1, \ldots, \alpha_{n_f}] \) | Coefficients for \( U = \Phi_{\Phi^\top} A \) |

feature extraction methods based on Fisher’s discriminant analysis and nonparametric kernel dependence estimates.

3) To review sparse and semisupervised approaches that make the kernel variants practical for large-scale or undersampled labeled datasets. We will illustrate how these approaches overcome some of the difficulties that may have limited a more widespread use of the most powerful kernel MVA methods.

4) To illustrate the wide applicability of these methods, for which we consider several publicly available data sets, and two real scenarios involving audio processing for music genre prediction and hyperspectral satellite images for Earth and climate monitoring. Methods will be assessed in terms of accuracy and robustness to the number of extracted features.

We continue the paper with a brief review of linear and kernel MVA algorithms, as well as connections to other methods. Then, Section III introduces some extensions that increase the applicability of kernel MVA methods in real applications. Section IV provides illustrative evidence of method’s performance. Finally, we conclude the paper in Section V with some discussion and future lines of research.

II. MULTIVARIATE ANALYSIS IN REPRODUCING KERNEL HILBERT SPACES

This section reviews the framework of MVA both in the linear case and with kernel methods. Interesting connections are also pointed out between particular classification and dependence estimation kernel methods. Table I provides a list of acronyms, and basic notation and variables that will be used throughout the paper.

A. Problem statement and notation

Let us consider a supervised regression or classification problem, and let \( X \) and \( Y \) be columnwise-centered input and target data matrices of sizes \( l \times d \) and \( l \times m \), respectively. Here, \( l \) is the number of training data points in the problem, and \( d \) and \( m \) are the dimensions of the input and output spaces, respectively. The target data can be either a set of variables that need to be approximated, or a matrix that encodes the class membership information. The sample covariance matrices are given by \( C_x = \frac{1}{l} X^\top X \) and \( C_y = \frac{1}{l} Y^\top Y \), whereas the covariance between the input and output data is \( C_{xy} = \frac{1}{l} X^\top Y \).

The objective of standard linear multiregression is to adjust a linear model for predicting the output variables from the input features, \( \hat{Y} = XW \), where \( W \) contains the regression model coefficients. Ordinary least-squares (LS) regression solution is \( W = X^\top Y \), where \( X^! = (X^\top X)^{-1}X^\top \) is the Moore-Penrose pseudoinverse of \( X \). Highly correlated input variables can result in rank-deficient \( C_x \), making the inversion of this matrix unfeasible. The same situation is encountered in the small-sample-size case (i.e., when \( l < d \), which is usually the case when using kernel extensions). Including a Tikhonov’s regularization term leads to better conditioned solutions: for instance, by also minimizing the Frobenius norm of the weights matrix \( \|W\|_F \), one obtains the regularized LS solution \( W = (X^\top X + \lambda I)^{-1}X^\top Y \), where parameter \( \lambda \) controls the amount of regularization.

The solution suggested by MVA to the above problem consists in projecting the input data into a subspace that preserves the most relevant information for the learning problem. Therefore, MVA methods obtain a transformed set of features via a linear transformation of the original data, \( Z = XU \), where \( U = [u_1, u_2, \ldots, u_{n_f}] \) will be referred hereafter as the projection matrix, \( u_i \) being the \( i \)th projection vector and \( n_f \) the
number of extracted features. Some MVA methods consider also a feature extraction in the output space, $Y' = YV$, with $V = [v_1, v_2, \ldots, v_n]$.

Generally speaking, MVA methods look for projections of the input data that are “maximally aligned” with the targets, and the different methods are characterized by the particular objectives they maximize. Table 3 provides a summary of the MVA methods that we will discuss in the rest of the section. An interesting property of linear MVA methods is that they are based on first and second order moments, and that their solutions can be formulated in terms of (generalized) eigenvalue problems. Thus, standard linear algebra methods can be readily applied.

### B. Linear Multivariate Analysis

Principal Component Analysis, which is also known as the Hotelling transform or the Karhunen-Loève transform [9], is probably the most widely used MVA method and the oldest dating back to 1901 [10]. PCA selects the maximum variance projections of the input data, imposing an orthonormality constraint for the projection vectors (see Table 3). PCA works under the hypothesis that high variance projections contain the relevant information for the learning task at hand. PCA is based on the input data alone, and is therefore an unsupervised feature extraction method. Methods that explicitly look for projections that better explain the target data should in principle be preferred in a supervised setup. Nevertheless, PCA and its kernel version, kPCA, are used as a preprocessing stage in many supervised problems, likely because of their simplicity and ability to discard irrelevant directions [11, 12].

The PLS algorithm [13] is based on latent variables that account for the information in $C_{xy}$. In order to do so, PLS extracts the projections that maximize the covariance between the projected input and output data, again under orthornormality constraints for the projection vectors. This is done either as an iterative procedure or by solving an eigenvalue problem. In the iterative schemes, the data sets $X$ and $Y$ are recursively transformed in a process which subtracts the information contained in the already estimated latent variables. This process, which is often referred to as deflation, can be done in a number of ways that define the many variants of PLS. Perhaps the most popular PLS method was presented in [14]. The algorithm, hereafter referred to as PLS2, assumes a linear relation between $X$ and $Y$ that implies a certain deflation scheme, where the latent variable of $X$ is used to deflate also $Y$ [7, p. 182]. Several other variants of PLS exist such as ‘PLS Mode A’ and PLS-SB; see [15] for a discussion of the early history of PLS and [16] for a well-written contemporary overview. Among its many advantages, PLS does not involve matrix inversion and deals efficiently with highly correlated data. This has justified its very extensive use in fields such as chemometrics and remote sensing, where signals typically are acquired in a range of highly correlated spectral wavelengths.

Rather than maximizing covariance, CCA maximizes the correlation between projected input and output data [17]. In this way, CCA can more conveniently deal with directions of the input or output spaces that present very high variance, and that would therefore be over-emphasized by PLS, even if the correlation between the projected input and output data is not very significant.

A final method we will pay attention to is Orthonormalized PLS (OPLS), also known as multilinear regression [18] or semi-penalized CCA [19]. OPLS is optimal for performing multilinear LS regression on the features extracted from the training data, i.e.,

$$U = \arg \min_U \|Y - X'W\|^2_F.$$  

with $W = X'Y$ being the matrix containing the optimal regression coefficients. It can be shown that this optimization problem is equivalent to the one stated in Table 3. Alternatively, this problem can also be associated with the maximization of a Rayleigh coefficient involving projections of both input and output data, $C_{xy}$.

It is in this sense that this method is called semi-penalized CCA, since it disregards variance of the projected input data, but rewards those input features that better predict large variance projections of the target data. This asymmetry makes sense in supervised subspace learning where matrix $Y$ contains target values to be approximated from the extracted input features. In fact, it has been shown that for classification problems OPLS is equivalent to Linear Discriminant Analysis (LDA), provided an appropriate labeling scheme is used for $Y$ [19]. However, in “two-view learning” problems in which $X$ and $Y$ represent different views of the data [7, Sec. 6.5], one would like to extract features that can predict both data representations simultaneously, and CCA could be preferred to OPLS.

A common framework for PCA, PLS, CCA and OPLS was proposed in [18], where it was shown that these methods can be reformulated as (generalized) eigenvalue problems, so that linear algebra packages can be used to solve them. Concretely:

**PCA**: $C_xu = \lambda u$

**PLS**: $\begin{pmatrix} C_{xy}^T & 0 \\ C_{xy} & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \end{pmatrix}$

**OPLS**: $C_{xy}C_{xy}^T u = \lambda C_x u$

**CCA**: $\begin{pmatrix} C_{xy}^T & 0 \\ C_{xy} & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_x & 0 \\ 0 & C_y \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$

We can see that CCA and OPLS require the inversion of matrices $C_x$ and $C_y$. If these are rank-deficient, then it becomes necessary to first extract the dimensions with non-zero variance using PCA, and then solve the CCA or OPLS problems. A very common approach is to solve the above problems using a two-step iterative procedure. In the first step, the projection vectors corresponding to the largest (generalized) eigenvalue are chosen, for which there exist

1Note that strictly speaking $U$ is not a projection operator, since it implies a transformation from $\mathbb{R}^d$ to $\mathbb{R}^{d'}$ and does not satisfy the idempotent property of projection operators. Nevertheless, if the columns of $U$ are linearly independent, vectors $u_i$ constitute a basis of the subspace of $\mathbb{R}^{d'}$ where the data is projected, and it is in this sense that we refer to $u_i$ and $U$ as projection vectors and matrix, and to $X' = XU$ as projected data. This nomenclature has been widely adopted in the machine learning field [8].
efficient methods such as the power method. The second step is known as deflation, and consists in removing from the data or covariance matrices the variance that can be already explained with the features extracted in the first step. Alternative solutions for these methods can be obtained by reformulating them as regularized least squares minimization problems. For instance, the work in [20]–[22] introduced sparse versions of PCA, CCA and OPLS by adding sparsity promotion terms, such as LASSO or \(l_1\)-norm on the projection vectors, to the LS functional.

Figure 1 illustrates the features extracted by the methods for a toy classification problem with three classes. The data was generated from three noisy sinusoid fragments, so that a certain overlap exists between classes. For the application of supervised methods, class membership is defined in matrix \(Y\) for a multi-view learning case. Here, we consider that it is known as \(Y\) which need to be well-represented by a certain overlap exists between classes. For the application of supervised methods, class membership is defined in matrix \(Y\) for a multi-view learning case. Here, we consider that it is known as \(Y\) which need to be well-represented by the extracted input features, so we will deal with the original \(Y\) as the new argument for the optimization.\(^2\) This is typically referred to as the kernel trick and has been used to develop kernel versions of the previous linear MVA, as indicated in the last four columns of Table II.

For PCA, it was Schölkopf, Smola and Müller who in 1998 introduced a kernel version denoted kPCA [6]. Lai and Fyfe in 2000 first introduced the kernel version of CCA denoted kCCA [24] (see also [7]). Later, Rosipal and Trejo presented a non-linear kernel variant of PLS in [25]. In that paper, \(K_x\) and the \(Y\) matrix are deflated the same way, which goes more in line with the PLS2 variant than to the traditional algorithm \(\text{PLS Mode A}\), and therefore we will denote it as kPLS2. A kernel variant of Orthonormalized PLS was presented in [26] and is here referred to as kOPLS. Specific versions of kernel methods to deal with signal processing applications have also been proposed, such as the temporal kCCA of [27], that is designed to exploit temporal structure in the data.

As for the linear case, kernel MVA methods can be implemented as (generalized) eigenvalue problems:

\[
K_x\alpha = \lambda\alpha
\]

\[
(0 \ K_y Y K_x 0) (\alpha_v) = \lambda (\alpha_v)
\]

\[
kOPLS : \ K_x YY^\dagger K_y \alpha = \lambda K_x K_y \alpha
\]

\[
kCCA : \ (0 \ K_y Y K_x 0) (\alpha_v) = \lambda (K_x K_y 0 C_y 0) (\alpha_v)
\]

It should be noted that the output data could also be mapped to some feature space \(H\), as it was considered for kCCA in [24] for a multi-view learning case. Here, we consider that it is the actual labels in \(Y\) which need to be well-represented by the extracted input features, so we will deal with the original representation of the output data.

For illustrative purposes, we have incorporated to Fig. 1 the projections obtained in the toy problem by kMVA methods using the radial basis function (RBF) kernel, \(k(x_i, x_j) = \exp\left(-||x_i - x_j||^2/(2\sigma^2)\right)\). Input data was normalized to zero mean and unit variance, and the kernel width \(\sigma\) was selected as the median of all pairwise distances between samples [28]. The same \(\sigma\) has been used for all methods, so that features are extracted from the same mapping of the input data. We can see that the non-linear mapping improves class separability. As expected, kPCA, kPLS and kCCA maximize in \(F\) the same variance, covariance and correlation objectives, respectively.

\(\text{TABLE II}
\]

| PCA | PLS | CCA | OPLS | kPCA | kPLS | kCCA | kOPLS |
|-----|-----|-----|------|------|------|------|-------|
| \(U^\dagger U = \mathbb{I}\) | \(U^\dagger U = \mathbb{I}\) | \(U^\dagger C_u U = \mathbb{I}\) | \(U^\dagger C_u U = \mathbb{I}\) | \(A^\dagger K_x A = \mathbb{I}\) | \(A^\dagger K_x Y Y^\dagger K_x A = \mathbb{I}\) | \(A^\dagger K_x Y Y^\dagger K_x A = \mathbb{I}\) | \(A^\dagger K_x Y Y^\dagger K_x A = \mathbb{I}\) |

Vectors \(u\) and \(\alpha\) are column vectors in matrices \(U\) and \(A\), respectively. \(r(\cdot)\) denotes the rank of a matrix.

C. Kernel Multivariate Analysis

The framework of kernel MVA (kMVA) algorithms is aimed at extracting nonlinear projections while actually working with linear algebra. Let us first consider a function \(\phi : \mathbb{R}^d \rightarrow \mathcal{F}\) that maps input data into a Hilbert feature space \(\mathcal{F}\). The new mapped data set is defined as \(\Phi = [\phi(x_1), \ldots, \phi(x_l)]^\top\), and the features extracted from the input data will now be given by \(\Phi = \Phi U\), where matrix \(U\) is of size \(\text{dim}(\mathcal{F}) \times n_f\). The direct application of this idea suffers from serious practical limitations when the dimension of \(\mathcal{F}\) is very large, which is typically the case.

To implement practical kernel MVA algorithms we need to rewrite the equations in the first half of Table II in terms of inner products in \(\mathcal{F}\) only. For doing so, we rely on the availability of a kernel matrix \(K_x = \Phi \Phi^\top\) of dimension \(l \times l\), and on the Representer’s Theorem [7], which states that the projection vectors can be written as a linear combination of the training samples, i.e., \(U = \Phi^\top A\), matrix \(A = [\alpha_1, \ldots, \alpha_n_f]\) being the new argument for the optimization.\(^2\) This is typically referred to as the kernel trick and has been used to develop kernel versions of the previous linear MVA, as indicated in the last four columns of Table II.

\(\text{TABLE II}
\]

| PCA | PLS | CCA | OPLS | kPCA | kPLS | kCCA | kOPLS |
|-----|-----|-----|------|------|------|------|-------|
| \(u^\top C_u u\) | \(u^\top C_u v\) | \(u^\top C_v v\) | \(u^\top C_v C_u u\) | \(\alpha^\top K_x \alpha\) | \(\alpha^\top K_x Y v\) | \(\alpha^\top K_x Y v\) | \(\alpha^\top K_x Y v\) |

Vectors \(u\) and \(\alpha\) are column vectors in matrices \(U\) and \(A\), respectively. \(r(\cdot)\) denotes the rank of a matrix.
as their linear counterparts. kOPLS looks for the directions with any linear projection of the target data. 

As already stated, close connections have been established between input-output data with a function that depends on 

Besides, in the recent years interesting connections of kMVA and statistical dependence estimates have been established. For instance, the Hilbert-Schmidt Independence Criterion (HSIC) is a simple yet very effective method to estimate statistical dependence between random variables. HSIC corresponds to estimating the norm of the cross-covariance in \( \mathcal{F} \), whose empirical (biased) estimator is

\[
\text{HSIC} := \frac{1}{n-1} \text{Tr}(K_x K_y),
\]

where \( K_x \) works with samples in the source domain and \( K_y = YY^T \). It can be shown that, if the RKHS kernel is universal, HSIC asymptotically tends to zero when the input and output data are independent. The so-called Hilbert-Schmidt Component Analysis (HSCA) method iteratively seeks for projections that maximize dependence with the target variables and simultaneously minimize the dependence with previously extracted features, both in HSIC terms. This objective translates into the iterative resolution of the generalized eigen-decomposition problem

\[
K_x K_y K_y \alpha = \lambda K_x K_y K_x \alpha,
\]

where \( K_F \) is a kernel matrix of already extracted features in the previous iteration. If one is only interested in maximizing source-target dependence in HSIC terms, the problem boils down to solving kOPLS.

Similarly, the connection between kCCA and other kernel measures of dependence, such as the kernel Generalized Variance (kGV) or the kernel Mutual Information (kMI), was introduced in [33]. The empirical kGV estimates dependence between input-output data with a function that depends on the entire spectrum of the associated correlation operator in RKHS, 

\[
k\text{GV}(\theta) = -\frac{1}{2} \log(\Pi_i (1 - \lambda_i^2)),
\]

where \( \lambda_i \) are the
solutions to the generalized eigenvalue problem
\[
\begin{bmatrix}
0 & K_x^y \\
K_x K_y & 0
\end{bmatrix}
\begin{bmatrix}
\alpha \\
v
\end{bmatrix} =
\lambda
\begin{bmatrix}
\theta K_x K_x + \eta (1 - \theta) K_x \\
0 & \theta K_y K_y + \eta (1 - \theta) K_y
\end{bmatrix}
\begin{bmatrix}
\alpha \\
v
\end{bmatrix},
\]

where \( K_x \) and \( K_y \) are defined using RKHS kernels obtained via convolution of the associated Parzen windows, \( \eta \) is a scaling factor, and \( \theta \) is a parameter in the range \([0, 1]\). Gretton et al. [33] showed that, under certain conditions, kGV reduces to kMI for \( \theta = 0 \) and to kCCA for \( \theta = 1 \) (cf. Eq. 4).

It is worth noting that the previous kernel measures of statistical dependence hold connections with Information Theoretic Learning concepts as well. For instance, it can be shown that HSIC is intimately related to the concept of correntropy [34]. All these connections could shed light in the future about the informative content of the extracted features in a principled way.

III. EXTENSIONS FOR LARGE SCALE AND SEMISUPERVISED PROBLEMS

Supervised kernel MVA methods are hampered either by the wealth or the scarcity of labeled samples, which can make these methods impractical for many applications. We next summarize some extensions to deal with large scale problems and semisupervised situations in which few labeled data is available.

A. Sparse Kernel Feature Extraction

A critical bottleneck of kernel methods is that for a dataset of \( l \) samples, the kernel matrices are \( l \times l \), which, even for a moderate number of samples, quickly becomes a problem with respect to both memory and computation time. Furthermore, in kernel MVA this is also an issue during the extraction of features for test data, since kernel MVA solutions will in general depend on all training data (i.e., matrix \( A \) will generally be dense): evaluating thousands of kernels for every new input vector is, in most applications, simply not acceptable. Furthermore, these so-called dense solutions may result in severe problems of overfitting, which is particularly true for kOPLS and kCCA [4, 29]. To address these problems, several solutions have been proposed to obtain sparse solutions that can be expressed as a combination of a subset of the training data, and therefore require only \( r \) kernel evaluations per sample (with \( r \ll l \)) for feature extraction. Note that, in contrast to the many linear MVA algorithms that induce sparsity with respect to the original variables, in this subsection we review methods that obtain sparse solutions in terms of the samples (i.e., sparsity in the \( \alpha_v \) vectors).

The approaches to obtain sparse solutions can be broadly divided into low rank approximation methods, that aim at working with reduced \( r \times r \) matrices (\( r \ll l \)), and reduced set methods that work with \( l \times r \) matrices. Following the first approach, the Nyström low-rank approximation of an \( l \times l \) kernel matrix \( K_l \) is expressed as \( \tilde{K}_l = K_{rl} K_{r}^{-1} K_{rl} \), where subscripts indicate row and column dimensions. The method was originally exploited in the context of Gaussian processes, and was later used in [35] to directly approximate the feature mapping itself rather than the kernel, thus giving rise to sparse versions of kPLS and kCCA.

Among the reduced set methods, a sparse kPCA (skPCA) was proposed by Tipping in [36], where the sparsity in the representation is obtained by assuming a generative model for the data in \( F \) that follows a normal distribution and includes a noise term with variance \( \nu_n \). The maximum likelihood estimation of the covariance matrix is shown to depend on just a subset of the training data, and so it does the resulting solution. A sparse kPLS (skPLS) was introduced in [37]. The method uses a fraction of the training samples for computing the projections. Each projection vector is found through a sort of \( \epsilon \)-insensitive loss similar to the one used in the support vector regression method. The sparsification is induced via a multi-step adaptation with high computational burden. In spite of obtaining sparse solutions, the algorithms from [36] and [37] still require the computation of the full kernel matrix during the training.

A reduced complexity kOPLS (rkOPLS) was proposed in [26] by imposing sparsity in the projection vectors representation \( \Phi^T \beta \), where \( \Phi_r \) is a subset of the training data containing \( r \) samples (\( r \ll l \)) and \( \beta \) is the new argument for the maximization problem, which now becomes:

\[
\max \beta^T \Phi_r \Phi_r^T \beta \quad \text{subject to :} \quad \beta^T \Phi_r K_{rl}^T \beta = 1,
\]

Since kernel matrix \( K_{rl} = \Phi_r \Phi_r^T \) involves the inner products in \( F \) of all training points with the patterns in the reduced set, this method still takes into account all data during the training phase, and is therefore different from simple subsampling. This sparsification procedure avoids the computation of the full kernel matrix at any step of the algorithm. An additional advantage of this method is that matrices \( K_{rl} \Phi_r \Phi_r^T K_{rl} \) and \( K_{rl}^T K_{rl} \) are both of size \( r \times r \), and can be expressed as sums over the training data, so the storage requirements become just quadratic with \( r \). Furthermore, the sparsity constraint acts as a regularizer that can significantly improve the generalization ability of the method. In the experiments section, we will apply the same sparsification procedure to kPCA and kCCA, obtaining reduced complexity versions of these methods to which we will refer to as rkPCA and rkCCA. Interestingly, the extension to kPLS2 is not straightforward, since the deflation step would still require the full kernel matrix \( K_l \).

Alternatively, two sparse kPLS schemes were presented in [38] under the name of Sparse Maximal Alignment (SMA) and Sparse Maximal Covariance (SMC). Here kPLS iteratively estimates projections that either maximize the kernel alignment (c.1) or the covariance (c.2) of the projected data and the true labels:

\[
\max \beta^T K_{jl} Y \quad \text{subject to (c.1):} \quad \beta^T K_j \beta = 1 \quad (c.1)
\]

subject to (c.2): \( \beta^T K_j = 1 \),

where \( K_j = K_x \) and \( K_j \) denotes the deflated kernel matrix at iteration \( j \), according to [38] Eq. (3). The method imposes the additional constraint that the cardinality of \( \beta \) is 1. This
restriction explicitly enforces sparsity through an $\ell_0$-norm in the weights space. At each iteration, $\beta$ is obtained by performing an exhaustive search over all training patterns. However, the complexity of the algorithm can be significantly reduced by constraining the search to just $p$ randomly chosen samples.

In Table III, we summarize some computational and implementation issues of the aforementioned sparse kMV A methods, and of standard non-sparse kMV A and linear methods. An analysis of the properties of each algorithm provides some hints that can help us choose the algorithm for a particular application. Firstly, a critical step when using kernel methods is the selection of an appropriate kernel function and tuning its parameters. To avoid overfitting, kMV A methods can be adjusted using cross-validation at the cost of higher computational cost. Sparse methods can help in this respect by regularizing the solution. Secondly, most methods can be implemented as either eigenvalue or generalized eigenvalue problems, whose complexity typically scales cubically with the size of the analyzed matrices. Therefore, both for memory and computational reasons, only linear MVA and the sparse approaches from [26] and [38] are affordable when dealing with large data sets. A final advantage of sparse kMV A is the reduced number of kernel evaluations to extract features for new data.

**B. Semisupervised Kernel Feature Extraction**

When few labeled samples are available, the extracted features do not capture the structure of the data manifold well, and hence using them for classification or regression may lead to very poor results. Recently, semisupervised learning approaches have been introduced to alleviate these problems. Two approaches are encountered: the information conveyed by the unlabeled samples is either modeled with graphs or via kernel functions derived from generative clustering models.

Notationally, we are given $l$ labeled and $u$ unlabeled samples, a total of $n = l + u$. The semisupervised kCCA (ss-kCCA) has been recently introduced in [28] by using the graph Laplacian. The method essentially solves the standard kCCA using kernel matrices computed with both labeled and unlabeled data, which are further regularized with the graph Laplacian:

\[
\begin{pmatrix}
0 & K_l^x K_l^y \\
K_l^x & 0 \\
K_l^y & 0
\end{pmatrix}
\begin{pmatrix}
\alpha \\
\nu
\end{pmatrix} =
\lambda
\begin{pmatrix}
K_l^x K_l^y + R_l^x & 0 \\
0 & K_l^y
\end{pmatrix}
\begin{pmatrix}
\alpha \\
\nu
\end{pmatrix},
\]

where $R_l^x = \alpha_x K_l^x + \gamma_x K_{nn}^x L_{nn}^x K_{nn}^y$ and $R_l^y = \alpha_y K_{nn}^y + \gamma_y K_{nn}^y L_{nn}^y K_{nn}^x$. For notation compactness, subindexes here indicate the size of the corresponding matrices while superscripts denote whether they involve input or output data. Parameters $\alpha_x, \alpha_y, \gamma_x$ and $\gamma_y$ trade off the contribution of labeled and unlabeled samples, and $L = D^{-1/2}(D - M)D^{1/2}$ represents the (normalized) graph Laplacian for the input and target domains, where $D$ is the degree matrix whose entries are the sums of the rows of the corresponding similarity matrix.

M, i.e. $D_{ii} = \sum_j M_{ij}$. It should be noted that for $n = l$ and null regularization, one obtains the standard kCCA (cf. Eq. [3]). Note also that this form of regularization through the graph Laplacian can be applied to any kMV A method. A drawback of this approach is that it involves tuning several parameters and working with larger matrices of size $2n \times 2n$, which can make its application difficult in practice.

Alternatively, cluster kernels have been used to develop semisupervised versions of kernel methods in general, and of kMV A methods in particular. The approach was used for kPLS and kOPLS in [39]. Essentially, the method relies on combining a kernel function based on labeled information only, $k_c(x_i, x_j)$, and a generative kernel directly learned by clustering all (labeled and unlabeled) data, $k_c(x_i, x_j)$. Building $k_c$ requires first running a clustering algorithm, such as Expectation-Maximization assuming a Gaussian mixture model (GMM) with different initializations, $q = 1, \ldots, Q$, and with different number of clusters, $g = 2, \ldots, G + 1$. This results in $Q \times G$ cluster assignments where each sample $x_i$ has its corresponding posterior probability vector $\pi(q, g) \in \mathbb{R}^q$. The probabilistic cluster kernel $k_c$ is computed by averaging all the dot products between posterior probability vectors,

\[
k_c(x_i, x_j) = \frac{1}{Z} \sum_{q=1}^{Q} \sum_{g=2}^{G+1} \pi(q, g)^\top \pi(q, g),
\]

where $Z$ is a normalization factor. The final kernel function is defined as the weighted sum of both kernels, $k(x_i, x_j) = \beta k_c(x_i, x_j) + (1 - \beta) k_c(x_i, x_j)$, where $\beta \in [0, 1]$ is a scalar parameter to be adjusted. Intuitively, the cluster kernel accounts for probabilistic similarities at small and large scales (number of clusters) between all samples along the data manifold. The method does not require computationally demanding procedures (e.g. current GMM clustering algorithms scale linearly with $n$), and the kMV A still relies just on the labeled data, and thus requires an $l \times l$ kernel matrix. All these properties are quite appealing from the practitioner’s point of view.

**IV. EXPERIMENTAL RESULTS**

In this section, we illustrate through different application examples the use and capabilities of the supervised kernel multivariate feature extraction framework. We start by comparing the performance of the linear and kernel MVA methods in a benchmark of classification problems from the publicly available Machine Learning Repository at University of California, Irvine (UCI). We then consider two real applications to show the potential of these algorithms: satellite image processing [40] and audio processing for music genre prediction [41]. The size of the data set used in this second scenario is sufficiently large to make standard kMV A methods impractical, a situation that we will use to illustrate the benefits of the sparse extensions.

**A. UCI repository benchmark**

Our first battery of experiments deals with standard benchmark data sets taken from the UCI repository, and will be

http://archive.ics.uci.edu/ml/.
TABLE III

MAIN PROPERTIES OF (K)MVA METHODS. COMPUTATIONAL COMPLEXITY AND IMPLEMENTATION ISSUES ARE CATEGORIZED FOR THE CONSIDERED DENSE AND SPARSE METHODS, INDICATING FROM LEFT TO RIGHT: THE FREE PARAMETERS, NUMBER OF KERNEL EVALUATIONS (KE) DURING TRAINING, STORAGE REQUIREMENTS, WHETHER AN EIGENPROBLEM (EIG) OR GENERALIZED EIGENPROBLEM (GEV) NEEDS TO BE SOLVED, AND THE NUMBER OF KERNELS THAT NEED TO BE EVALUATED TO EXTRACT PROJECTIONS FOR NEW DATA.

| Method   | Parameters KE tr. | Storage Req. EIG / GEV / Other KE test |
|----------|-------------------|----------------------------------------|
| PCA      | none none         | $O(d^2)$                                |
| PLS      | none none         | $O((d+m)\gamma)$ EIG                    |
| CCA      | none none         | $O((d+m)\gamma)$ GEV                    |
| OPLS     | none none         | $O(d^2)$                                |
| kPCA     | kernel $l^2$      | $O(l^2)$                                |
| kPLS     | kernel $l^2$      | $O((l+m)\gamma)$ EIG                    |
| kCCA     | kernel $l^2$      | $O((l+m)\gamma)$ GEV                    |
| kOPLS    | kernel $l^2$      | $O(l^2)$                                |
| skPCA    | kernel, $v_n$, $l^2$ | $O(l^2)$                              |
| rkPCA    | kernel, $r$, $l^2$ | $O(r^2)$                                |
| rkCCA    | kernel, $r$, $l^2$ | $O((r+m)^2)$ GEV                        |
| rkOPLS   | kernel, $r$, $l^2$ | $O(r^2)$                                |
| SMA / SMC | kernel, $r$, $l^2$ | $O(l^2)$                                |

† A maximum likelihood estimation step is required. ‡ By constraining the search to $p$ random samples at each step of the algorithm, kernel evaluations and storage requirements during training can be reduced to $r l p$ and $O(l p)$, respectively.

oriented to discuss some important properties of the standard linear and kernel MVA methods. The main properties of the selected problems are given in Table IV, namely the number of training and test patterns $(l, l_{test})$, the dimensionality of the input space $(d)$, the number of classes $(c)$, the ratio of training patterns per dimension, and the Kullback-Leibler divergence (KL) between the sample probabilities of each class and a uniform distribution, that can be seen as an indicator of balance among classes. The train/test partition has left 60% of the total data (or alternatively a maximum of 500 samples) in the training set, so that standard kMVA complexity is kept under control. Since all selected problems involve a classification task, matrix $Y$ was used to store the class information using 1-of-$c$ coding. To obtain the overall accuracies (OA) displayed in the table, we have trained an LS model to approximate $Y$, followed by a “winner-takes-all” activation function.

We start the discussion by comparing the performance of linear methods. For OPLS and CCA we have used the maximum number of features $(c-1)$, whereas for PCA $n_f$ has been fixed through a 10-fold cross-validation scheme on the training set, and is indicated next to the results of the method. When the maximum number of features are extracted, linear OPLS and CCA become identical. We can see that in most cases they perform similarly to PCA, but requiring significantly fewer features. It is important to see the very poor performance of OPLS and CCA in two particular problems: semeion and sonar. We see that the ratio $l/d$ is very small for these problems, so the input covariance matrix is likely to be ill-conditioned. Then, very low variance directions of the input space are used by CCA and OPLS to overfit the data. To avoid this problem, it becomes necessary to regularize these algorithms, e.g., by loading the main diagonal of the covariance matrix $C_x$ or by executing the method on the projections already extracted by PCA. The results of regularized OPLS and CCA following the latter approach, and using the maximum of $(c-1)$ features, are given in the last two columns of Table IV where we can see that the regularization does indeed help to overcome the “small-sample-size” problem. For completeness, we present also the results of the PLS2 approach. To get a fair comparison with the other supervised schemes, PLS2 is also trained to extract $(c-1)$ features. Thus, we can conclude that OPLS and CCA allow to obtain more discriminative features than the other methods, but one needs to be aware of the likely need to regularize the solution.

Next, we turn our attention to non-linear versions, whose results have been displayed in the bottom half of the table. An RBF kernel has been used in all cases, selecting the kernel width with 10-fold cross-validation. A first consideration is that kernel approaches considerably outperform the linear schemes. Since the RBF kernel implies a mapping to a very high dimensional space, it is not surprising that standard kOPLS and kCCA are even more prone to overfitting than before, this being a well-known problem of these methods. As before, regularized solutions allow kOPLS and kCCA to achieve comparable performance to kPCA, but retaining a much smaller number of features $(c-1)$, which demonstrates the superior discriminative capabilities of the features extracted by these methods. For completeness, kPLS2 was used to extract the same number of features as kOPLS and kCCA, achieving considerably smaller OAs in all problems. Nevertheless, with kPLS2 it is possible to extract a larger number of projections, and this method is known to be more robust to overfitting than kOPLS and kCCA.

B. Remote sensing image analysis

The last few hundred years human activities have precipitated an environmental crisis on Earth. In the last decade, advanced statistical methods have been introduced to quantify our impact on the land/vegetation and atmosphere, to better understand their interactions. Nowadays, multi- and hyperspectral sensors mounted on satellite or airborne platforms may acquire the reflected energy by the Earth with high spatial detail and in several wavelengths. Recent infrared sounders also allow us to estimate the profiles of atmospheric parameters with unprecedented accuracy and vertical resolution. Here,
we pay attention to the performance of several KMVA methods for both image segmentation of hyperspectral images, and estimation of climate parameters from infrared sounders.

a) Hyperspectral image classification: The first case study deals with image segmentation of hyperspectral images [42]. We have used the standard AVIRIS image taken over NW Indiana’s Indian Pine test site in June 1992. We removed 20 noisy bands covering the region of water absorption, and finally worked with 200 spectral bands. The high number of narrow spectral bands induce a high collinearity among features. Discriminating among the major crop classes in the area can be very difficult (in particular, given the moderate spatial resolution of 20 meters), which has made the scene a challenging benchmark to validate classification accuracy of hyperspectral imaging algorithms. The image is $145 \times 145$ pixels and contains 16 quite unbalanced classes (ranging from 20 – 2468 pixels). Among the available 10366 labeled pixels, 20% were used for training the feature extractors, and the remaining 80% for testing. The discriminative power of all extracted features was tested using a simple classifier consisting of a linear least squares model followed by a “winner-takes all” activation function.

Figure 2 shows the test classification accuracy for a varying number of extracted features, $n_f$. For linear models, OPLS performs better than all other methods for any number of extracted features. Even though CCA provides similar results to OPLS, it performs better than all other methods for any number of features. Even though CCA provides similar results to OPLS, it achieves virtually the same performance.

b) Temperature estimation from infrared sounding data: The second case study focuses on the estimation of temperature from spaceborne very high spectral resolution infrared sounding instruments. Despite the constant advances in sensor design and retrieval techniques, it is not trivial to invert the full information of the atmospheric state contained by such hyperspectral measurements. Statistical regression and feature extraction methods have overcome the numerical difficulties of radiative transfer models (RTMs), and enable fast retrievals from high volumes of data [43].

We concentrate here on the Infrared Atmospheric Sounding Interferometer (IASI) onboard the MetOp-A satellite data. IASI spectra consist of 8461 spectral channels (input features) with a spatial resolution of 25 km at nadir. Due to its large spatial coverage and low radiometric noise, IASI provides twice daily global measurements of key atmospheric species such as ozone, carbon monoxide, methane and methanol. Due to the impossibility to obtain real radiosound data for the whole atmospheric column, we resorted to the standard hybrid approach for developing the prediction models: we used synthetic data for training the models, and then applied it to a full IASI orbit (91800 ‘pixels’ on March 4th, 2008). A total amount of 67475 synthetic samples were simulated with an infrared RTM according to input profiles of temperature at 90 pressure levels.

We are confronted here with a challenging multi-output regression problem ($x_i \in \mathbb{R}^{8461}$ and $y_i \in \mathbb{R}^{90}$), which needs fast methods for retrieval (prediction). Note that the IASI mission delivers approximately $1.3 \times 10^6$ spectra per day,

### Table IV

| data set    | $t$ | $d$ | $l/d$ | $n_f$/PCA | PCA | PLS2 | OPLS | CCA | reg. OPLS | reg. CCA |
|-------------|-----|-----|-------|-----------|-----|------|------|-----|-----------|----------|
| car         | 500 | 6   | 83.3  | 197       | 93  | 92.1 | 93   | 91  | 91.6      | 91.6     |
| glass       | 128 | 9   | 14.2  | 9         | 57  | 57.3 | 57   | 57  | 57        | 57       |
| optdigits   | 500 | 62  | 8.06  | 27        | 88  | 80.9 | 86.6 | 86.6| 88.8      | 88.8     |
| semeion     | 500 | 256 | 1.95  | 78        | 83.8| 82.4 | 69.1 | 69.1| 83.8      | 83.8     |
| sonar       | 125 | 60  | 2.08  | 7         | 74  | 47.4 | 67.5 | 51  | 74.7      | 74.7     |
| vehicle     | 500 | 18  | 27.8  | 18        | 78  | 78.2 | 78   | 78  | 78        | 78       |
| vowel       | 500 | 13  | 38.5  | 33.8      | 48.9| 46.9 | 48.8 | 48.8| 48.8      | 48.8     |
| yeast       | 500 | 8   | 62.5  | 2.5       | 55  | 55   | 55   | 55  | 55        | 55       |

4The calibrated data is available online (along with detailed ground-truth information) from [http://dynamo.ecn.purdue.edu/~biehl/MultiSpec](http://dynamo.ecn.purdue.edu/~biehl/MultiSpec)
which gives a rate of about 29 Gbytes/day to be processed. We compared several MVA methods followed by LS in terms of the root-MSE (RMSE) computed as the discrepancy to the official European Centre for Medium-Range Weather Forecasts (ECMWF) estimations. We obtain the RMSE for all spatial ‘pixels’ in the orbit and for all layers of the atmosphere. We fixed a maximum $n_f = 100$, and used only $l = 2000$ samples for learning the transformation and the LS regression weights. For the kernel approaches, an RBF kernel is used, selecting the width using 5-fold cross validation.

The left panel of Fig. 2 illustrates the RMSE obtained for different pressure levels, using a spatial average over all the pixels. Results show that kernel methods outperform linear approaches in RMSE terms, specially in the lower parts of the atmosphere, probably due to the presence of clouds and haze. Estimated surface temperature maps for a particular area are also given in the right panel of Fig. 2 for PCA, kPCA, and kOPLS. These plots reveal that kernel methods yield closer maps to those provided by the ECMWF in averaged RMSE terms. These results can be of high value because the kernel-based estimations are obtained with a drastic reduction in computational time compared to the physical-inversion methods used by the ECMWF.

C. Analysis of integrated short-time music features for genre prediction

In this subsection, we consider the problem of predicting the genre of a song using the audio data only, a task which has recently been a subject of much interest. The data set we analyze has been previously used in [26], [41], and consists of 1317 snippets of 30 seconds distributed evenly over 11 music genres: alternative, country, easy listening, electronica, jazz, latin, pop&dance, rap&hip-hop, r&b, reggae and rock. This is a rather complex data set with an average of 1.83 songs per artist. An estimate of human performance on this data set has been carried out via subjective tests, providing an average accuracy rate around 55%.

The music snippets are MP3 (MPEG1-layer3) encoded music with a bitrate of 128 kbps or higher, down sampled to 22050 Hz, and they are processed following the method in [41]: Mel Frequency Cepstral Coefficients (MFCC) features are extracted from overlapping frames of the song, using a
window size of 20 ms. Then, a multivariate autoregressive (AR) model is adjusted for every 1.2 seconds of the song to capture temporal correlation, and finally the parameters of the AR model are stacked into a 135 length feature vector for every such frame.

For training and testing the system we have split the data set into two subsets with 817 and 500 songs, respectively. After processing the audio data, we have 57388 36556 135-dimensional vectors in the training and test partitions, an amount which for standard kernel MVA methods is prohibitively large. For this reason, in this subsection we study the performance of linear MVA methods, as well as of sparse kernel methods that promote sparsity following the approach of \cite{26}: rkPCA, rkOPLS, and rkCCA. For completeness, we have also considered the kPLS2 method of \cite{17}: in this case the deflation scheme does not allow to use reduced set methods, so we applied mere subsampling. As in the previous subsections, we use an RBF kernel, with a 10-fold validation scheme over the training data to adjust the kernel width. We also resorted to an LS scheme followed by “winner-takes-all” to carry out the classification.

Figure 4 illustrates the performance of all methods for different number of features. For the sparse methods, the influence of the machine size, measured as the number of kernel evaluations required to extract features for new data, is also analyzed. In rkPCA, rkOPLS and rkCCA this is given by the cardinality of the reduced set, whereas in kPLS2 it coincides with the number of samples used to train the feature extractor. Since every song consists of about 70 AR vectors, we can measure the classification accuracy in two different ways:

1) On the level of individual AR vectors, or 2) by majority voting across the AR vectors of a given song. The left panel of Fig. 4 illustrates the discriminative power of the features extracted by all considered methods. Overall, the best performance is obtained by OPLS- and CCA-type methods, with the kernel schemes outperforming the linear ones for $n_f > 5$. The poor performance of PCA and rkPCA makes evident the need of exploiting label information during the feature extraction step. Finally, it is evident that a mere subsampling does not provide kPLS2 with enough data to extract relevant features, and this method performs worse than its linear counterpart.

On the right panel of Fig. 4 we analyze the accuracy of kernel methods as a function of machine size. As before, it is clear that rkOPLS and rkCCA are the best performing methods both at the AR and song levels. Increasing the size of the machine results in better performance, although the improvement is not very noticeable in excess of 250 nodes. Altogether, these results allow us to conclude that sparse methods can be used to enhance the applicability of kMVA for large data sets. In this subsection, we have focused on the sparse-promotion technique of \cite{26}, but similar advantages can be expected from other sparse approaches.

V. Conclusions

We reviewed the field of supervised feature extraction from the unified framework of multivariate analysis. The use of these techniques in real world applications is becoming increasingly popular. Beyond standard PCA, there is a plethora of linear and kernel methods that are generally better suited for supervised applications since they seek for projections that maximize the alignment with the target variables. We analyzed the commonalities and basic differences of the most representative MVA approaches in the literature, as well as the relationships to existing kernel discriminative feature extraction and statistical dependence estimation approaches. We also studied recent methods to make kernel MVA more suitable to real life applications, both for large scale data sets and for problems with few labeled data. In such approaches, sparse and semi-supervised learning extensions have been successfully introduced for most of the models. Actually, seeking for the appropriate features that facilitate classification or regression cuts to the heart of manifold learning. We have illustrated MVA methods in challenging real problems that typically exhibit complex manifolds. A representative subset of the UCI repository has been used to illustrate the general applicability of the standard MVA implementations in moderate data sizes. We have completed the panorama with challenging real-life applications: the prediction of music genre from recorded signals, and the classification of land-cover classes and estimation of climate variables to monitor our Planet. This tutorial presented the framework of kernel multivariate methods and outlined relations and possible extensions. The adoption of the methods in many disciplines of science and engineering is nowadays a fact. New exciting advances in the theory and applications are yet to come.

Acknowledgments

The authors would like to thank the reviewers and the special issue editors for useful comments that improved the presentation of the present manuscript significantly.

This work was partially supported by Banco Santander and Universidad Carlos III de Madrid’s Excellence Chair programme, and by the Spanish Ministry of Economy and Competitiveness (MINECO) under projects TIN2012-38102-C03-01, TEC2011-22480 and PRI-PIBIN-2011-1266.

References

[1] H. Wold, “Estimation of principal components and related models by iterative least squares,” in P. R. Krishnaiah (ed.) Multivariate Analysis, pp. 391–420, Academic Press, 1966.
[2] R. Rosipal, “Nonlinear Partial Least Squares: An Overview,” in Chemoinformatics and Advanced Machine Learning Perspectives: Complex Computational Methods and Collaborative Techniques, H. Lodhi and Y. Yamanishi (eds.), ACCM, IGI Global, pp. 169–189, 2011.
[3] S. Wold, N. Kettaneh-Wold, and B. Skagerberg, “Nonlinear PLS Modeling,” Chemometrics and Intelligent Laboratory Systems, vol. 7, pp. 53–65, 1989.
[4] S. Qin and T. McAvoy, “Non-linear PLS modelling using neural networks,” Computers & Chemical Engineering, vol. 16, pp. 579–391, 1992.
[5] B. E. Boser, I. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in Proc. COLT’92, Pittsburgh, PA, 1992, pp. 144–152.
[6] B. Schölkopf, A. Smola and K.-R. Muller, “Nonlinear Component Analysis as a Kernel Eigenvalue Problem,” Neural Computation, vol. 10, pp. 1299–1319, 1998.
[7] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.
Fig. 4. Genre classification accuracy of linear and sparse kernel MVA methods. Left: Accuracy at the song level as a function of the number of extracted features. Right: Accuracy of the sparse methods measured as percentage of correctly classified AR patterns and songs for different machine sizes.